Benutzer-Leitfaden

Amazon EKS



Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon EKS: Benutzer-Leitfaden

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Die Marken und Handelsmarken von Amazon dürfen nicht in einer Weise in Verbindung mit nicht von Amazon stammenden Produkten oder Services verwendet werden, die geeignet ist, Kunden irrezuführen oder Amazon in irgendeiner Weise herabzusetzen oder zu diskreditieren. Alle anderen Handelsmarken, die nicht Eigentum von Amazon sind, gehören den jeweiligen Besitzern, die möglicherweise zu Amazon gehören oder nicht, mit Amazon verbunden sind oder von Amazon gesponsert werden.

Table of Contents

... xxvi
... xxvii
Was ist Amazon EKS? ... 1
Amazon EKS: Vereinfachtes Kubernetes-Management ... 1
Aufbau und Skalierung mit Kubernetes: Amazon EKS-Funktionen ... 2
Features von Amazon EKS ... 3
Zugehörige Services ... 4
Preisgestaltung von Amazon EKS ... 5
Häufige Anwendungsfälle ... 6
Architektur ... 8
Steuerebene ... 8
Datenverarbeitung ... 8
EKS-Funktionen ... 10
Kubernetes-Konzepte ... 11
Warum Kubernetes? ... 12
Cluster ... 17
Workloads ... 22
Nächste Schritte ... 28
Optionen für die Bereitstellung ... 29
Bereitstellungsoptionen von Amazon EKS kennenlernen ... 29
Amazon EKS in der Cloud ... 29
Amazon EKS in Ihrem Rechenzentrum oder in Ihren Edge-Umgebungen ... 30
Amazon EKS Anywhere für Air-Gapped-Umgebungen ... 31
Amazon-EKS-Tools ... 32
Einrichten ... 33
Nächste Schritte ... 33
AWS-CLI einrichten ... 33
So erstellen Sie einen Zugriffsschlüssel ... 34
Konfigurieren der AWS-CLI ... 34
So rufen Sie ein Sicherheitstoken ab ... 35
So überprüfen Sie die Benutzeridentität ... 35
Nächste Schritte ... 36
kubectl und eksctl einrichten ... 36
Installieren oder Aktualisieren von `kubectl` ... 36

Schritt 1: Überprüfen, ob kubectl installiert ist ... 37
Step 2: kubectl installieren oder aktualisieren ... 37
Installieren eksctl ... 52
Nächste Schritte ... 52
Quickstart ... 53
In diesem Tutorial ... 53
Voraussetzungen ... 53
Konfigurieren des Clusters ... 54
Erstellen IngressClass ... 54
Bereitstellung der Beispielanwendung für das Game 2048 ... 55
Persistente Daten unter Verwendung von Amazon EKS Auto Mode ... 58
Bereinigen ... 60
Amazon EKS kennenlernen ... 61
Übersicht ... 61
Workshop zu Amazon EKS ... 62
Amazon-EKS-Tutorials zur praktischen Einrichtung von Clustern ... 63
Amazon-EKS-Beispiele ... 64
AWS-Tutorials ... 65
Workshop für Entwickler ... 65
Terraform-Workshop ... 65
AWS-Amazon-EKS-Training ... 66
Erste Schritte ... 67
Cluster erstellen (EKS Auto Mode) ... 67
Cluster erstellen (eksctl) ... 68
Voraussetzungen ... 69
Schritt 1: Erstellen Sie Ihre Amazon-EKS-Cluster und -Knoten ... 69
Schritt 2: Anzeigen von Kubernetes-Ressourcen ... 71
Schritt 3: Löschen Ihrer Cluster und Knoten ... 73
Nächste Schritte ... 73
Cluster erstellen (Konsole und CLI) ... 73
Voraussetzungen ... 74
Schritt 1: Erstellen Sie Ihre Amazon-EKS-Cluster ... 75
Schritt 2: Konfigurieren Sie Ihren Computer für die Kommunikation mit Ihrem Cluster ... 78
Schritt 3: Erstellen von Knoten ... 79
Schritt 4: Ressourcen anzeigen ... 81
Schritt 5: Löschen von Ressourcen ... 81

Nächste Schritte ... 82
EKS Auto Mode ... 84
Features ... 84
Automatisierte Komponenten ... 86
Konfiguration ... 87
Modell der geteilten Verantwortung ... 88
Cluster erstellen ... 89
eksctl CLI ... 90
AWS CLI ... 92
Management-Konsole ... 100
Vorhandene Cluster aktivieren ... 102
Migrationsreferenz ... 103
Migration von EBS-Volumes ... 104
Migration von Load Balancern ... 105
Cluster aktivieren ... 105
Von Karpenter migrieren ... 110
Von MNGs migrieren ... 113
Von Fargate migrieren ... 114
Workloads ausführen ... 120
Erhöhte Workload bereitstellen ... 121
Load Balancer bereitstellen ... 124
Workload mit Statusverwaltung bereitstellen ... 129
Beschleunigten Workload bereitstellen ... 135
Konfiguration ... 142
Knotenklasse erstellen ... 146
Knoten-Pool erstellen ... 155
Knotenpools mit statischer Kapazität ... 161
Ingress-Klasse erstellen ... 168
Service erstellen ... 177
Erstellen StorageClass ... 183
EKS Auto Mode deaktivieren ... 193
Kubernetes-Version aktualisieren ... 195
Integrierte Knoten-Pool überprüfen ... 196
Kontrollbereitstellung ... 198
Kritische Add-Ons ausführen ... 200
Netzwerkrichtlinien verwenden ... 202

Subnetze kennzeichnen 213
CIS-Bericht generieren 215
Root-Volumes verschlüsseln 218
AMI-Steuerungen aktualisieren 222
Steuerung von Kapazitätsreservierungen 226
Local Zones verwenden 231
Erweiterte Sicherheit 235
Funktionsweise 237
Verwaltete Instances 238
Identität und Zugriff 242
Netzwerk 248
Fehlerbehebung 252
Knotenüberwachungsagent 254
Rufen Sie mithilfe der AWS EC2 CLI die Konsolenausgabe von einer EC2 verwalteten Instanz ab 254
Knotenprotokolle mithilfe von Debug-Containern und der `kubectl`-CLI abrufen 255
Zeigen Sie die mit dem automatischen Modus von EKS verknüpften Ressourcen in der AWS Konsole an 256
IAM-Fehler in Ihrem AWS Konto anzeigen 256
Fehlerbehebung bei fehlgeschlagener Pod-Planung in Knoten im Automatikmodus 257
Fehlerbehebung für Knoten, die nicht dem Cluster beitreten 257
Freigabe von Volumes über Pods hinweg 260
Anzeige von Karpenter-Ereignissen in den Protokollen der Steuerebene 260
Fehlerbehebung für im Automatikmodus enthaltene Controller 262
Zugehörige Ressourcen 218
Versionshinweise 263
19. Dezember 2025 263
19. November 2025 263
19. November 2025 263
23. Oktober 2025 264
1. Oktober 2025 264
30. September 2025 264
29. September 2025 264
10. September 2025 264
24. August 2025 264
15. August 2025 264

06. August 2025 ... 265
30. Juni 2025 ... 265
20. Juni 2025 ... 265
13. Juni 2025 ... 265
30. April 2025 ... 265
18. April 2025 ... 265
11. April 2025 ... 266
4. April 2025 ... 266
31. März 2025 ... 266
21. März 2025 ... 266
14. März 2025 ... 266
Cluster konfigurieren ... 267
Automatik-Cluster erstellen ... 267
Voraussetzungen ... 53
Cluster erstellen — AWS Konsole ... 269
Cluster erstellen — AWS CLI ... 274
Nächste Schritte ... 100
Erstellen eines Clusters ... 280
Voraussetzungen ... 53
Schritt 1: Cluster-IAM-Rolle erstellen ... 281
Schritt 2: Cluster erstellen ... 282
Schritt 3: kubeconfig aktualisieren ... 292
Schritt 4: Cluster-Einrichtung ... 293
Nächste Schritte ... 100
Eine bereitgestellte Kontrollebene erstellen ... 295
-Übersicht ... 98
Anwendungsfälle ... 152
Skalierungsstufen auf der Steuerungsebene ... 298
Überlegungen ... 105
Erste Schritte mit der Provisioned Control Plane ... 302
Cluster-Einblicke ... 310
Cluster-Erkenntnis-Typen ... 310
Überlegungen ... 310
Anwendungsfälle ... 310
Erste Schritte ... 312
Cluster-Erkenntnisse anzeigen ... 312

Kubernetes-Version aktualisieren 319
Überlegungen zu Amazon EKS Auto Mode 320
Zusammenfassung 321
Schritt 1: Upgrade vorbereiten 321
Schritt 2: Überlegungen zum Upgrade prüfen 322
Schritt 3: Cluster-Steuerebene aktualisieren 323
Schritt 4: Cluster-Komponenten aktualisieren 326
Herabstufen der Kubernetes-Version für einen Amazon-EKS-Cluster 327
Einen Cluster löschen 327
Überlegungen 105
Cluster löschen (eksctl) 328
Cluster löschen (AWS Konsole) 329
Cluster löschen (AWS CLI) 330
Schützen Ihrer EKS-Cluster vor versehentlichem Löschen 332
Cluster-Endpunktzugriff 333
IPv6-Cluster-Endpunkt-Format 334
IPv4-Cluster-Endpunkt-Format 334
Privater Cluster-Endpunkt 335
Ändern des Cluster-Endpunktzugriffs 336
Zugriff auf einen privaten API-Server 340
Endpunktzugriff konfigurieren 341
Windows-Support aktivieren 345
Überlegungen 105
Voraussetzungen 53
Windows-Support aktivieren 346
Windows-Pods bereitstellen 349
Unterstützung einer höheren Pod-Dichte in Windows-Knoten 349
Windows-Support deaktivieren 350
Private Cluster 350
Anforderungen an die Cluster-Architektur 351
Knoten-Anforderungen 351
Pod-Anforderungen 353
Auto Scaling 355
EKS Auto Mode 355
Weitere Lösungen 356
Weitere Informationen zur Zonenverschiebung 357

Verständnis des Ost-West-Netzwerkdatenverkehrs zwischen Pods ... 357
Anforderungen der EKS-Zonenverschiebung ... 362
Häufig gestellte Fragen ... 371
Weitere Ressourcen ... 374
Zonenverschiebung aktivieren ... 374
Überlegungen ... 375
Was ist Amazon Application Recovery Controller? ... 375
Was ist eine Zonenverschiebung? ... 376
Was ist die automatische Zonenverschiebung? ... 376
Was unternimmt EKS während einer automatischen Verschiebung? ... 376
Registrieren Sie den EKS-Cluster beim Amazon Application Recovery Controller (ARC) (AWS Konsole) ... 377
Nächste Schritte ... 100
Zugriff verwalten ... 378
Allgemeine Aufgaben ... 378
Hintergrund ... 213
Überlegungen zu EKS Auto Mode ... 328
Kubernetes-API-Zugriff ... 380
IAM-Identitäten Kubernetes-Berechtigungen zuordnen ... 381
Einstellen des Cluster-Authentifizierungsmodus ... 382
Zugriffseinträge ... 383
aws-auth ConfigMap ... 443
OIDC-Anbieter verknüpfen ... 453
OIDC-Anbieter trennen ... 459
Zugriff auf Cluster-Ressourcen ... 459
Erforderliche Berechtigungen ... 460
Zugriff auf den Cluster mit kubectl ... 466
`kubeconfig`-Datei automatisch erstellen ... 467
Workload-Zugriff auf AWS ... 469
Servicekonto-Tokens ... 469
Cluster-Add-ons ... 471
Gewährung von AWS Identity and Access Management für Workloads in Clustern von Amazon Elastic Kubernetes Service ... 471
Anmeldeinformationen mit IRSA ... 476
Pod Identity ... 501
Verwaltung von Rechenleistung ... 538

Vergleich von Rechenoptionen 538
Verwaltete Knotengruppen 545
Konzepte für verwaltete Knotengruppen 546
Kapazitätstypen für verwaltete Knotengruppen 549
Create 553
Aktualisierung 565
Verhaltensdetails aktualisieren 570
Startvorlagen 575
Delete 593
Selbstverwaltete Knoten 595
Amazon Linux 597
Flaschenrakete 606
Windows 610
Ubuntu Linux 620
Update-Methoden 623
AWS-Fargate 637
Überlegungen zu AWS-Fargate 638
Fargate-Vergleichstabelle 641
Erste Schritte 643
Profile definieren 649
Profile löschen 654
Details zur Pod-Konfiguration 656
Betriebssystem-Patching-Ereignisse 659
Metriken erfassen 662
Protokollierung 664
Amazon EC2-Instance-Typen 676
Von Amazon EKS empfohlene maximale Pods für jeden Amazon-EC2-Instance-Typ 678
Überlegungen zu EKS Auto Mode 328
Vorgefertigte optimierte AMIs 680
AL2 AMI-Abwertung 681
Amazon Linux 687
Bottlerocket 698
Ubuntu Linux 704
Windows 704
Knotenintegrität 776
Knotenüberwachungsagent 776

Automatische Knotenreparatur ... 776
Probleme mit dem Zustand des Knotens ... 779
Knotenzustand anzeigen ... 791
Knotenprotokolle abrufen ... 794
Hybridknoten ... 797
Features ... 797
Grenzwerte ... 798
Überlegungen ... 798
Weitere Ressourcen ... 799
Voraussetzungen ... 799
Hybridknoten ausführen ... 870
Konfigurieren ... 893
Funktionsweise ... 983
nodeadm für Hybridknoten ... 1027
Fehlerbehebung ... 1047
Anwendungsdatenspeicher ... 1070
Amazon EBS ... 1070
Überlegungen ... 1071
Voraussetzungen ... 1071
Schritt 1: Erstellen einer IAM-Rolle ... 1072
Schritt 2: Amazon-EBS-CSI-Treiber herunterladen ... 1080
Schritt 3: Beispielanwendung bereitstellen ... 1081
Amazon EFS ... 1081
Überlegungen ... 1082
Voraussetzungen ... 1082
Schritt 1: Erstellen einer IAM-Rolle ... 1083
Schritt 2: Amazon-EFS-CSI-Treiber herunterladen ... 1089
Schritt 3: Erstellen eines Amazon EFS-Dateisystems ... 1090
Schritt 4: Beispielanwendung bereitstellen ... 1090
Amazon FSx für Lustre ... 1090
Treiber bereitstellen ... 1090
Optimieren (EFA) ... 1098
Optimieren (nicht-EFA) ... 1115
Amazon FSx für NetApp ONTAP ... 1122
Amazon FSx für OpenZFS ... 1123
Amazon-Datei-Cache ... 1124

Mountpoint für Amazon S3 ... 1124
Überlegungen ... 1124
Treiber bereitstellen ... 1125
Treiber entfernen ... 1136
CSI-Snapshot-Controller ... 1137
Netzwerk konfigurieren ... 1139
Hinzufügen eines vorhandenen VPC-Subnetzes zu einem Amazon-EKS-Cluster über die Management-Konsole ... 1139
VPC- und Subnetz-Anforderungen ... 1139
VPC-Anforderungen und -Überlegungen ... 1140
Subnetz-Anforderungen und -Überlegungen ... 1142
Anforderungen und Überlegungen für gemeinsam genutzte Subnetze ... 1149
Erstellen einer VPC ... 1150
Voraussetzungen ... 53
Öffentliche und private Subnetze ... 1151
Nur öffentliche Subnetze ... 1153
Nur private Subnetze ... 1155
Anforderungen an Sicherheitsgruppen ... 1157
Standard-Cluster-Sicherheitsgruppe ... 1157
Einschränken des Clusterverkehrs ... 1159
Freigegebene Sicherheitsgruppen ... 1160
Netzwerk-Add-Ons verwalten ... 1161
Integrierte Add-Ons ... 1161
Optionale AWS Netzwerk-Add-Ons ... 1162
Amazon VPC CNI ... 1163
Abweichende CNI-Plugins ... 1294
Multus ... 1295
AWS Load Balancer Balancer-Controller ... 1297
CoreDNS ... 1316
kube-proxy ... 1335
Workloads ... 1341
Beispiel-Bereitstellung (Linux) ... 1341
Voraussetzungen ... 53
Namespaces erstellen ... 1342
Erstellen Sie eine Kubernetes-Bereitstellung. ... 1342
Erstellen eines Services ... 1344

Erstellte Ressourcen überprüfen 1345
Shell in einem Pod ausführen 1348
Nächste Schritte 1349
Beispiel-Bereitstellung (Windows) 1349
Voraussetzungen 53
Namespaces erstellen 1342
Erstellen Sie eine Kubernetes-Bereitstellung. 1342
Erstellen eines Services 1344
Erstellte Ressourcen überprüfen 1353
Shell in einem Pod ausführen 1348
Nächste Schritte 1357
Vertical Pod Autoscaler 1357
Bereitstellen des Vertical Pod Autoscalers 1357
Testen der Installation von Vertical Pod Autoscaler 1359
Horizontal Pod Autoscaler 1363
Ausführen einer Horizontal Pod Autoscaler-Testanwendung 1364
Netzwerk-Load-Balancer 1366
Voraussetzungen 53
Überlegungen 105
Erstellen eines Network Load Balancers 1370
(Optional) Bereitstellen einer Beispielanwendung 1373
Application Load Balancing 1377
Voraussetzungen 53
Wiederverwendung mit Ingress-Gruppen ALBs 1380
(Optional) Bereitstellen einer Beispielanwendung 1382
Externe IP-Adressen für Service einschränken 1385
Kopieren eines Images in ein Repository 1387
Amazon-Image-Registries anzeigen 1391
Amazon-EKS-Add-ons 1394
Überlegungen 1395
Benutzerdefinierter Namespace für Add-Ons 1396
Überlegungen zu Amazon EKS Auto Mode 1397
Support 1398
AWS Add-Ons 1400
Community-Erweiterungen 1421
Marketplace-Add-Ons 1428

Add-On erstellen ... 1445
Add-On aktualisieren ... 1459
Kompatibilität überprüfen ... 1467
Add-On entfernen ... 1469
IAM roles ... 1473
Felder, die angepasst werden können ... 1480
Überprüfen von Container-Images ... 1484
EKS-Fähigkeiten ... 1485
Verfügbare Funktionen ... 1486
AWS Controller für Kubernetes (ACK) ... 1486
Argo CD ... 1486
kro (Kube Resource Orchestrator) ... 1487
Vorteile der EKS-Fähigkeiten ... 1487
Preisgestaltung ... 1488
So funktionieren die Funktionen von EKS ... 1488
Häufige Anwendungsfälle ... 1489
Mit Fähigkeiten arbeiten ... 1491
Ressourcen für EKS-Fähigkeiten ... 1491
Den Status der Fähigkeit verstehen ... 1491
Fähigkeiten schaffen ... 1493
Funktionen auflisten ... 1493
Beschreiben Sie eine Fähigkeit ... 1495
Aktualisieren Sie die Konfiguration einer Funktion ... 1496
Löschen Sie eine Funktion ... 1497
Nächste Schritte ... 100
Kubernetes-Ressourcen ... 1498
Ressourcen von Argo CD ... 1499
Kro-Ressourcen ... 1500
ACK-Ressourcen ... 1501
Ressourcenlimits ... 1502
Nächste Schritte ... 100
Überlegungen ... 1503
Capability: IAM-Rollen und Kubernetes RBAC ... 1503
Architekturmuster mit mehreren Clustern ... 1504
Vergleich der EKS-Funktionen mit selbstverwalteten Lösungen ... 1506
AWS Controller für Kubernetes ... 1509

Wie funktioniert ACK 1509
Vorteile von ACK 1510
Unterstützte AWS Dienste 1511
Integration mit anderen von EKS verwalteten Funktionen 1511
Erste Schritte mit ACK 1512
ACK-Fähigkeit erstellen 1512
ACK-Konzepte 1523
Konfigurieren von -Berechtigungen 1531
Überlegungen 1539
Fehlerbehebung 1547
Vergleich mit selbstverwaltetem System 1551
Argo CD 1553
Wie funktioniert Argo CD 1553
Vorteile von Argo CD 1555
Integration mit AWS Identity Center 1555
Integration mit anderen von EKS verwalteten Funktionen 1511
Erste Schritte mit Argo CD 1556
Erstellen der Argo CD-Funktion 1556
Argo-CD-Konzepte 1568
Konfigurieren von -Berechtigungen 1574
Arbeiten mit Argo CD 1587
Überlegungen zu Argo CD 1632
Fehlerbehebung 1642
Vergleich mit selbstverwalteter Software 1646
kro 1652
Wie funktioniert Kro 1652
Vorteile von Kro 1653
Integration mit anderen von EKS verwalteten Funktionen 1511
Erste Schritte mit Kro 1654
Erstellen Sie eine Kro-Fähigkeit 1655
Kro-Konzepte 1667
Konfigurieren von -Berechtigungen 1674
Überlegungen 1679
Fehlerbehebung 1685
Vergleich mit selbstverwaltetem System 1691
Problembehebung bei Funktionen 1693

Allgemeiner Ansatz zur Fehlerbehebung 1693
Überprüfen Sie den Zustand der Funktionen 1693
Allgemeine Status der Fähigkeiten 1694
Überprüfen Sie den Kubernetes-Ressourcenstatus 1695
Überprüfen Sie die IAM-Berechtigungen und den Clusterzugriff 1696
Funktionsspezifische Problembehebung 1698
Häufig auftretende Probleme in allen Funktionen 1698
Nächste Schritte 100
Clusterverwaltung 1700
Kostenüberwachung 1700
Kosten pro Pod anzeigen 1701
Kubecost installieren 1702
Zugriff auf das Kubecost-Dashboard 1704
Weitere Informationen zu Kubecost 1705
Kennzahlen-Server 1719
Überlegungen 105
Bereitstellung als Community-Add-On mit Amazon-EKS-Add-Ons 1719
Bereitstellung mit Manifest 1720
Apps mit Helm bereitstellen 1721
Markieren Ihrer -Ressourcen 1723
Grundlagen zu Tags (Markierungen) 1723
Markieren Ihrer -Ressourcen 1724
Tag-Einschränkungen 1725
Markieren von Ressourcen für die Fakturierung 1726
Arbeiten mit Tags über die Konsole 1727
Arbeiten mit Tags mittels CLI, API oder `eksctl` 1728
Servicekontingente 1730
EKS-Servicekontingente finden Sie in der AWS-Managementkonsole 1730
EKS-Dienstkontingente mit der AWS CLI anzeigen 1730
Amazon-EKS-Service-Quotas 1731
AWS Fargate-Servicekontingente 1732
Sicherheit 1733
Bewährte Methoden 1734
Schwachstellen analysieren 1735
Der Center for Internet Security (CIS)-Benchmark für Amazon EKS 1735
Amazon-EKS-Plattformversionen 1735

Liste der Betriebssystem-Schwachstellen 1736
Knoten-Erkennung mit Amazon Inspector 1736
Cluster- und Knoten-Erkennung mit Amazon GuardDuty 1736
Compliance validieren 1736
Überlegungen zu EKS 1738
Sicherheit der Infrastruktur 1738
Ausfallsicherheit 1743
Serviceübergreifende Confused-Deputy-Prävention 1744
Überlegungen zu Kubernetes 1746
Zertifikatsignierung 1746
-Standardrollen und Benutzer 1749
Geheime Verschlüsselung aktivieren 1754
AWS Secrets Manager 1759
Standardmäßige Umschlagverschlüsselung für alle Kubernetes-API-Daten 1759
Überlegungen zu EKS Auto 1768
API-Sicherheit und Authentifizierung 1769
Netzwerksicherheit 1769
Sicherheit für verwaltete EC2-Instances 1769
Automatisiertes Ressourcenmanagement 1771
Bewährte Methoden für die Gewährleistung der Sicherheit 1771
Überlegungen zu den Funktionen von EKS 1772
Fähigkeit: IAM-Rolle 1772
EKS-Zugriffseinträge 1773
Zusätzliche Kubernetes-Berechtigungen 1775
Je nach Funktion sind IAM-Berechtigungen erforderlich 1776
Bewährte Methoden für die Gewährleistung der Sicherheit 1777
Nächste Schritte 100
IAM-Referenz 1781
Zielgruppe 1781
Authentifizierung mit Identitäten 1781
Verwalten des Zugriffs mit Richtlinien 1785
Amazon EKS und IAM 1788
Identitätsbasierte Richtlinien 1793
Service-verknüpfte Rollen 1800
IAM-Rolle zur Pod-Ausführung 1818
Konnektor-IAM-Rolle 1823

AWS verwaltete Richtlinien ... 1828
Fehlerbehebung ... 1859
Cluster-IAM-Rolle ... 1861
Knoten-IAM-Rolle ... 1865
IAM-Rolle für Cluster im Automatikmodus ... 1871
IAM-Rolle für Knoten im Automatikmodus ... 1875
Funktion: IAM-Rolle ... 1878
Cluster überwachen ... 1885
Überwachung und Protokollierung in Amazon EKS ... 1886
Überwachungs- und Protokollierungs-Tools für Amazon EKS ... 1887
Dashboard zur Beobachtbarkeit ... 1891
Zusammenfassung ... 1891
Cluster-Integrität ... 1891
Überwachen der Steuerebene ... 1892
Cluster-Erkenntnisse ... 1894
Probleme mit dem Zustand des Knotens ... 1895
EKS-Funktionen ... 1895
Beobachtbarkeit von Container-Netzwerken ... 1895
Anwendungsfälle ... 152
Features ... 84
Erste Schritte ... 312
Voraussetzungen und wichtige Hinweise ... 1898
Verwenden von AWS CLI, EKS-API und NFM-API ... 1899
Verwendung von Infrastruktur als Code (IaC) ... 1901
Funktionsweise ... 205
Überlegungen und Einschränkungen ... 1912
Prometheus-Metriken ... 1913
Schritt 1: Prometheus-Metriken aktivieren ... 1914
Schritt 2: Prometheus-Metriken verwenden ... 1916
Schritt 3: Prometheus-Scraper verwalten ... 1916
Bereitstellung mit Helm ... 1917
Steuerebene ... 1920
Amazon CloudWatch ... 1928
Grundlegende Kennzahlen in Amazon CloudWatch ... 1928
Amazon CloudWatch Observability-Betreiber ... 1938
Protokolle der Steuerebene ... 1938

Aktivierung oder Deaktivierung der Steuerebenen-Protokolle ... 1940
Cluster-Steuerebenen-Protokolle anzeigen ... 1942
AWS CloudTrail ... 1944
Referenzen ... 1944
Protokoll-Dateieinträge ... 1945
Metriken zu Auto-Scaling-Gruppen ... 1948
ADOT Operator ... 1949
Amazon-EKS-Dashboard ... 1949
Was ist das Amazon-EKS-Dashboard? ... 1949
Wie verwendet das Dashboard AWS Organizations? ... 1950
Aktivieren Sie das EKS-Dashboard über die AWS Konsole ... 1951
EKS-Dashboard anzeigen ... 1952
Dashboard konfigurieren ... 1952
Deaktivieren Sie das EKS-Dashboard über die AWS Konsole ... 1954
Fehlerbehebung im EKS-Dashboard ... 1954
Organizations konfigurieren ... 1955
Arbeiten mit anderen -Services ... 1961
AWS Backup ... 1961
AWS CloudFormation ... 1962
Amazon-EKS- und AWS-CloudFormation-Vorlagen ... 1962
Weitere Informationen zu AWS CloudFormation ... 1963
AWS CodeConnections ... 1963
Zur Verwendung AWS CodeConnections mit Argo CD ... 1963
Überlegungen zur Verwendung mit Argo CD CodeConnections ... 1964
Amazon Detective ... 1965
Verwenden Sie Amazon Detective mit Amazon EKS ... 1965
Amazon GuardDuty ... 1966
AWS Local Zones ... 1967
AWS Resilience Hub ... 1968
AWS Secrets Manager ... 1968
Verwenden Sie AWS Secrets Manager mit Argo CD ... 1969
Amazon Security Lake ... 1971
Vorteile der Verwendung von Security Lake mit Amazon Amazon EKS ... 1971
Aktivierung von Security Lake für Amazon EKS ... 1972
Analyse von EKS-Protokollen in Security Lake ... 1972
Amazon VPC Lattice ... 1973

Amazon EKS Connector 1974
Überlegungen zum Amazon EKS Connector 1974
Erforderliche IAM-Rollen für Amazon EKS Connector 1975
Cluster verbinden 1975
Überlegungen 1976
Voraussetzungen 1976
Schritt 1: Registrieren des Clusters 1976
Schritt 2: Installieren des `eks-connector`-Agents 1979
Nächste Schritte 1981
Zugriff auf Cluster gewähren 1981
Voraussetzungen 1982
Einen Cluster abmelden 1983
Abmelden des Kubernetes-Clusters 1984
Ressourcen in Ihrem Kubernetes-Cluster bereinigen 1985
Fehlerbehebung für EKS Connector 1985
Grundlegende Fehlersuche 1985
Helm-Ausgabe: 403 Forbidden 1987
Konsolenfehler: Der Cluster steckt im Status „Ausstehend“ fest 1988
Konsolenfehler: Benutzer „system:serviceaccount:eks-connector:eks-connector“ kann keine Ressourcen-Benutzer in der API-Gruppe auf Cluster-Ebene imitieren. 1988
Konsolenfehler: […] ist verboten: Benutzer […] kann Ressource […] nicht in der API-Gruppe im Cluster-Bereich auflisten 1989
Konsolenfehler: Amazon EKS kann nicht mit dem API-Server Ihres Kubernetes-Clusters kommunizieren. Der Cluster muss sich im ACTIVE-Status befinden, um eine erfolgreiche Verbindung zu erreichen. Try again in few minutes. (Versuchen Sie es in ein paar Minuten erneut.) 1989
Amazon-EKS-Connector-Pods stürzen fortlaufend ab 1990
eks-connector konnte nicht initiiert werden: InvalidActivation 1990
Im Cluster-Knoten fehlt die ausgehende Konnektivität 1992
Amazon-EKS-Connectors-Pods befinden sich im `ImagePullBackOff`-Status 1992
Häufig gestellte Fragen 1992
Sicherheitsüberlegungen 1993
Pflichten von AWS 1994
Pflichten des Kunden 1994
Amazon EKS in AWS Outposts 1996
Wann die einzelnen Bereitstellungsoptionen verwendet werden sollten 1996

Vergleich der Optionen für die Bereitstellung ... 1997
Lokale Cluster ausführen ... 2000
Unterstützte AWS-Regionen ... 2000
Einen lokalen Cluster bereitstellen ... 2000
EKS-Plattformversionen ... 2012
Erstellen von VPC und Subnetzen ... 2025
Auf Unterbrechungen vorbereiten ... 2028
Überlegungen zur Kapazität ... 2034
Fehlersuche bei Clustern ... 2036
Knoten ... 2049
`eksctl` ... 2051
AWS-Managementkonsole ... 2053
KI/ML in EKS ... 2060
Warum sollten Sie sich für EKS im Bereich KI/ML entscheiden? ... 2060
Wichtige Anwendungsfälle ... 2061
Fallstudien ... 2062
Einsatz von Machine Learning in EKS ... 2063
Echtzeit-Inferenz ... 2063
Cluster erstellen ... 2063
LLM-Inferenz mit vLLM ... 2102
Cluster-Konfiguration ... 2120
Verwenden Sie die EKS-Linux-GPU AMIs ... 2121
Installieren Sie das Geräte-Plugin für GPUs ... 2126
Windows-GPU-AMIs einrichten ... 2132
Trainings-Cluster mit EFA einrichten ... 2139
Inferenz-Clustern mit Inferentia einrichten ... 2150
Kapazitätsverwaltung ... 2156
GPUs für verwaltete Knotengruppen reservieren ... 2157
GPUs für selbstverwaltete Knoten reservieren ... 2160
Verwenden Sie P6e-00 mit EKS GB2 ... 2164
Rezepte ... 2175
Pods von der Planung auf bestimmten Knoten ausschließen ... 2175
Ressourcen ... 2177
Workshops ... 2177
Bewährte Methoden ... 2178
Referenzarchitekturen ... 2179

Tutorials 2180
Tools 2182
Modell Context Protocol (MCP) 2182
-Übersicht 98
Beispiele für die Integration 2184
Erste Schritte 312
Erste Schritte 2185
Konfiguration 2202
Tools 2204
Amazon Q 2215
Funktionsweise 520
Verwendung von Amazon Q zur Fehlerbehebung 2216
Überlegungen 105
Weitere Informationen 2217
Versionsverwaltung 2218
Kubernetes-Versionen 2218
Verfügbare Versionen mit Standard-Support 2219
Verfügbare Versionen mit verlängertem Support 2219
Amazon-EKS-Kubernetes-Veröffentlichungskalender 2219
Versionsinformationen mit AWS CLI abrufen 2220
Amazon EKS-Version FAQs 2222
Erweiterter Support für Amazon EKS FAQs 2224
Standard-Supportversionen 2227
Versionen mit verlängerten Support 2232
Supportzeitraum anzeigen 2235
Upgrade-Richtlinie anzeigen 2236
Erweiterten Support aktivieren 2238
Erweiterten Support deaktivieren 2240
Versionen der Plattform 2241
Kubernetes-Version `1.34` 2242
Kubernetes-Version `1.33` 2243
Kubernetes-Version `1.32` 2246
Kubernetes-Version `1.31` 2250
Kubernetes-Version `1.30` 2255
Kubernetes-Version `1.29` 2260
Aktuelle Plattformversion abrufen 2267

Plattformversion ändern ... 2267
Fehlersuche ... 2269
Unzureichende Kapazität ... 2269
Knoten können nicht mit dem Cluster verknüpft werden ... 2269
Nicht autorisiert oder Zugriff verweigert (kubectl) ... 2271
hostname doesn't match ... 2272
getsockopt: no route to host ... 2273
Instances failed to join the Kubernetes cluster ... 2273
Fehlercodes bei verwalteten Knotengruppen ... 2273
Not authorized for images ... 2278
Knoten befindet sich im NotReady-Zustand ... 2279
EKS-Protokollsammler ... 2279
Container-Laufzeitnetzwerk nicht bereit ... 2280
TLS-Handshake-Zeitüberschreitung ... 2282
InvalidClientTokenId ... 2282
Knotengruppen müssen vor dem Upgrade der Steuerebene mit der Kubernetes-Version übereinstimmen ... 2283
Beim Starten vieler Knoten gibt es Too Many Requests-Fehler ... 2283
Unautorisierte Fehlerantwort (HTTP 401) bei Kubernetes-API-Serveranfragen ... 2284
Die Amazon-EKS-Plattformversion liegt mehr als zwei Versionen hinter der aktuellen Plattformversion ... 2284
Häufig gestellte Fragen zur Cluster-Integrität und Fehlercodes mit Lösungspfaden ... 2287
Projekte im Zusammenhang mit Amazon EKS ... 2294
Support für in EKS bereitgestellte Software ... 2294
Verwaltungs-Tools ... 2295
eksctl ... 2295
AWS Controller für Kubernetes ... 2295
kro (Kube Resource Orchestrator) ... 2296
Argo CD ... 2296
Flux CD ... 2296
CDK für Kubernetes ... 2296
Netzwerk ... 2297
Amazon-VPC-CNI-Plug-In für Kubernetes ... 2297
AWS Load Balancer Controller für Kubernetes ... 2297
ExternalDNS ... 2297
Machine Learning ... 2298

Kubeflow 2298
Auto Scaling 2298
Cluster Autoscaler 2298
Karpenter 2298
Escalator 2299
Überwachung 2299
Prometheus 2299
Fortlaufende Integration/Fortlaufende Bereitstellung 2299
Jenkins X 2299
Neue Feature und Roadmap 2300
Dokumentverlauf 2301
Beitragen 2366
Einzelne Seite bearbeiten 2366
Voraussetzungen 53
Verfahren 108
Übersicht über Abrufanfragen 2368
Bearbeiten Sie Dateien mit GitHub 2369
Voraussetzungen 53
Verfahren 108
Stilempfehlungen anzeigen 2370
Vale installieren 2371
Erweiterung „VS Code Vale“ installieren 2371
Vale synchronisieren 2372
Stilempfehlungen in VS Code anzeigen 2372
Seite erstellen 2372
Seite erstellen 2373
Seite zur Navigation hinzufügen 2373
Link einfügen 2374
Link zu einer Seite oder einem Abschnitt im EKS-Benutzerhandbuch 2374
Link zu einem anderen Leitfaden in den AWS Dokumenten 2374
Link zu einer externen Webseite 2375
Mit Amazon Q erstellen 2375
Amazon Q mit VS Code installieren 2376
Bei Amazon Q anmelden 2376
Verwendung von Amazon Q zur Erstellung von Inhalten 2377
Tipps 2377

PR-Vorschau anzeigen 2377
Vorschau anzeigen 2378
Einschränkungen in der Vorschau 2379
AsciiDoc-Syntax 2379
Neue Seite 2380
Überschriften 2380
Text-Formatierung 2380
Listen 2380
Links 2381
Codebeispiele 2381
Bilder 2381
Tabellen 2382
Callouts 2382
Andere Dateien einschließen 2382
Attribute (ähnlich wie Entitäten) 2383
Verfahren 2383

Unterstützung für die Verbesserung dieser Seite beitragen

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.

Um zu diesem Benutzerhandbuch beizutragen, wählen Sie den GitHub Link Diese Seite bearbeiten auf, der sich im rechten Bereich jeder Seite befindet.

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.

Was ist Amazon EKS?

> **Tip**
>
> Melden Sie sich für bevorstehende Amazon EKS-Workshops an.

Amazon EKS: Vereinfachtes Kubernetes-Management

Amazon Elastic Kubernetes Service (EKS) bietet einen vollständig verwalteten Kubernetes-Service, der die Komplexität des Betriebs von Kubernetes-Clustern beseitigt. Mit EKS können Sie:

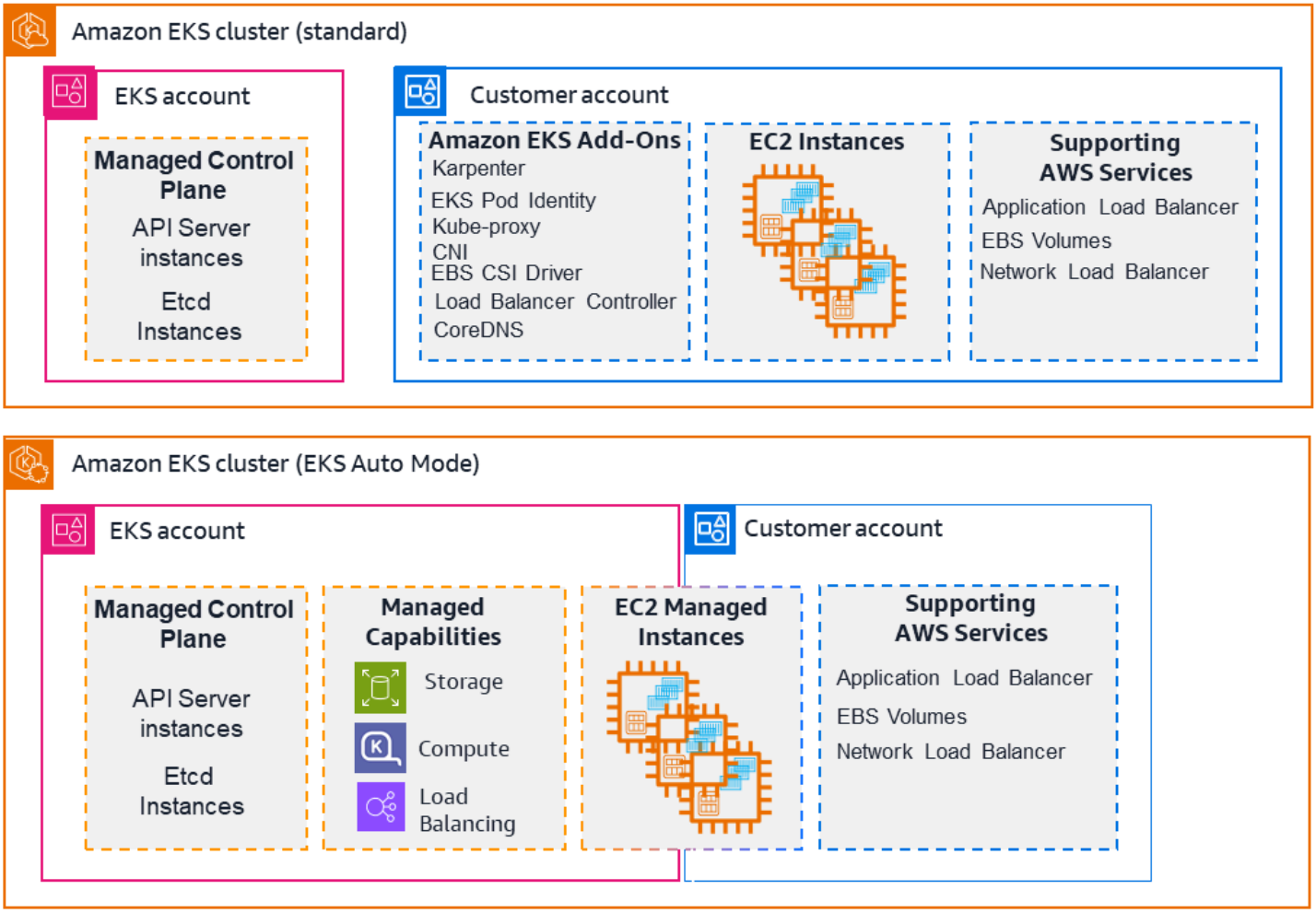
- Anwendungen schneller und mit weniger Betriebsaufwand bereitstellen.
- Nahtlos skalieren, um sich ändernden Workload-Anforderungen gerecht zu werden
- Verbessern Sie die Sicherheit durch AWS Integration und automatisierte Updates
- Zwischen dem Standard-EKS und dem vollständig automatisierten EKS Auto Mode wählen

Amazon Elastic Kubernetes Service (Amazon EKS) ist die führende Plattform für den Betrieb von Kubernetes-Clustern, sowohl in der Amazon Web Services (AWS)-Cloud als auch in Ihren eigenen Rechenzentren (EKS Anywhere und Amazon EKS Hybrid Nodes).

Amazon EKS vereinfacht die Entwicklung, Sicherung und Wartung von Kubernetes-Clustern. Es kann kostengünstiger sein, ausreichend Ressourcen für Spitzennachfragen bereitzustellen, als eigene Rechenzentren zu betreiben. Zwei der wichtigsten Ansätze für die Nutzung von Amazon EKS sind wie folgt:

- EKS-Standard: AWS verwaltet die Kubernetes-Steuerebene, wenn Sie einen Cluster mit EKS erstellen. Komponenten, die Knoten verwalten, Workloads planen, in die AWS Cloud integrieren und Informationen zur Kontrollebene speichern und skalieren, damit Ihre Cluster am Laufen bleiben, werden automatisch für Sie verwaltet.
- EKS Auto Mode: Mit dem Feature EKS Auto Mode erweitert EKS seine Steuerungsfunktionen auch auf die Verwaltung von Knoten (Kubernetes-Datenebene). Es vereinfacht das Kubernetes-Management, indem es automatisch die Infrastruktur bereitstellt, optimale Recheninstanzen auswählt, Ressourcen dynamisch skaliert, die Kosten kontinuierlich optimiert, Betriebssysteme patcht und Sicherheitsdienste integriert. AWS

Das folgende Diagramm zeigt, wie Amazon EKS Ihre Kubernetes-Cluster in die AWS Cloud integriert, je nachdem, welche Methode der Clustererstellung Sie wählen:



Amazon EKS hilft Ihnen dabei, Reibungsverluste zu vermeiden und die Produktionszeit zu verkürzen, Leistung, Verfügbarkeit und Stabilität zu verbessern und die Systemsicherheit zu verbessern. Weitere Informationen finden Sie unter Amazon Elastic Kubernetes Service.

Aufbau und Skalierung mit Kubernetes: Amazon EKS-Funktionen

Amazon EKS unterstützt Sie nicht nur beim Aufbau und der Verwaltung von Clustern, sondern auch beim Aufbau und der Skalierung von Anwendungssystemen mit Kubernetes. Amazon EKS Capabilities sind vollständig verwaltete Cluster-Services, die die Funktionalität Ihres Clusters um benutzerfreundliche Kubernetes-native Tools erweitern, darunter:

- Argo CD: Argo CD bietet eine deklarative, kontinuierliche Bereitstellung für Ihre GitOps Workloads, Ressourcen und Cloud-Infrastruktur. AWS

- AWS Controller for Kubernetes (ACK): ACK ermöglicht die native Erstellung und das Lebenszyklusmanagement von Ressourcen auf Kubernetes-Ebene und vereinheitlicht so Workload-Orchestrierung und Workflows. AWS Infrastructure-as-code
- kro (Kube Resource Orchestrator): kro erweitert die nativen Kubernetes-Funktionen, um die Erstellung, Orchestrierung und Zusammenstellung benutzerdefinierter Ressourcen zu vereinfachen, und bietet Ihnen die Tools, mit denen Sie Ihre eigenen maßgeschneiderten Cloud-Bausteine erstellen können.

EKS-Funktionen sind Cloud-Ressourcen, die den betrieblichen Aufwand für die Installation, Wartung und Skalierung dieser grundlegenden Plattformkomponenten in Ihren Clustern minimieren, sodass Sie sich auf die Entwicklung von Software statt auf den Betrieb von Clusterplattformen konzentrieren können.

Weitere Informationen hierzu finden Sie unter EKS-Fähigkeiten.

Features von Amazon EKS

Amazon EKS bietet die folgenden allgemeinen Feature:

Verwaltungsschnittstellen

EKS bietet mehrere Schnittstellen zur Bereitstellung, Verwaltung und Wartung von Clustern AWS-Managementkonsole, darunter Amazon EKS API/SDKs, CDK, AWS CLI, eksctl CLI und Terraform. AWS CloudFormation Weitere Informationen erhalten Sie unter Erste Schritte und Cluster konfigurieren.

Tools zur Zugriffskontrolle

EKS stützt sich sowohl auf Kubernetes- als auch auf AWS Identity and Access Management (AWS IAM) -Funktionen, um den Zugriff von Benutzern und Workloads zu verwalten. Weitere Informationen erhalten Sie unter the section called “Kubernetes-API-Zugriff” und the section called “Workload-Zugriff auf AWS ”.

Ressourcen berechnen

Für Rechenressourcen ermöglicht EKS die gesamte Palette der EC2 Amazon-Instance-Typen und AWS Innovationen wie Nitro und Graviton mit Amazon EKS, sodass Sie die Rechenleistung für Ihre Workloads optimieren können. Weitere Informationen finden Sie unter Verwaltung von Rechenleistung.

Speicherung

EKS Auto Mode erstellt automatisch Speicherklassen unter Verwendung von EBS-Volumes. Mithilfe von CSI-Treibern (Container Storage Interface) können Sie auch Amazon S3, Amazon EFS, Amazon FSX und Amazon File Cache für die Speicheranforderungen Ihrer Anwendungen nutzen. Weitere Informationen finden Sie unter Anwendungsdatenspeicher.

Sicherheit

In Bezug auf die Sicherheit in Amazon EKS wird das Modell der geteilten Verantwortung angewendet. Weitere Informationen finden Sie unter Bewährte Verfahren für die Sicherheit, Infrastruktursicherheit und Kubernetes-Sicherheit.

Tools zur Überwachung

Verwenden Sie für die Überwachung von Amazon-EKS-Clustern das Dashboard zur Beobachtbarkeit. Zu den Überwachungstools gehören Prometheus CloudWatch, Cloudtrail und ADOT Operator. Weitere Informationen zu Dashboards, Metriken-Servern und anderen Tools finden Sie unter EKS-Cluster-Kosten und Kubernetes Metrics Server.

Cluster-Fähigkeiten

EKS bietet verwaltete Cluster-Funktionen für die kontinuierliche Bereitstellung, Cloud-Ressourcenverwaltung und Ressourcenzusammensetzung auf der Grundlage von Open-Source-Innovationen. EKS installiert Kubernetes APIs in Ihren Clustern, aber Controller und andere Komponenten laufen in EKS und werden vollständig verwaltet, sodass automatisiertes Patchen, Skalieren und Überwachen möglich ist. Weitere Informationen finden Sie unter EKS-Fähigkeiten.

Kubernetes-Kompatibilität und -Unterstützung

Amazon EKS ist als Kubernetes-konform zertifiziert, sodass Sie Kubernetes-kompatible Anwendungen ohne Refactoring bereitstellen und die Tools und Plugins der Kubernetes-Community nutzen können. EKS bietet sowohl Standard-Support als auch erweiterten Support für Kubernetes an. Weitere Informationen finden Sie unter the section called “Kubernetes-Versionen”.

Zugehörige Services

Services zur Verwendung mit Amazon EKS

Sie können andere AWS Services mit den Clustern verwenden, die Sie mit Amazon EKS bereitstellen:

Amazon EC2

Besorgen Sie sich skalierbare Rechenkapazität auf Abruf mit Amazon EC2.

Amazon EBS

Fügen Sie mit Amazon EBS skalierbare, leistungsstarke Blockspeicherressourcen hinzu.

Amazon ECR

Speichern Sie Container-Images sicher mit Amazon ECR.

Amazon CloudWatch

Überwachen Sie AWS Ressourcen und Anwendungen in Echtzeit mit Amazon CloudWatch.

Amazon Prometheus

Verfolgen Sie Metriken für containerisierte Anwendungen mit Amazon Managed Service für Prometheus.

Elastic Load Balancing

Verteilen Sie eingehenden Datenverkehr mit Elastic Load Balancing auf mehrere Ziele.

Amazon GuardDuty

Erkennen Sie mit Amazon Bedrohungen für EKS-Cluster GuardDuty.

AWS Resilience Hub

Bewerten Sie die Ausfallsicherheit des EKS-Clusters mit AWS Resilience Hub.

Preisgestaltung von Amazon EKS

Amazon EKS bietet Preise pro Cluster basierend auf dem Support für die Kubernetes-Cluster-Version, Preise für Amazon EKS Auto Mode und Preise pro vCPU für Amazon EKS Hybrid Nodes.

Wenn Sie Amazon EKS verwenden, zahlen Sie separat für die AWS Ressourcen, die Sie für die Ausführung Ihrer Anwendungen auf Kubernetes-Worker-Knoten verwenden. Wenn Sie beispielsweise Kubernetes-Worker-Knoten als EC2 Amazon-Instances mit Amazon EBS-Volumes und öffentlichen IPv4 Adressen ausführen, werden Ihnen die Instance-Kapazität über Amazon EC2, die Volume-Kapazität über Amazon EBS und die IPv4 Adresse über Amazon VPC in Rechnung gestellt.

Detaillierte Preisinformationen finden Sie auf den jeweiligen Preisseiten der AWS Services, die Sie mit Ihren Kubernetes-Anwendungen nutzen.

- Die Preise für Amazon EKS-Cluster, Amazon EKS Auto Mode, Amazon EKS-Funktionen und Amazon EKS-Hybridknoten finden Sie unter Amazon EKS-Preise.
- EC2 Die Amazon-Preise finden Sie unter Amazon EC2 On-Demand-Preise und Amazon EC2 Spot-Preise.
- Informationen zu AWS Fargate-Preisen finden Sie unter AWS Fargate-Preise.
- Sie können Ihre Savings Plans für die in Amazon-EKS-Clustern genutzte Rechenleistung verwenden. Weitere Informationen finden Sie unter Preise und Savings Plans.

Häufige Anwendungsfälle in Amazon EKS

Amazon EKS bietet robuste verwaltete Kubernetes-Services, die darauf ausgelegt sind AWS, containerisierte Anwendungen zu optimieren. Im Folgenden finden Sie einige der häufigsten Anwendungsfälle von Amazon EKS, die Ihnen dabei helfen, die Stärken von Amazon EKS für Ihre spezifischen Anforderungen zu nutzen.

Bereitstellung von Anwendungen mit hoher Verfügbarkeit

: Mit Elastic Load Balancing können Sie sicherstellen, dass Ihre Anwendungen über mehrere Availability Zones hinweg hochverfügbar sind.

Entwicklung von Microservices-Architekturen

: Verwenden Sie die Serviceerkennungs-Feature von Kubernetes mit AWS Cloud Map oder Amazon VPC Lattice, um ausfallsichere Systeme zu erstellen.

Automatisierung von Softwareveröffentlichungsprozessen

: Verwalten Sie CI/CD-Pipelines (Continuous Integration and Continuous Deployment), die den Prozess der automatisierten Erstellung, Prüfung und Bereitstellung von Anwendungen vereinfachen. Informationen zur deklarativen kontinuierlichen Bereitstellung finden Sie unter Kontinuierliche Bereitstellung mit Argo CD.

Serverlose Anwendungen ausführen

: Verwenden Sie AWS Fargate mit Amazon EKS, um Serverless-Anwendungen auszuführen. Das bedeutet, dass Sie sich ausschließlich auf die Anwendungsentwicklung konzentrieren können, während Amazon EKS und Fargate die zugrunde liegende Infrastruktur verwalten.

Ausführung von Workloads für maschinelles Lernen

Amazon EKS ist mit gängigen Machine-Learning-Frameworks wie TensorFlow, MXNet und PyTorch kompatibel. Mit GPU-Unterstützung können Sie selbst komplexe Machine-Learning-Aufgaben effektiv bewältigen.

Konsistente Bereitstellung vor Ort und in der Cloud

Um die Ausführung von Kubernetes in On-Premises-Umgebungen zu vereinfachen, können Sie dieselben Amazon-EKS-Cluster, -Features und -Tools verwenden, um selbstverwaltete Knoten in AWS Outposts auszuführen, oder Sie können Amazon EKS Hybrid Nodes mit Ihrer eigenen Infrastruktur nutzen. Für in sich geschlossene, Air-Gap-Umgebungen können Sie Amazon EKS Anywhere verwenden, um die Lebenszyklusverwaltung von Kubernetes-Clustern auf Ihrer eigenen Infrastruktur zu automatisieren.

Ausführung kostengünstiger Batch-Verarbeitungs- und Big-Data-Workloads

Nutzen Sie Spot Instances, um Ihre Batch-Verarbeitungs- und Big-Data-Workloads wie Apache Hadoop und Spark zu einem Bruchteil der Kosten auszuführen. Auf diese Weise können Sie ungenutzte EC2 Amazon-Kapazitäten zu ermäßigten Preisen nutzen.

AWS Ressourcen von Kubernetes aus verwalten

Verwenden Sie AWS Controllers for Kubernetes (ACK), um AWS Ressourcen mit nativem Kubernetes direkt von Ihrem Kubernetes-Cluster aus zu erstellen und zu verwalten. APIs

Abstraktionen für die Plattformtechnik erstellen

Erstellen Sie mithilfe von kro (Kube APIs Resource Orchestrator) benutzerdefinierte Kubernetes, die mehrere Ressourcen zu Abstraktionen auf höherer Ebene zusammensetzen.

Sicherung von Anwendungen und Sicherstellung der Einhaltung von Vorschriften

Implementieren Sie strenge Sicherheitspraktiken und sorgen Sie für die Einhaltung der Vorschriften mit Amazon EKS, das in AWS Sicherheitsservices wie AWS Identity and Access Management (IAM), Amazon Virtual Private Cloud (Amazon VPC) und AWS Key Management Service (AWS KMS) integriert ist. Dadurch werden der Datenschutz und die Datensicherheit gemäß den Branchenstandards gewährleistet.

Amazon-EKS-Architektur

Amazon EKS entspricht der allgemeinen Cluster-Architektur von Kubernetes. Weitere Informationen finden Sie unter Kubernetes-Komponente in der Kubernetes-Dokumentation. In den folgenden Abschnitten werden einige zusätzliche Architekturdetails für Amazon EKS zusammengefasst.

Steuerebene

Amazon EKS sorgt dafür sicher, dass jeder Cluster über eine eigene, eindeutige Kubernetes-Steuerebene verfügt. Bei diesem Design bleibt die Infrastruktur jedes Clusters getrennt, sodass es keine Überschneidungen zwischen Clustern oder Konten gibt. AWS Das Setup beinhaltet:

Verteilte Komponenten

: Die Kontrollebene positioniert mindestens zwei API-Serverinstanzen und drei etcd-Instanzen in drei AWS Availability Zones innerhalb einer AWS Region.

Optimale Leistung

: Amazon EKS überwacht und passt die Instances der Steuerebene aktiv an, um eine Spitzenleistung aufrechtzuerhalten.

Ausfallsicherheit

: Wenn eine Kontrollebeneninstanz ins Stocken gerät, ersetzt Amazon EKS sie schnell und verwendet bei Bedarf eine andere Availability Zone.

Konsistente Verfügbarkeit

: Durch den Betrieb von Clustern in mehreren Availability Zones wird ein zuverlässiges Service Level Agreement (SLA) für die Verfügbarkeit von API-Server-Endpunkten erzielt.

Amazon EKS verwendet Amazon Virtual Private Cloud (Amazon VPC), um den Datenverkehr zwischen Komponenten der Steuerebene innerhalb eines einzelnen Clusters zu begrenzen. Clusterkomponenten können keine Kommunikation von anderen Clustern oder AWS Konten anzeigen oder empfangen, es sei denn, dies ist durch die Richtlinien für die rollenbasierte Zugriffskontrolle (RBAC) von Kubernetes autorisiert.

Datenverarbeitung

Zusätzlich zur Steuerebene verfügt ein Amazon-EKS-Cluster über eine Reihe von Worker-Computern, die als Knoten bezeichnet werden. Die Auswahl des geeigneten Amazon-EKS-

Cluster-Knotentyps ist entscheidend, um Ihre spezifischen Anforderungen zu erfüllen und die Ressourcennutzung zu optimieren. Amazon EKS bietet die folgenden Primärknotentypen:

EKS Auto Mode

Der EKS Auto Mode erweitert die AWS Verwaltung über die Steuerungsebene hinaus auf die Datenebene und automatisiert so die Verwaltung der Cluster-Infrastruktur. Er integriert zentrale Kubernetes-Funktionen als integrierte Komponenten, darunter automatische Skalierung der Rechenleistung, Netzwerke, Lastenausgleich, DNS, Speicher und GPU-Unterstützung. Der automatische Modus von EKS verwaltet Knoten dynamisch auf der Grundlage der Arbeitslastanforderungen und verwendet unveränderliche Funktionen AMIs mit erweiterten Sicherheitsfunktionen. Er automatisiert Aktualisierungen und Upgrades unter Berücksichtigung der Budgets für Pod-Unterbrechung und umfasst verwaltete Komponenten, die andernfalls eine zusätzliche Verwaltung erfordern würden. Diese Option ist ideal für Benutzer, die AWS Fachwissen für den day-to-day Betrieb nutzen, den Betriebsaufwand minimieren und sich auf die Anwendungsentwicklung statt auf die Infrastrukturverwaltung konzentrieren möchten.

AWS Fargate

Fargate ist eine Serverless-Compute-Engine für Container, die die Verwaltung der zugrunde liegenden Instances überflüssig macht. Mit Fargate spezifizieren Sie den Ressourcenbedarf Ihrer Anwendung und stellen die Infrastruktur AWS automatisch bereit, skalieren und warten. Diese Option ist ideal für Benutzer, die Prioritäten setzen ease-of-use und sich auf die Anwendungsentwicklung und -bereitstellung konzentrieren möchten, anstatt die Infrastruktur zu verwalten.

Karpenter

Karpenter ist ein flexibler, leistungsstarker Kubernetes-Cluster-Autoscaler, der zur Verbesserung der Anwendungsverfügbarkeit und Cluster-Effizienz beiträgt. Karpenter stellt als Reaktion auf die sich ändernde Anwendungslast Rechenressourcen in der richtigen Größe bereit. Mit dieser Option können just-in-time Rechenressourcen bereitgestellt werden, die den Anforderungen Ihrer Arbeitslast entsprechen.

Verwaltete Knotengruppen

Verwaltete Knotengruppen sind eine Mischung aus Automatisierung und Anpassung für die Verwaltung einer Sammlung von EC2 Amazon-Instances innerhalb eines Amazon EKS-Clusters. AWS kümmert sich um Aufgaben wie das Patchen, Aktualisieren und Skalieren von Knoten und vereinfacht so betriebliche Aspekte. Gleichzeitig werden benutzerdefinierte `kubelet`-Argumente unterstützt, die Möglichkeiten für erweiterte CPU- und Speicherverwaltungsrichtlinien eröffnen.

Darüber hinaus verbessern sie die Sicherheit durch AWS Identity and Access Management (IAM) -Rollen für Dienstkonten und reduzieren gleichzeitig den Bedarf an separaten Berechtigungen pro Cluster.

Selbstverwaltete Knoten

Selbstverwaltete Knoten bieten die volle Kontrolle über Ihre EC2 Amazon-Instances innerhalb eines Amazon EKS-Clusters. Sie sind für die Verwaltung, Skalierung und Wartung der Knoten zuständig und haben somit die volle Kontrolle über die zugrunde liegende Infrastruktur. Diese Option eignet sich für Benutzer, die eine differenzierte Kontrolle und Anpassung ihrer Knoten benötigen und bereit sind, Zeit in die Verwaltung und Wartung ihrer Infrastruktur zu investieren.

Amazon EKS Hybrid Nodes

Mit Amazon EKS Hybrid Nodes können Sie Ihre On-Premises- und Edge-Infrastruktur als Knoten in Amazon-EKS-Clustern verwenden. Amazon EKS Hybrid Nodes vereinheitlicht das Kubernetes-Management in allen Umgebungen und verlagert die Verwaltung der Kubernetes-Steuerungsebene auf AWS Ihre lokalen und Edge-Anwendungen.

EKS-Funktionen

Amazon EKS bietet vollständig verwaltete Cluster-Funktionen und installiert und verwaltet Kubernetes APIs (mit benutzerdefinierten Kubernetes-Ressourcendefinitionen) in Ihrem Cluster, während Controller und andere Komponenten in einer AWS eigenen Infrastruktur betrieben werden, die von Ihrem Cluster getrennt ist. EKS bietet automatisiertes Patchen, Skalieren und Überwachen dieser Funktionen und verwaltet deren gesamten Lebenszyklus, um den Aufwand für den Betrieb von Cluster-Services für Workload-Orchestrierung, Ressourcenmanagement und mehr zu reduzieren. AWS

EKS bietet die folgenden Funktionstypen:

AWS Controller für Kubernetes (ACK)

AWS Mit Controllers for Kubernetes (ACK) können Sie AWS Ressourcen mithilfe von Kubernetes verwalten und S3-Buckets APIs, RDS-Datenbanken, IAM-Rollen und andere Ressourcen als benutzerdefinierte Kubernetes-Ressourcen definieren. AWS Sie können AWS Ressourcen zusammen mit Ihren Kubernetes-Workloads mithilfe derselben Tools und Workflows verwalten, wobei mehr als 50 AWS Dienste unterstützt werden, darunter S3, RDS, DynamoDB und Lambda.

Argo CD

Argo CD implementiert ein GitOps basiertes kontinuierliches Deployment für Ihre Anwendungs-Workloads, AWS Ressourcen und Cluster-Konfigurationen, wobei Git-Repositorys als Quelle der Wahrheit verwendet werden. Argo CD synchronisiert Ihre Cluster automatisch mit Ihren Git-Repositorys, erkennt Abweichungen und gleicht sie kontinuierlich ab, um sicherzustellen, dass Ihre bereitgestellten Anwendungen und Ressourcen Ihrem gewünschten Status in der Versionskontrolle entsprechen. Sie können Argo CD verwenden, um Anwendungen auf einem bestimmten Cluster zu verwalten, oder Anwendungen über mehrere Cluster von einer einzigen Argo-CD-Ressource aus bereitstellen und verwalten, wobei die automatische Bereitstellung aus Git-Repositorys erfolgt, wann immer Änderungen festgeschrieben werden.

kro (Kube Resource Orchestrator)

kro (Kube Resource Orchestrator) ermöglicht es Ihnen, benutzerdefinierte Kubernetes zu erstellen, die mehrere Ressourcen zu Abstraktionen auf höherer Ebene zusammensetzen APIs , sodass Plattformteams wiederverwendbare Muster für gängige Ressourcenkombinationen definieren können. Auf diese Weise können Plattformteams Self-Service-Funktionen mit entsprechenden Leitplanken bereitstellen, sodass Entwickler komplexe Infrastrukturen mithilfe einfacher, speziell entwickelter Systeme bereitstellen und gleichzeitig die organisatorischen Standards und Best Practices einhalten können. APIs

Kubernetes-Konzepte

Amazon Elastic Kubernetes Service (Amazon EKS) ist ein AWS-verwalteter Service, der auf dem Open-Source-Projekt Kubernetes basiert. Es gibt Dinge, die Sie über die Integration des Amazon-EKS-Services in die AWS Cloud wissen müssen (insbesondere, wenn Sie zum ersten Mal einen Amazon-EKS-Cluster erstellen), aber sobald er betriebsbereit ist, verwenden Sie Ihren Amazon-EKS-Cluster auf die gleiche Weise wie jeden anderen Kubernetes-Cluster. Um mit der Verwaltung von Kubernetes-Clustern und dem Bereitstellen von Workloads zu beginnen, benötigen Sie daher zumindest ein grundlegendes Verständnis der Kubernetes-Konzepte.

Auf dieser Seite werden die Kubernetes-Konzepte in drei Abschnitte unterteilt: the section called “Warum Kubernetes?”, the section called “Cluster” und the section called “Workloads”. Der erste Abschnitt beschreibt den Wert der Ausführung eines Kubernetes-Services, insbesondere als verwalteter Service wie Amazon EKS. Der Abschnitt „Workloads“ behandelt, wie Kubernetes-Anwendungen erstellt, gespeichert, ausgeführt und verwaltet werden. Im Abschnitt „Cluster“ werden

die verschiedenen Komponenten erläutert, aus denen Kubernetes-Cluster bestehen, und welche Aufgaben Sie beim Erstellen und Verwalten von Kubernetes-Clustern haben.

Themen

- Warum Kubernetes?
- Cluster
- Workloads
- Nächste Schritte

Beim Durchgehen dieser Inhalte führen Sie Links zu weiteren Beschreibungen der Kubernetes-Konzepte in der Amazon-EKS- und Kubernetes-Dokumentation, falls Sie tiefer in die hier behandelten Themen eintauchen möchten. Weitere Informationen zur Implementierung der Kubernetes-Steuerebene und der Rechenfunktionen in Amazon EKS finden Sie unter the section called “Architektur”.

Warum Kubernetes?

Kubernetes wurde entwickelt, um die Verfügbarkeit und Skalierbarkeit beim Ausführen unternehmenskritischer containerisierter Anwendungen in Produktionsqualität zu verbessern. Anstatt Kubernetes nur auf einem einzelnen Rechner auszuführen (obwohl dies möglich ist), erreicht Kubernetes diese Ziele, indem es Ihnen ermöglicht, Anwendungen auf einer Reihe von Rechnern auszuführen, die je nach Bedarf erweitert oder reduziert werden können. Kubernetes enthält Funktionen, die Ihnen Folgendes vereinfachen:

- Anwendungen auf mehreren Rechnern (mithilfe von in Pods bereitgestellten Containern) bereitstellen
- Container-Zustand und Neustart ausgefallener Container überwachen
- Container je nach Auslastung nach oben und unten zu skalieren
- Container mit neuen Versionen aktualisieren
- Ressourcen zwischen Containern zuzuweisen
- Datenverkehr über mehrere Rechner hinweg ausgleichen

Durch die Automatisierung dieser komplexen Aufgaben durch Kubernetes kann sich ein Anwendungsentwickler auf die Erstellung und Verbesserung seiner Anwendungs-Workloads konzentrieren, anstatt sich um die Infrastruktur zu kümmern. Der Entwickler erstellt in der Regel

Konfigurationsdateien im YAML-Format, die den gewünschten Zustand der Anwendung beschreiben. Dies kann beispielsweise die auszuführenden Container, Ressourcenlimits, die Anzahl der Pod-Replikate, die CPU-/Speicherzuweisung, Affinitätsregeln und mehr beinhalten.

Eigenschaften von Kubernetes

Um seine Ziele zu erreichen, verfügt Kubernetes über die folgenden Eigenschaften:

- Containerisiert – Kubernetes ist ein Container-Orchestrierungstool. Um Kubernetes verwenden zu können, müssen Sie zunächst Ihre Anwendungen containerisieren. Je nach Art der Anwendung kann dies als eine Reihe von Microservices, als Batch-Aufträge oder in anderen Formen erfolgen. Anschließend können Ihre Anwendungen die Vorteile eines Kubernetes-Workflows nutzen, der ein umfangreiches Ökosystem von Tools beinhaltet, in dem Container als Images in einer Container-Registry gespeichert, in einem Kubernetes-Cluster bereitgestellt und in einem verfügbaren Knoten ausgeführt werden können. Sie können einzelne Container auf Ihrem lokalen Computer mit Docker oder einer anderen Container-Laufzeitumgebung erstellen und testen, bevor Sie sie in Ihrem Kubernetes-Cluster bereitstellen.
- Skalierbar – Wenn die Nachfrage nach Ihren Anwendungen die Kapazität der auszuführenden Instances dieser Anwendungen übersteigt, kann Kubernetes entsprechend hochskaliert werden. Bei Bedarf kann Kubernetes erkennen, ob Anwendungen mehr CPU oder Speicher benötigen, und darauf reagieren, indem es entweder automatisch die verfügbare Kapazität erweitert oder mehr der vorhandenen Kapazität nutzt. Die Skalierung kann auf Pod-Ebene erfolgen, wenn genügend Rechenleistung für die Ausführung mehrerer Anwendungs-Instances zur Verfügung steht (horizontale Pod-Autoskalierung), oder auf Knotenebene, wenn mehr Knoten für die erhöhte Kapazität hochgefahren werden müssen (Cluster Autoscaler oder Karpenter). Da keine Kapazität mehr benötigt wird, können diese Services unnötige Pods löschen und nicht mehr benötigte Knoten herunterfahren.
- Verfügbar – Wenn eine Anwendung oder ein Knoten fehlerhaft oder nicht mehr verfügbar ist, kann Kubernetes die ausgeführten Workloads auf einen anderen verfügbaren Knoten verschieben. Sie können das Problem beheben, indem Sie einfach eine ausgeführte Instance einer Workload oder einen Knoten löschen, auf dem Ihre Workload ausgeführt wird. Im Endeffekt können Workloads an anderen Standorten hochgefahren werden, wenn sie dort nicht mehr ausgeführt werden können.
- Deklarativ – Kubernetes verwendet aktive Abstimmung, um ständig zu überprüfen, ob der von Ihnen für Ihren Cluster deklarierte Status mit dem tatsächlichen Status übereinstimmt. Durch die Anwendung von Kubernetes-Objekten auf einen Cluster, in der Regel über Konfigurationsdateien im YAML-Format, können Sie beispielsweise den Start der Workloads anfordern, die Sie in Ihrem Cluster ausführen möchten. Sie können die Konfigurationen später ändern, um beispielsweise

eine neuere Version eines Containers zu verwenden oder mehr Speicher zuzuweisen. Kubernetes wird die erforderlichen Maßnahmen ergreifen, um den gewünschten Zustand herzustellen. Dazu gehört unter anderem das Hoch- oder Herunterfahren von Knoten, das Beenden und Neustarten von Workloads oder das Abrufen aktualisierter Container.

- Zusammensetzbar – Da eine Anwendung in der Regel aus mehreren Komponenten besteht, ist es wünschenswert, eine Reihe dieser Komponenten (oft durch mehrere Container repräsentiert) gemeinsam verwalten zu können. Während Docker Compose eine Möglichkeit bietet, dies direkt mit Docker zu tun, kann Ihnen der Befehl Kubernetes Kompose dabei helfen, dies mit Kubernetes zu erreichen. Ein Beispiel hierfür finden Sie unter Übersetzen einer Docker-Compose-Datei in Kubernetes-Ressourcen.
- Erweiterbar – Im Gegensatz zu proprietärer Software ist das Open-Source-Projekt Kubernetes so konzipiert, dass Sie Kubernetes beliebig erweitern können, um Ihren Anforderungen gerecht zu werden. APIs und Konfigurationsdateien können direkt geändert werden. Drittanbieter werden aufgefordert, eigene Controller zu schreiben, um sowohl die Infrastruktur- als auch die Endbenutzer-Feature von Kubernetes zu erweitern. Mit Webhooks können Sie Cluster-Regeln einrichten, um Richtlinien durchzusetzen und sich an veränderte Bedingungen anzupassen. Weitere Informationen zur Erweiterung von Kubernetes-Clustern finden Sie unter Erweiterung von Kubernetes.
- Portabel – Viele Unternehmen haben ihre Abläufe in Kubernetes standardisiert, da sie damit alle ihre Anwendungsanforderungen auf einheitliche Weise verwalten können. Entwickler können dieselben Pipelines zum Entwickeln und Speichern von containerisierten Anwendungen verwenden. Diese Anwendungen können dann in Kubernetes-Clustern bereitgestellt werden, die On-Premises, in Clouds, auf POS-Terminals in Restaurants oder auf IoT-Geräten an Fern-Standorten des Unternehmens ausgeführt werden. Da es sich um eine Open-Source-Software handelt, können Nutzer diese speziellen Kubernetes-Distributionen sowie die für deren Verwaltung erforderlichen Tools entwickeln.

Verwaltung von Kubernetes

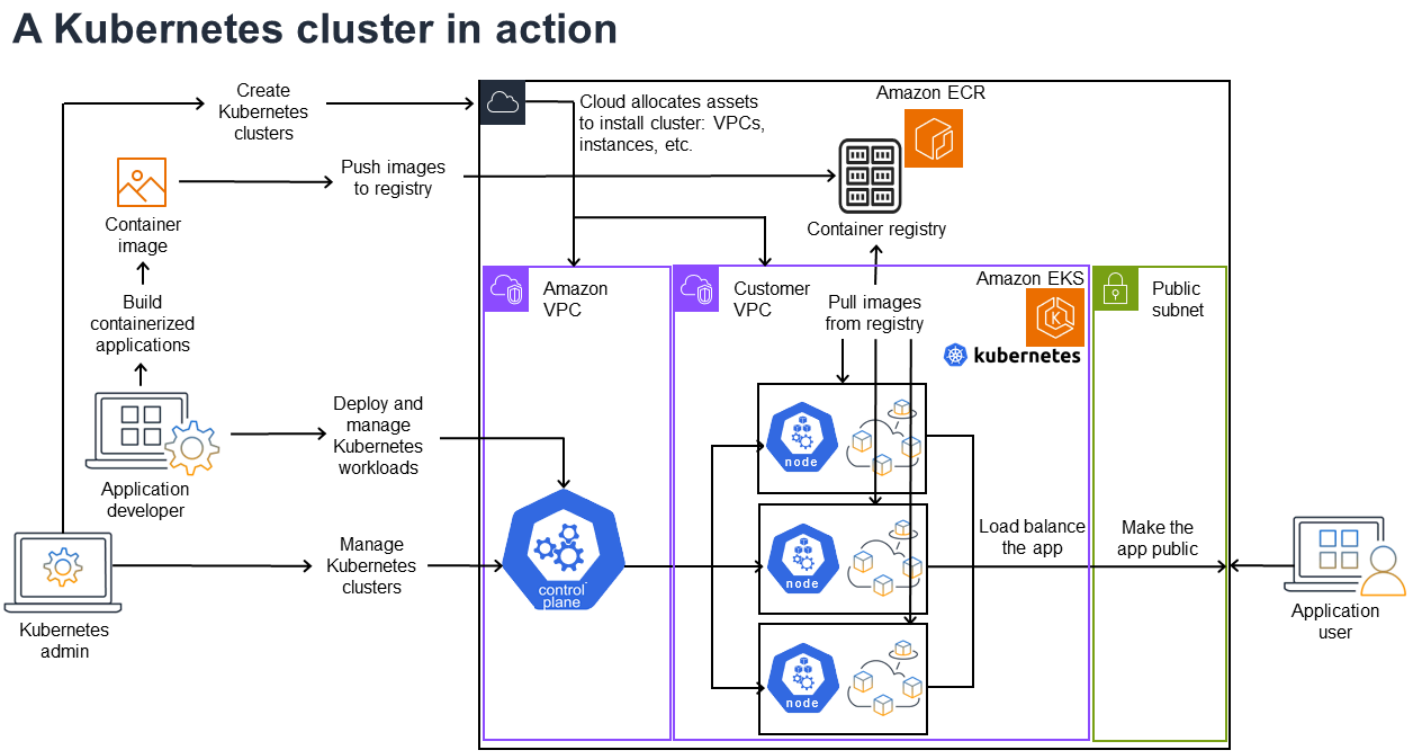
Der Quellcode von Kubernetes ist kostenlos verfügbar, sodass Sie Kubernetes mit Ihrer eigenen Ausrüstung selbst installieren und verwalten können. Die eigenständige Verwaltung von Kubernetes erfordert jedoch fundierte Fachkenntnisse und ist mit einem erheblichen Zeit- und Arbeitsaufwand verbunden. Aus diesen Gründen entscheiden sich die meisten Anwender, die Produktions-Workloads bereitstellen, für einen Cloud-Anbieter (z. B. Amazon EKS) oder einen On-Premises-Anbieter (z. B. Amazon EKS Anywhere) mit einer eigenen erprobten Kubernetes-Verteilung und Support durch

Kubernetes-Experten. Dadurch können Sie einen Großteil der undifferenzierten, aufwändigen Aufgaben, die für die Wartung Ihrer Cluster erforderlich sind, auslagern, darunter:

- Hardware – Wenn Sie nicht über die erforderliche Hardware verfügen, um Kubernetes gemäß Ihren Anforderungen auszuführen, können Sie mit einem Cloud-Anbieter wie AWS Amazon EKS Vorabkosten einsparen. Mit Amazon EKS können Sie die besten Cloud-Ressourcen von AWS nutzen. Dazu gehören Computer-Instances (Amazon Elastic Compute Cloud), Ihre eigene private Umgebung (Amazon VPC), zentrale Identitäts- und Berechtigungsverwaltung (IAM) und Speicher (Amazon EBS). AWS verwaltet die Computer, Netzwerke, Rechenzentren und alle anderen physischen Komponenten, die für den Betrieb von Kubernetes erforderlich sind. Ebenso müssen Sie Ihr Rechenzentrum nicht für die maximale Kapazität an Tagen mit höchster Auslastung auslegen. Bei Amazon EKS Anywhere oder anderen On-Premises-Kubernetes-Clustern sind Sie für die Verwaltung der in Ihren Kubernetes-Bereitstellungen verwendeten Infrastruktur verantwortlich, können sich jedoch weiterhin auf AWS verlassen, um Kubernetes auf dem neuesten Stand zu halten.
- Verwaltung der Steuerebene – Amazon EKS verwaltet die Sicherheit und Verfügbarkeit der von AWS gehosteten Kubernetes-Steuerebene, die für die Planung von Containern, die Verwaltung der Anwendungsverfügbarkeit und andere wichtige Aufgaben verantwortlich ist, sodass Sie sich auf Ihre Anwendungs-Workloads konzentrieren können. Wenn Ihr Cluster ausfällt, sollte AWS über die erforderlichen Mittel verfügen, um Ihren Cluster wieder in einen auszuführenden Status zu versetzen. Bei Amazon EKS Anywhere verwalten Sie die Steuerebene selbst.
- Geprüfte Upgrades – Bei der Aktualisierung Ihrer Cluster können Sie sich darauf basieren, dass Amazon EKS oder Amazon EKS Anywhere geprüfte Versionen ihrer Kubernetes-Verteilungen bereitstellen.
- Add-Ons – Es gibt Hunderte von Projekten, die für die Erweiterung und Arbeit mit Kubernetes entwickelt wurden und die Sie der Infrastruktur Ihres Clusters hinzufügen oder zur Unterstützung der Ausführung Ihrer Workloads verwenden können. Anstatt diese Add-Ons selbst zu entwickeln und zu verwalten, stellt AWS the section called “Amazon-EKS-Add-ons” bereit, die Sie mit Ihren Clustern verwenden können. Amazon EKS Anywhere bietet kuratierte Pakete, die Entwicklungen vieler beliebter Open-Source-Projekte enthalten. Daher ist es nicht erforderlich, die Software selbst zu entwickeln oder wichtige Sicherheitspatches, Fehlerbehebungen oder Upgrades zu verwalten. Wenn die Standardeinstellungen Ihren Anforderungen entsprechen, ist in der Regel nur ein sehr geringer Konfigurationsaufwand für diese Add-Ons erforderlich. Weitere Informationen zur Erweiterung Ihres Clusters mit Add-Ons finden Sie unter the section called “Cluster erweitern”.

Kubernetes in Aktion

Das folgende Diagramm zeigt die wichtigsten Aktivitäten, die Sie als Kubernetes-Administrator oder Anwendungsentwickler zum Erstellen und zur Nutzung eines Kubernetes-Clusters durchführen würden. Dabei wird veranschaulicht, wie Kubernetes-Komponenten miteinander interagieren. Die AWS Cloud dient dabei als Beispiel für den zugrunde liegenden Cloud-Anbieter.



Ein Kubernetes-Administrator erstellt den Kubernetes-Cluster mit einem Tool, das speziell auf den Anbietertyp zugeschnitten ist, auf dem der Cluster entwickelt wird. In diesem Beispiel wird die AWS Cloud als Anbieter verwendet, der den verwalteten Kubernetes-Service Amazon EKS bereitstellt. Der verwaltete Service weist automatisch die zum Erstellen des Clusters erforderlichen Ressourcen zu. Dazu gehört das Erstellen von zwei neuen Virtual Private Clouds (Amazon VPCs) für den Cluster, die Einrichtung des Netzwerks und die Zuordnung von Kubernetes-Berechtigungen zu den neuen VPCs für die Verwaltung von Cloud-Ressourcen. Der verwaltete Service stellt außerdem sicher, dass die Services der Steuerebene über ausreichend Platz zum Ausführen verfügen, und weist null oder mehr Amazon-EC2-Instances als Kubernetes-Knoten zum Ausführen von Workloads zu. AWS verwaltet selbst eine Amazon VPC für die Steuerebene, während die andere Amazon VPC die Kundenknoten enthält, die Workloads ausführen.

Viele Aufgaben des Kubernetes-Administrators werden in Zukunft mithilfe von Kubernetes-Tools wie `kubectl` ausgeführt. Dieses Tool stellt Service-Anfragen direkt an die Steuerebene des Clusters. Die Vorgehensweise bei Abfragen und Änderungen am Cluster entspricht im Wesentlichen der Vorgehensweise, die Sie bei jedem Kubernetes-Cluster anwenden würden.

Ein Anwendungsentwickler, der Workloads in diesem Cluster bereitstellen möchte, kann mehrere Aufgaben ausführen. Der Entwickler muss die Anwendung in ein oder mehrere Container-Images integrieren und diese Images dann in ein Container-Registry übertragen, auf das der Kubernetes-Cluster zugreifen kann. AWS bietet hierfür das Amazon Elastic Container Registry (Amazon ECR) an.

Um die Anwendung auszuführen, kann der Entwickler Konfigurationsdateien im YAML-Format erstellen, die dem Cluster mitteilen, wie die Anwendung ausgeführt werden soll, einschließlich der Angabe, welche Container aus der Registry abgerufen und wie diese Container in Pods verpackt werden sollen. Die Steuerebene (Scheduler) plant die Container für einen oder mehrere Knoten ein und die Container-Laufzeit auf jedem Knoten ruft die benötigten Container ab und führt sie aus. Der Entwickler kann auch einen Application Load Balancer einrichten, um den Datenverkehr auf die verfügbaren Container zu verteilen, die im jeweiligen Knoten ausgeführt werden, und die Anwendung so freizugeben, dass sie in einem öffentlichen Netzwerk für die Außenwelt verfügbar ist. Anschließend kann sich jeder, der die Anwendung nutzen möchte, mit dem Anwendungsendpunkt verbinden und darauf zugreifen.

In den folgenden Abschnitten werden die einzelnen Features aus der Perspektive von Kubernetes-Clustern und -Workloads detailliert beschrieben.

Cluster

Wenn Sie Kubernetes-Cluster starten und verwalten, sollten Sie wissen, wie Kubernetes-Cluster erstellt, erweitert, verwaltet und gelöscht werden. Sie sollten außerdem wissen, aus welchen Komponenten ein Cluster besteht und wie Sie diese Komponenten warten.

Tools zur Cluster-Verwaltung verwalten die Überschneidungen zwischen den Kubernetes-Services und dem zugrunde liegenden Hardware-Anbieter. Aus diesem Grund wird die Automatisierung dieser Aufgaben in der Regel vom Kubernetes-Anbieter (z. B. Amazon EKS oder Amazon EKS Anywhere) mithilfe von anbieterspezifischen Tools durchgeführt. Um beispielsweise einen Amazon-EKS-Cluster zu starten, können Sie `eksctl create cluster` verwenden, während Sie für Amazon EKS Anywhere `eksctl anywhere create cluster` verwenden können. Beachten Sie, dass diese Befehle zwar einen Kubernetes-Cluster erstellen, jedoch spezifisch für den Anbieter sind und nicht Teil des Kubernetes-Projekts selbst sind.

Tools zur Erstellung und Verwaltung von Clustern

Das Kubernetes-Projekt bietet Tools zur manuellen Erstellung eines Kubernetes-Clusters. Wenn Sie Kubernetes auf einem einzelnen Rechner installieren oder die Steuerebene auf einem Rechner ausführen und Knoten manuell hinzufügen möchten, können Sie CLI-Tools wie kind, minikube oder kubeadm verwenden, die unter Kubernetes-Installations-Tools aufgeführt sind. Um den gesamten Lebenszyklus der Cluster-Erstellung und -Verwaltung zu vereinfachen und zu automatisieren, ist es viel einfacher, Tools zu verwenden, die von einem etablierten Kubernetes-Anbieter unterstützt werden, wie z. B. Amazon EKS oder Amazon EKS Anywhere.

In der AWS Cloud können Sie Cluster von Amazon EKS mithilfe von CLI-Tools wie eksctl oder deklarativeren Tools wie Terraform erstellen (siehe Amazon-EKS-Vorlagen für Terraform). Sie können auch einen Cluster aus dem AWS-Managementkonsole erstellen. Eine Liste der Funktionen von Amazon EKS finden Sie unter Amazon-EKS-Feature. Zu den Kubernetes-Aufgaben, die Amazon EKS für Sie übernimmt, gehören:

- Verwaltete Steuerebene – AWS stellt sicher, dass der Amazon-EKS-Cluster verfügbar und skalierbar ist, da es die Steuerebene für Sie verwaltet und sie über Availability Zones von AWS hinweg verfügbar macht.
- Knotenverwaltung – Anstatt Knoten manuell hinzuzufügen, können Sie Amazon EKS Knoten bei Bedarf automatisch erstellen lassen, indem Sie verwaltete Knotengruppen (siehe the section called “Verwaltete Knotengruppen”) oder Karpenter verwenden. Verwaltete Knotengruppen verfügen über Integrationen mit Kubernetes Cluster Autoscaling. Mithilfe von Knotenverwaltungstools können Sie Kosteneinsparungen erzielen, beispielsweise durch Spot Instances und Knotenkonsolidierung, sowie die Verfügbarkeit verbessern, indem Sie Feature zur Planung verwenden, um festzulegen, wie Workloads bereitgestellt und Knoten ausgewählt werden.
- Cluster-Netzwerk – `eksctl` richtet mithilfe von CloudFormation-Vorlagen das Netzwerk zwischen Steuerebenen- und Datenebenenkomponenten (Knoten) im Kubernetes-Cluster ein. Außerdem werden Endpunkte für die interne und externe Kommunikation eingerichtet. Weitere Informationen finden Sie unter Entmystifizierung von Cluster-Netzwerken für Amazon-EKS-Worker-Knoten. Die Kommunikation zwischen Pods in Amazon EKS erfolgt mithilfe von Amazon EKS Pod Identities (siehe the section called “Pod Identity”), wodurch Pods auf AWS-Cloud-Methoden zur Verwaltung von Anmeldeinformationen und Berechtigungen zugreifen können.
- Add-Ons – Mit Amazon EKS entfällt die Notwendigkeit, Software-Komponenten zu erstellen und hinzuzufügen, die üblicherweise zur Unterstützung von Kubernetes-Clustern verwendet werden. Wenn Sie beispielsweise einen Amazon-EKS-Cluster aus dem AWS-Managementkonsole erstellen, werden automatisch die Amazon-EKS-Add-Ons kube-proxy (the section called “kube-

proxy"), Amazon-VPC-CNI-Plugin für Kubernetes (the section called "Amazon VPC CNI") und CoreDNS (the section called "CoreDNS") hinzugefügt. Weitere Informationen zu diesen Add-Ons, einschließlich einer Liste der verfügbaren Add-Ons, finden Sie unter the section called "Amazon-EKS-Add-ons".

Um Ihre Cluster auf Ihren eigenen On-Premises-Computern und -Netzwerken auszuführen, bietet Amazon Amazon EKS Anywhere an. Anstatt die AWS Cloud als Anbieter zu verwenden, haben Sie die Möglichkeit, Amazon EKS Anywhere in VMWare vSphere, bare metal (Tinkerbell-Anbieter), Snow, CloudStack oder Nutanix-Plattformen mithilfe Ihrer eigenen Ausrüstung auszuführen.

Amazon EKS Anywhere basiert auf derselben Amazon EKS Distro-Software, die auch von Amazon EKS verwendet wird. Amazon EKS Anywhere basiert jedoch auf verschiedenen Implementierungen der Kubernetes Cluster API (CAPI)-Benutzeroberfläche, um den gesamten Lebenszyklus der Rechner in einem Cluster von Amazon EKS Anywhere zu verwalten (z. B. CAPV für vSphere und CAPC für CloudStack). Da der gesamte Cluster auf Ihrer Ausrüstung ausgeführt wird, übernehmen Sie die zusätzliche Verantwortung für die Verwaltung der Steuerebene und die Datensicherung (siehe `etcd` weiter unten in diesem Dokument).

Cluster-Komponenten

Die Komponenten eines Kubernetes-Clusters lassen sich in zwei Hauptbereiche unterteilen: die Steuerebene und die Worker-Knoten. Komponente der Steuerebene verwalten den Cluster und bieten Zugriff auf seine APIs. Worker-Knoten (manchmal auch einfach als Knoten bezeichnet) stellen die Orte bereit, an denen die eigentlichen Workloads ausgeführt werden. Knotenkomponente bestehen aus Services, die in jedem Knoten ausgeführt werden, um mit der Steuerebene zu kommunizieren und Container auszuführen. Die Gruppe der Worker-Knoten, die für Ihren Cluster festgelegt ist, wird als Datenebene bezeichnet.

Steuerebene

Die Steuerebene besteht aus einer Reihe von Services, die den Cluster verwalten. Diese Services können alle auf einem einzigen Computer ausgeführt werden oder auf mehrere Computer verteilt sein. Intern werden diese als Instances der Steuerebene (CPIs) bezeichnet. Die Ausführung von CPIs hängt von der Größe des Clusters und den Anforderungen an die Hochverfügbarkeit ab. Bei steigender Nachfrage im Cluster kann ein Service der Steuerebene skaliert werden, um mehr Instances dieses Services bereitzustellen, wobei die Anfragen zwischen den Instances lastverteilt werden.

Zu den Aufgaben, die von Komponenten der Kubernetes-Steuerebene ausgeführt werden, gehören:

- Kommunikation mit Cluster-Komponenten (API-Server) – Der API-Server (kube-apiserver) stellt die Kubernetes-API bereit, sodass Anfragen an den Cluster sowohl innerhalb als auch außerhalb des Clusters gestellt werden können. Mit anderen Worten: Anforderungen zum Hinzufügen oder Ändern von Objekten eines Clusters (Pods, Services, Knoten usw.) können von externen Befehlen stammen, beispielsweise Anforderungen von `kubectl` zum Ausführen eines Pods. Ebenso können vom API-Server Anfragen an Komponenten innerhalb des Clusters gestellt werden, beispielsweise eine Abfrage an den `kubelet`-Service zum Status eines Pods.
- Speichern von Daten über den Cluster (**`etcd`**-Schlüsselwertspeicher) – Der `etcd`-Service übernimmt die wichtige Aufgabe, den aktuellen Status des Clusters zu verfolgen. Wenn der `etcd`-Service nicht mehr erreichbar ist, können Sie den Status des Clusters nicht mehr aktualisieren oder abfragen, obwohl die Workloads noch eine Weile ausgeführt werden. Aus diesem Grund werden in kritischen Clustern in der Regel mehrere Instances des `etcd`-Services mit Lastausgleich gleichzeitig ausgeführt und regelmäßige Backups des `etcd`-Schlüsselwertspeichers für den Fall eines Datenverlusts oder einer Datenbeschädigung durchgeführt. Denken Sie daran, dass dies alles in Amazon EKS standardmäßig automatisch für Sie erledigt wird. Amazon EKS Anywhere bietet Anweisungen für die Sicherung und Wiederherstellung von etcd. Informationen darüber, wie `etcd` Daten verwaltet, finden Sie im etcd-Datenmodell.
- Pods für Knoten planen (Scheduler) – Anfragen zum Starten oder Stoppen eines Pods in Kubernetes werden an den Kubernetes Scheduler (kube-scheduler) weitergeleitet. Da ein Cluster über mehrere Knoten verfügen kann, auf denen der Pod ausgeführt werden kann, muss Scheduler auswählen, in welchem Knoten (oder auf welchen Knoten im Falle von Replikaten) der Pod ausgeführt werden soll. Wenn nicht genügend Kapazität verfügbar ist, um den angeforderten Pod in einem vorhandenen Knoten auszuführen, schlägt die Anforderung fehl, sofern Sie keine anderen Vorkehrungen getroffen haben. Zu diesen Vorkehrungen könnte die Aktivierung von Services wie verwaltete Knotengruppen (the section called “Verwaltete Knotengruppen”) oder Karpenter gehören, die automatisch neue Knoten starten können, um die Workloads zu bewältigen.
- Komponenten im gewünschten Zustand halten (Controller Manager) – Der Kubernetes Controller Manager wird als Daemon-Prozess (kube-controller-manager) ausgeführt, um den Zustand des Clusters zu überwachen und Änderungen am Cluster vorzunehmen, um die erwarteten Zustände wiederherzustellen. Insbesondere gibt es mehrere Controller, die verschiedene Kubernetes-Objekte überwachen, darunter `statefulset-controller`, `endpoint-controller`, `cronjob-controller`, `node-controller` und andere.
- Verwalten von Cloud-Ressourcen (Cloud Controller Manager) – Die Interaktionen zwischen Kubernetes und dem Cloud-Anbieter, der die Anfragen für die zugrunde liegenden Rechenzentrumsressourcen ausführt, werden vom Cloud Controller Manager (cloud-controller-manager) verwaltet Zu den vom Cloud Controller Manager verwalteten Controllern können ein

Route Controller (für die Einrichtung von Cloud-Netzwerkweiterleitungen), ein Service Controller (für die Nutzung von Cloud-Lastausgleichsservices) und ein Knoten-Lebenszyklus-Controller (um Knoten während ihres Lebenszyklus mit Kubernetes synchron zu halten) gehören.

Worker-Knoten (Datenebene)

Bei einem Kubernetes-Cluster mit einem Knoten werden Workloads auf demselben Rechner wie die Steuerebene ausgeführt. Eine gängigere Konfiguration ist jedoch ein oder mehrere separate Computersysteme (Knoten), die für die Ausführung von Kubernetes-Workloads vorgesehen sind.

Wenn Sie zum ersten Mal einen Kubernetes-Cluster erstellen, können Sie mit einigen Tools zur Cluster-Erstellung eine bestimmte Anzahl von Knoten konfigurieren, die dem Cluster hinzugefügt werden sollen (entweder durch die Identifizierung vorhandener Computersysteme oder durch die Erstellung neuer Systeme durch den Anbieter). Bevor Workloads zu diesen Systemen hinzugefügt werden, werden jedem Knoten Services hinzugefügt, um diese Features zu implementieren:

- Knoten einzeln verwalten (**`kubelet`**) – Der API-Server kommuniziert mit dem Kubelet-Service, der in jedem Knoten ausgeführt wird, um sicherzustellen, dass der Knoten ordnungsgemäß registriert ist und die vom Scheduler angeforderten Pods ausgeführt werden. Das Kubelet kann die Pod-Manifeste lesen und Speicher-Volumes oder andere von den Pods benötigte Feature auf dem lokalen System einrichten. Es kann auch den Zustand der lokal ausgeführten Container überprüfen.
- Container in einem Knoten ausführen (Container-Laufzeit) – Die Container-Laufzeit in jedem Knoten verwaltet die Container, die für jeden dem Knoten zugewiesenen Pod angefordert werden. Das bedeutet, dass es Container-Images aus der entsprechenden Registry abrufen, den Container starten und anhalten kann und auf Anfragen zum Container antwortet. Die Standard-Container-Laufzeit ist containerd. Ab Kubernetes 1.24 wurde die spezielle Integration von Docker (`dockershim`), die als Container-Laufzeit verwendet werden konnte, aus Kubernetes entfernt. Sie können Docker zwar weiterhin zum Testen und Ausführen von Containern auf Ihrem lokalen System verwenden, aber um Docker mit Kubernetes zu verwenden, müssen Sie jetzt Docker Engine auf jedem Knoten installieren, um es mit Kubernetes zu verwenden.
- Netzwerk zwischen Containern verwalten (**`kube-proxy`**) – Um die Kommunikation zwischen Pods zu unterstützen, verwendet Kubernetes eine als Service bezeichnetes Feature zum Einrichten von Pod-Netzwerken, welche die mit diesen Pods verknüpften IP-Adressen und Ports verfolgen. Der Service kube-proxy wird in jedem Knoten ausgeführt, damit die Kommunikation zwischen Pods stattfinden kann.

Cluster erweitern

Es gibt einige Services, die Sie zu Kubernetes hinzufügen können, um den Cluster zu unterstützen, die aber nicht in der Steuerebene ausgeführt werden. Diese Services werden oft direkt in Knoten im kube-system-Namespace oder in einem eigenen Namespace ausgeführt (wie dies bei Drittanbietern oft der Fall ist). Ein gängiges Beispiel ist der CoreDNS-Service, der DNS-Services für den Cluster bereitstellt. Informationen dazu, wie Sie feststellen können, welche Cluster-Services in Ihrem Cluster im Kube-System ausgeführt werden, finden Sie unter Integrierte Services ermitteln.

Es gibt verschiedene Arten von Add-Ons, die Sie Ihren Clustern hinzufügen können. Um die Funktionsfähigkeit Ihrer Cluster zu gewährleisten, können Sie Beobachtbarkeitsfunktionen hinzufügen (siehe Cluster überwachen), mit denen Sie beispielsweise Protokollierung, Überwachung und Metriken durchführen können. Mit diesen Informationen können Sie auftretende Probleme beheben, oft über dieselben Beobachtbarkeitsschnittstellen. Beispiele für diese Art von Services sind Amazon GuardDuty CloudWatch (siehe the section called “Amazon CloudWatch”), AWS Distro for OpenTelemetry, Amazon VPC CNI Plugin für Kubernetes (siehe the section called “Amazon VPC CNI”) und Grafana Kubernetes Monitoring. Zu den Add-Ons für Amazon EKS im Bereich Speicher (siehe Anwendungsdatenspeicher) gehören Amazon Elastic Block Store CSI Driver (siehe the section called “Amazon EBS”), Amazon Elastic File System CSI Driver (siehe the section called “Amazon EFS”) und mehrere Speicher-Add-Ons von Drittanbietern wie Amazon FSx für NetApp-ONTAP-CSI-Treiber the section called “Amazon FSx für NetApp ONTAP”.

Eine umfassendere Liste der verfügbaren Amazon-EKS-Add-Ons finden Sie unter the section called “Amazon-EKS-Add-ons”.

Workloads

Kubernetes definiert einen Workload als „eine in Kubernetes ausgeführte Anwendung“. Diese Anwendung kann aus einer Reihe von Microservices bestehen, die als Container in Pods ausgeführt werden, oder als Batch-Auftrag oder als eine andere Art von Anwendung. Kubernetes stellt sicher, dass Ihre Anforderungen für die Einrichtung oder Bereitstellung dieser Objekte ausgeführt werden. Als jemand, der Anwendungen bereitstellt, sollten Sie lernen, wie Container erstellt werden, wie Pods definiert werden und welche Methoden Sie für deren Bereitstellung verwenden können.

Container

Das grundlegendste Element eines Anwendungs-Workloads, den Sie in Kubernetes bereitstellen und verwalten, ist ein Pod . Ein Pod stellt eine Möglichkeit dar, die Komponenten einer Anwendung zu

speichern und Spezifikationen zu definieren, die die Attribute des Pods beschreiben. Im Gegensatz dazu bietet beispielsweise ein RPM- oder Deb-Paket Software für ein Linux-System, das selbst jedoch nicht als Einheit ausgeführt wird.

Da der Pod die kleinste bereitstellbare Einheit ist, enthält er in der Regel einen einzigen Container. In Fällen, in denen Container eng miteinander verbunden sind, können sich jedoch mehrere Container in einem Pod befinden. Ein Webserver-Container kann beispielsweise in einem Pod mit einem Sidecar-Container verpackt sein, der die Protokollierung, Überwachung oder einen anderen Service bereitstellt, der eng mit dem Webserver-Container verbunden ist. In diesem Fall wird durch die Zugehörigkeit zum selben Pod sichergestellt, dass für jede ausgeführte Instance des Pods beide Container immer auf demselben Knoten ausgeführt werden. Ebenso teilen sich alle Container in einem Pod die gleiche Umgebung, wobei die Container in einem Pod so ausgeführt werden, als würden sie sich in demselben isolierten Host befinden. Dies hat zur Folge, dass die Container eine einzige IP-Adresse gemeinsam nutzen, die Zugriff auf den Pod bietet, und die Container miteinander kommunizieren können, als würden sie auf ihrem eigenen lokalen Host ausgeführt.

Pod-Spezifikationen (PodSpec) definieren den gewünschten Zustand des Pods. Sie können einen einzelnen Pod oder mehrere Pods bereitstellen, indem Sie Workload-Ressourcen zur Verwaltung von Pod-Vorlagen verwenden. Zu den Workload-Ressourcen gehören Bereitstellungen (zum Verwalten mehrerer Pod-Replikate), StatefulSets (zum Bereitstellen von Pods, die eindeutig sein müssen, z. B. Datenbank-Pods) und DaemonSets (wobei ein Pod kontinuierlich auf jedem Knoten ausgeführt werden muss). Dazu später mehr.

Während ein Pod die kleinste Einheit ist, die Sie bereitstellen, ist ein Container die kleinste Einheit, die Sie entwickeln und verwalten.

Entwicklung von Containern

Der Pod ist eigentlich nur eine Struktur um einen oder mehrere Container, wobei jeder Container selbst das Dateisystem, ausführbare Dateien, Konfigurationsdateien, Bibliotheken und andere Komponenten enthält, um die Anwendung tatsächlich auszuführen. Da Container erstmals von einem Unternehmen namens Docker Inc. eingeführt wurden, werden sie manchmal als Docker-Container bezeichnet. Inzwischen hat die Open Container Initiative seitdem Container-Laufzeiten, Images und Verteilungsmethoden für die Branche definiert. Hinzu kommt, dass Container aus vielen vorhandenen Linux-Features erstellt wurden. Andere bezeichnen Container oft als OCI-Container, Linux-Container oder einfach nur als Container.

Wenn Sie einen Container entwickeln, beginnen Sie in der Regel mit einer Docker-Datei (die wörtlich so heißt). Innerhalb dieser Docker-Datei identifizieren Sie:

- Ein Basis-Image – Ein Basis-Container-Image ist ein Container, der normalerweise entweder aus einer Minimalversion des Dateisystems eines Betriebssystems (wie Red Hat Enterprise Linux oder Ubuntu) oder einem Minimalsystem erstellt wird, das erweitert wurde, um Software zum Ausführen bestimmter Arten von Anwendungen bereitzustellen (wie Node.js- oder Python-Apps).
- Anwendungssoftware – Sie können Ihre Anwendungssoftware Ihrem Container auf die gleiche Weise hinzufügen, wie Sie sie einem Linux-System hinzufügen würden. Beispielsweise können Sie in Ihrer Docker-Datei `npm` und `yarn` ausführen, um eine Java-Anwendung zu installieren, oder `yum` und `dnf`, um RPM-Pakete zu installieren. Anders ausgedrückt: Mit einem RUN-Befehl in einer Dockerdatei können Sie jeden Befehl ausführen, der im Dateisystem Ihres Basis-Images verfügbar ist, um Software zu installieren oder innerhalb des resultierenden Container-Images zu konfigurieren.
- Anweisungen – Die Dockerfile-Referenz beschreibt die Anweisungen, die Sie einer Docker-Datei hinzufügen können, wenn Sie es konfigurieren. Dazu gehören Anweisungen zum Erstellen des Inhalts des Containers selbst (`ADD` oder `COPY`-Dateien aus dem lokalen System), zum Identifizieren von Befehlen, die beim Ausführen des Containers ausgeführt werden sollen (`CMD` oder `ENTRYPOINT`), und zum Verbinden des Containers mit dem System, in dem er ausgeführt wird (durch Identifizieren des auszuführenden `USER`, eines einzubindenden lokalen `VOLUME` oder der Ports zu `EXPOSE`).

Während der `docker`-Befehl und der Service üblicherweise zur Erstellung von Containern verwendet werden (`docker build`), gibt es auch andere Tools wie podman und nerdctl, die zur Erstellung von Container-Images verwendet werden können. Weitere Informationen zur Erstellung von Containern finden Sie unter Entwicklung besserer Container-Images oder Überblick über Docker Build.

Aufbewahrungscontainer

Sobald Sie Ihr Container-Image entwickelt haben, können Sie es in einer Registry für die Container-Verteilung in Ihrer Workstation oder in einer öffentlichen Container-Registry speichern. Durch die Ausführung einer privaten Container-Registry in Ihrer Workstation können Sie Container-Images lokal speichern und haben so jederzeit Zugriff darauf.

Um Container-Images auf eine öffentlichere Art und Weise zu speichern, können Sie sie in eine öffentliche Container-Registry verschieben. Öffentliche Container-Registries bieten einen zentralen Ort zum Speichern und Verteilen von Container-Images. Beispiele für öffentliche Container-Registrys sind die Amazon Elastic Container Registry, Red Hat Quay Registry und Docker Hub Registry.

Bei Ausführung von containerisierten Workloads in Amazon Elastic Kubernetes Service (Amazon EKS) empfehlen wir, Kopien der offiziellen Docker-Images abzurufen, die in der Amazon Elastic

Container Registry gespeichert sind. Amazon ECR speichert diese Images seit 2021. Sie können in der Amazon ECR Public Gallery nach beliebten Container-Images suchen, und speziell nach den Docker-Hub-Images in der Amazon ECR Docker Gallery suchen.

Ausführung von Containern

Da Container in einem Standardformat entwickelt werden, kann ein Container auf jedem Rechner ausgeführt werden, auf dem eine Container-Laufzeitumgebung (wie Docker) ausgeführt werden kann und dessen Inhalt der Architektur des lokalen Rechners entspricht (z. B. `x86_64` oder `arm`). Um einen Container zu testen oder ihn einfach auf Ihrem lokalen Desktop auszuführen, können Sie die Befehle `docker run` oder `podman run` verwenden, um einen Container auf dem lokalen Host zu starten. Bei Kubernetes hingegen ist auf jedem Worker-Knoten eine Container-Laufzeitumgebung bereitgestellt und es liegt an Kubernetes, einen Knoten aufzufordern, einen Container auszuführen.

Sobald ein Container zur Ausführung auf einem Knoten zugewiesen wurde, prüft der Knoten, ob die angeforderte Version des Container-Images bereits auf dem Knoten vorhanden ist. Ist dies nicht der Fall, weist Kubernetes die Container-Laufzeitumgebung an, den Container aus der entsprechenden Container-Registry abzurufen und ihn dann lokal auszuführen. Beachten Sie, dass sich ein Container-Image auf das Softwarepaket bezieht, das zwischen Ihrem Laptop, der Container-Registry und den Kubernetes-Knoten verschoben wird. Ein Container bezieht sich auf eine ausgeführte Instance dieses Images.

Pods

Sobald Ihre Container bereit sind, umfasst die Arbeit mit Pods das Konfigurieren, Bereitstellen und Zugänglichmachen der Pods.

Konfigurierung von Pods

Wenn Sie einen Pod definieren, weisen Sie ihm eine Reihe von Attributen zu. Diese Attribute müssen mindestens den Pod-Namen und das auszuführende Container-Image enthalten. Es gibt jedoch noch viele weitere Dinge, die Sie mit Ihren Pod-Definitionen konfigurieren möchten (Details dazu, was in einen Pod aufgenommen werden kann, finden Sie auf der Seite PodSpec). Dazu zählen:

- Speicher – Wenn ein ausgeführter Container angehalten und gelöscht wird, verschwindet der Datenspeicher in diesem Container, sofern Sie keinen dauerhafteren Speicher einrichten. Kubernetes unterstützt viele verschiedene Speicher und fasst diese unter dem Begriff Volumes zusammen. Zu den Speichertypen gehören CephFS, NFS, iSCSI und andere. Sie können sogar ein lokales Blockgerät vom lokalen Computer aus verwenden. Mit einem dieser in Ihrem

Cluster verfügbaren Speichertypen können Sie das Speichervolume an einem ausgewählten Einbindepunkt im Dateisystem Ihres Containers einbinden. Ein persistentes Volume ist ein Volume, das nach dem Löschen des Pods weiterhin vorhanden ist, während ein flüchtiges Volume gelöscht wird, wenn der Pod gelöscht wird. Wenn Ihr Cluster-Administrator verschiedene Speicherklassen für Ihren Cluster erstellt hat, haben Sie möglicherweise die Möglichkeit, die Attribute des von Ihnen verwendeten Speichers auszuwählen, z. B. ob das Volume nach der Verwendung gelöscht oder wiederhergestellt wird, ob es erweitert wird, wenn mehr Speicherplatz benötigt wird, und sogar, ob es bestimmte Leistungsanforderungen erfüllt

- Geheimnisse – Indem Sie Containern in Pod-Spezifikationen Geheimnisse zur Verfügung stellen, können Sie diesen Containern die Berechtigungen gewähren, die sie für den Zugriff auf Dateisysteme, Datenbanken oder andere geschützte Komponenten benötigen. Schlüssel, Passwörter und Token gehören zu den Elementen, die als Geheimnisse gespeichert werden können. Durch die Verwendung von Geheimnissen müssen Sie diese Informationen nicht in Container-Images speichern, sondern müssen sie nur ausgeführten Containern zur Verfügung stellen. Ähnlich wie Geheimnisse sind ConfigMaps. Ein `ConfigMap` enthält in der Regel weniger kritische Informationen, wie beispielsweise Schlüssel-Wert-Paare zur Konfiguration eines Service.
- Container-Ressourcen – Objekte zur weiteren Konfiguration von Containern können in Form von Ressourcenkonfigurationen vorliegen. Sie können für jeden Container die Speicher- und CPU-Menge anfordern, die er verwenden kann, und auch Beschränkungen für die Gesamtmenge dieser Ressourcen festlegen, die der Container verwenden kann. Beispiele finden Sie unter Ressource-Verwaltung für Pods und Container.
- Störungen – Pods können unfreiwillig (wenn ein Knoten ausfällt) oder freiwillig (wenn ein Upgrade gewünscht wird) unterbrochen werden. Durch die Konfiguration eines Budgets für Pod-Unterbrechungen können Sie eine gewisse Kontrolle darüber ausüben, wie verfügbar Ihre Anwendung bleibt, wenn Störungen auftreten. Beispiele finden Sie unter Festlegen eines Budgets für Unterbrechung für Ihre Anwendung.
- Namespaces – Kubernetes bietet verschiedene Möglichkeiten, Kubernetes-Komponenten und -Workloads voneinander zu isolieren. Das Ausführen aller Pods für eine bestimmte Anwendung im selben Namespace ist eine gängige Methode, um diese Pods gemeinsam zu sichern und zu verwalten. Sie können Ihre eigenen Namespaces erstellen oder keinen Namespace angeben (was dazu führt, dass Kubernetes den `default`-Namespace verwendet). Die Komponenten der Kubernetes-Steuerebene werden in der Regel im Namespace kube-system ausgeführt.

Die soeben beschriebene Konfiguration wird in der Regel in einer YAML-Datei zusammengefasst, die auf den Kubernetes-Cluster angewendet wird. Für persönliche Kubernetes-Cluster können Sie diese

YAML-Dateien einfach auf Ihrem lokalen System speichern. Bei kritischeren Clustern und Workloads ist GitOps jedoch eine beliebte Methode, um sowohl den Speicher als auch die Aktualisierung von Workloads- und Kubernetes-Infrastrukturressourcen zu automatisieren.

Die zum Erfassen und Bereitstellen von Pod-Informationen verwendeten Objekte werden durch eine der folgenden Bereitstellungsmethoden definiert.

Bereitstellen von Pods

Die Methode, die Sie zum Bereitstellen von Pods wählen, hängt von der Art der Anwendung ab, die Sie mit diesen Pods ausführen möchten. Hier finden Sie einige Optionen:

- Zustandslose Anwendungen – Eine zustandslose Anwendung speichert die Sitzungsdaten eines Clients nicht, sodass eine andere Sitzung nicht auf die Vorgänge einer vorherigen Sitzung zurückgreifen muss. Dies vereinfacht das Ersetzen von Pods durch neue, wenn sie fehlerhaft werden, oder das Verschieben ohne Speicherung des Zustands. Wenn Sie eine zustandslose Anwendung (z. B. einen Webserver) ausführen, können Sie eine Bereitstellung verwenden, um Pods und ReplicaSets bereitzustellen. Ein ReplicaSet legt fest, wie viele Instances eines Pods gleichzeitig ausgeführt werden sollen. Obwohl Sie ein ReplicaSet direkt ausführen können, ist es üblich, Replikate direkt innerhalb einer Bereitstellung auszuführen, um zu definieren, wie viele Replikate eines Pods gleichzeitig ausgeführt werden sollen.
- Zustandsbehaftete Anwendungen – Bei einer zustandsbehafteten Anwendung sind die Identität des Pods und die Reihenfolge, in der die Pods gestartet werden, wichtig. Diese Anwendungen benötigen einen stabilen persistenten Speicher und müssen auf konsistente Weise bereitgestellt und skaliert werden. Um eine zustandsbehaftete Anwendung in Kubernetes bereitzustellen, können Sie StatefulSets verwenden. Ein Beispiel für eine Anwendung, die typischerweise als StatefulSet ausgeführt wird, ist eine Datenbank. Innerhalb eines StatefulSet können Sie Replikate, den Pod und seine Container, einzubindende Speicher sowie Standorte im Container definieren, an denen Daten gespeichert werden. Ein Beispiel für eine Datenbank, die als ReplicaSet bereitgestellt wird, finden Sie unter Ausführen einer replizierten zustandsbehafteten Anwendung.
- Anwendungen pro Knoten – Es kann vorkommen, dass Sie eine Anwendung auf jedem Knoten in Ihrem Kubernetes-Cluster ausführen möchten. Beispielsweise könnte Ihr Rechenzentrum die Ausführung einer Überwachungsanwendung oder eines bestimmten Fernzugriffs auf jedem Computer verlangen. Für Kubernetes können Sie DaemonSet verwenden, um sicherzustellen, dass die ausgewählte Anwendung auf jedem Knoten in Ihrem Cluster ausgeführt wird.
- Anwendungen werden vollständig ausgeführt – Es gibt einige Anwendungen, die Sie ausführen möchten, um eine bestimmte Aufgabe abzuschließen. Dazu könnten beispielsweise Anwendungen gehören, die monatliche Statusberichte erstellen oder alte Daten bereinigen. Mit einem Objekt vom

Typ Job kann eine Anwendung so eingerichtet werden, dass sie gestartet und ausgeführt wird und nach Abschluss der Aufgabe beendet wird. Mit einem Objekt vom Typ CronJob können Sie eine Anwendung so einrichten, dass sie zu einer bestimmten Stunde, Minute, an einem bestimmten Tag im Monat, Monat oder Wochentag ausgeführt wird. Dabei wird eine Struktur verwendet, die durch das Linux-Format crontab definiert ist.

Anwendungen über das Netzwerk zugänglich machen

Da Anwendungen häufig als eine Reihe von Microservices bereitgestellt werden, die an verschiedene Orte verschoben werden, benötigte Kubernetes eine Möglichkeit, damit diese Microservices einander finden können. Damit andere auf eine Anwendung außerhalb des Kubernetes-Clusters zugreifen können, benötigte Kubernetes außerdem eine Möglichkeit, diese Anwendung über externe Adressen und Ports verfügbar zu machen. Diese netzwerkbezogenen Features werden mit Service- und Ingress-Objekten durchgeführt:

- Services – Da sich ein Pod zu verschiedenen Knoten und Adressen bewegen kann, kann es für einen anderen Pod, der mit dem ersten Pod kommunizieren muss, schwierig sein, dessen Standort zu ermitteln. Um dieses Problem zu lösen, ermöglicht Kubernetes die Darstellung einer Anwendung als Service. Mit einem Service können Sie einen Pod oder eine Gruppe von Pods mit einem bestimmten Namen identifizieren und dann angeben, welcher Port den Service dieser Anwendung vom Pod aus verfügbar macht und welche Ports eine andere Anwendung verwenden könnte, um diesen Service zu kontaktieren. Ein anderer Pod innerhalb eines Clusters kann einfach einen Service nach Namen anfordern und Kubernetes leitet diese Anforderung an den richtigen Port für eine Instance des Pods weiter, auf dem dieser Service ausgeführt wird.
- Ingress – Ingress ermöglicht es, Anwendungen, die durch Kubernetes-Services dargestellt werden, für Clients außerhalb des Clusters verfügbar zu machen. Zu den grundlegenden Features von Ingress gehören ein Load Balancer (verwaltet von Ingress), der Ingress-Controller und Regeln für die Weiterleitung von Anfragen vom Controller an den Service. Es gibt mehrere Ingress-Controller, aus denen Sie mit Kubernetes auswählen können.

Nächste Schritte

Das Verständnis der grundlegenden Kubernetes-Konzepte und ihrer Beziehung zu Amazon EKS wird Ihnen dabei helfen, sowohl die Amazon-EKS-Dokumentation als auch die Kubernetes-Dokumentation zu durchsuchen, um die Information zu finden, die Sie für die Verwaltung von Amazon-EKS-Clustern und das Bereitstellen von Workloads in diesen Clustern benötigen. Um mit der Nutzung von Amazon EKS zu beginnen, wählen Sie eine der folgenden Optionen aus:

- the section called “Cluster erstellen (eksctl)”
- the section called “Erstellen eines Clusters”
- the section called “Beispiel-Bereitstellung (Linux)”
- Clusterverwaltung

Bereitstellung von Amazon-EKS-Clustern in Cloud- und On-Premises-Umgebungen

Bereitstellungsoptionen von Amazon EKS kennenlernen

Amazon Elastic Kubernetes Service (Amazon EKS) ist ein vollständig verwalteter Kubernetes-Service, mit dem Sie Kubernetes nahtlos in der Cloud und in Ihren On-Premises-Umgebungen ausführen können.

In der Cloud automatisiert Amazon EKS die Verwaltung der Kubernetes-Cluster-Infrastruktur für die Kubernetes-Steuerebene und die Knoten. Dies ist für die Planung von Containern, die Verwaltung der Anwendungsverfügbarkeit, die dynamische Skalierung von Ressourcen, die Optimierung der Rechenleistung, die Speicherung von Cluster-Daten und die Ausführung anderer wichtiger Funktionen unerlässlich. Mit Amazon EKS profitieren Sie von einer stabilen Leistung, Skalierbarkeit, Zuverlässigkeit und Verfügbarkeit der AWS-Infrastruktur sowie von nativen Integrationen mit AWS-Netzwerk-, Sicherheits-, Speicher- und Beobachtbarkeitsservices.

Um die Ausführung von Kubernetes in Ihren On-Premises-Umgebungen zu vereinfachen, können Sie dieselben Amazon-EKS-Cluster, -Features und -Tools für the section called “Knoten” oder Amazon EKS Hybrid Nodes in Ihrer eigenen Infrastruktur verwenden oder Amazon EKS Anywhere für eigenständige Air-Gap-Umgebungen verwenden.

Amazon EKS in der Cloud

Sie können Amazon EKS mit Rechenleistung in AWS-Regionen, AWS Local Zones und AWS Wavelength Zones nutzen. Mit Amazon EKS in der Cloud werden die Sicherheit, Skalierbarkeit und Verfügbarkeit der Kubernetes- Steuerebene vollständig von AWS in der AWS-Region verwaltet. Wenn Sie Anwendungen mit Rechenleistung in AWS-Regionen ausführen, können Sie die gesamte Bandbreite von AWS- und Amazon-EKS-Features nutzen, einschließlich Amazon EKS Auto Mode, der die Verwaltung der Kubernetes-Cluster-Infrastruktur für Rechenleistung, Speicher und Netzwerk in AWS mit einem einzigen Klick vollständig automatisiert. Wenn Sie Anwendungen

mit Rechenleistung in AWS Local Zones und AWS Wavelength Zones ausführen, können Sie selbstverwaltete Amazon-EKS-Knoten verwenden, um Amazon-EC2-Instances für Ihre Cluster-Rechenleistung zu verbinden. Außerdem können Sie die anderen verfügbaren AWS-Services in AWS Local Zones und AWS Wavelength Zones nutzen. Weitere Informationen finden Sie unter Features für AWS Local Zones und Features für AWS Wavelength Zones.

	Amazon EKS in AWS-Regionen	Amazon EKS in Local/Wavelength Zones
Verwaltung der Kubernetes-Steuerebene	AWS-verwaltet	AWS-verwaltet
Ort der Kubernetes-Steuerebene	AWS-Regionen	AWS-Regionen
Kubernetes-Datenebene	• Amazon EKS Auto Mode • Von Amazon EKS verwaltete Knotengruppen • Selbstverwaltete Amazon-EC2-Knoten • AWS-Fargate	• Von Amazon EKS verwaltete Knotengruppen (nur Local Zones) • Selbstverwaltete Amazon-EC2-Knoten
Ort der Kubernetes-Datenebene	AWS-Regionen	AWS Local or Wavelength Zones

Amazon EKS in Ihrem Rechenzentrum oder in Ihren Edge-Umgebungen

Wenn Sie Anwendungen in Ihren eigenen Rechenzentren oder Edge-Umgebungen ausführen müssen, können Sie Amazon EKS in AWS Outposts oder Amazon EKS Hybrid Nodes verwenden. Sie können selbstverwaltete Knoten mit Amazon-EC2-Instances in AWS Outposts für Ihre Cluster-Berechnung verwenden oder Amazon EKS Hybrid Nodes mit Ihrer eigenen On-premises-oder Edge-Infrastruktur für Ihre Cluster-Berechnung verwenden. AWS Outposts ist eine von AWS verwaltete Infrastruktur, die Sie in Ihren Rechenzentren oder Co-Location-Einrichtungen betreiben, während Amazon EKS Hybrid Nodes auf Ihren physischen oder virtuellen Rechnern ausgeführt wird, die Sie in Ihren On-Premises- oder Edge-Umgebungen verwalten. Amazon EKS in AWS Outposts und Amazon EKS Hybrid Nodes erfordern eine zuverlässige Verbindung von Ihren On-Premises-Umgebungen zu

einer AWS-Region. Sie können dieselben Amazon-EKS-Cluster, -Feature und -Tools verwenden, die Sie auch für die Ausführung von Anwendungen in der Cloud einsetzen. Bei der Ausführung in AWS Outposts besteht die Möglichkeit, den gesamten Kubernetes-Cluster in AWS Outposts mit lokalen Amazon-EKS-Clustern in AWS Outposts bereitzustellen.

	Amazon EKS Hybrid Nodes	Amazon EKS in AWS Outposts
Verwaltung der Kubernetes-Steuerebene	AWS-verwaltet	AWS-verwaltet
Ort der Kubernetes-Steuerebene	AWS-Regionen	AWS Regions oder AWS Outposts
Kubernetes-Datenebene	Vom Kunden verwaltete physische oder virtuelle Rechner	Selbstverwaltete Amazon-EC2-Knoten
Ort der Kubernetes-Datenebene	Rechenzentrum oder Edge-Umgebung des Kunden	Rechenzentrum oder Edge-Umgebung des Kunden

Amazon EKS Anywhere für Air-Gapped-Umgebungen

Amazon EKS Anywhere vereinfacht die Verwaltung von Kubernetes-Clustern durch die Automatisierung undifferenzierter, aufwändiger Aufgaben wie der Einrichtung der Infrastruktur und der Verwaltung des Lebenszyklus von Kubernetes-Clustern in On-Premises- und Edge-Umgebungen. Im Gegensatz zu Amazon EKS ist Amazon EKS Anywhere ein vom Kunden verwaltetes Produkt, und die Kunden sind für den Lebenszyklusvorgang und die Wartung von Amazon-EKS-Anywhere-Clustern verantwortlich. Amazon EKS Anywhere basiert auf dem Kubernetes-Nebenprojekt Cluster API (CAPI) und unterstützt eine Reihe von Infrastrukturen, darunter VMware vSphere, Bare Metal, Nutanix, Apache CloudStack und AWS Snow. Amazon EKS Anywhere kann in Air-Gapped-Umgebungen ausgeführt werden und bietet optionale Integrationen mit regionalen AWS-Services für Beobachtbarkeit und Identitätsmanagement. Um Support für Amazon EKS Anywhere und Zugriff auf von AWS bereitgestellte Kubernetes-Add-Ons zu erhalten, können Sie Abonnements für Amazon EKS Anywhere Enterprise erwerben.

	Amazon EKS Anywhere
Verwaltung der Kubernetes-Steuerebene	Vom Kunden verwaltet
Ort der Kubernetes-Steuerebene	Rechenzentrum oder Edge-Umgebung des Kunden
Kubernetes-Datenebene	Vom Kunden verwaltete physische oder virtuelle Rechner
Ort der Kubernetes-Datenebene	Rechenzentrum oder Edge-Umgebung des Kunden

Amazon-EKS-Tools

Mit dem Amazon EKS Connector können Sie jeden konformen Kubernetes-Cluster bei AWS registrieren und verbinden und ihn in der Amazon-EKS-Konsole anzeigen. Nachdem ein Cluster verbunden wurde, können Sie den Status, die Konfiguration und die Workloads für diesen Cluster in der Amazon-EKS-Konsole anzeigen. Mit diesem Feature können Sie verbundene Cluster in der Amazon-EKS-Konsole anzeigen, jedoch ermöglicht Amazon EKS Connector keine Verwaltungs- oder Änderungsvorgänge für Ihre verbundenen Cluster über die Amazon-EKS-Konsole.

Amazon EKS Distro ist die AWS-Verteilung der zugrunde liegenden Kubernetes-Komponenten, die alle Amazon-EKS-Angebote unterstützen. Es enthält die Kernkomponenten, die für einen funktionierenden Kubernetes-Cluster erforderlich sind, wie z. B. Kubernetes-Steuerebenen-Komponenten (etcd, Kube-API-Server, Kube-Scheduler, Kube-Controller-Manager) und Netzwerkkomponenten (CoreDNS, Kube-Proxy, CNI-Plugins). Amazon EKS Distro kann zur Selbstverwaltung von Kubernetes-Clustern mit den Tools Ihrer Wahl verwendet werden. Bereitstellungen für Amazon EKS Distro sind nicht durch AWS-Support-Pläne abgedeckt.

Einrichtung zur Verwendung von Amazon EKS

Zur Vorbereitung der Befehlszeilenverwaltung Ihrer Amazon-EKS-Cluster müssen Sie mehrere Tools installieren. Mit den folgenden Schritten können Sie Anmeldeinformationen einrichten, Cluster erstellen und ändern sowie mit Clustern arbeiten, sobald diese ausgeführt werden:

- AWS-CLI einrichten – Rufen Sie die AWS-CLI zum Einrichten und Verwalten der Services ab, die Sie für die Arbeit mit Amazon-EKS-Clustern benötigen. Sie benötigen die AWS-CLI insbesondere zum Konfigurieren von Anmeldeinformationen, aber auch für andere AWS-Services.
- kubectl und eksctl einrichten – Die `eksctl`-CLI interagiert mit AWS, um Amazon-EKS-Cluster zu erstellen, zu ändern und zu löschen. Sobald ein Cluster aktiv ist, verwenden Sie den Open-Source-Befehl `kubectl`, um Kubernetes-Objekte innerhalb Ihrer Amazon-EKS-Cluster zu verwalten.
- Entwicklungsumgebung einrichten (optional) – Erwägen Sie, die folgenden Tools hinzuzufügen:
  - Lokales Bereitstellungstool – Wenn Sie noch nicht mit Kubernetes vertraut sind, empfiehlt es sich gegebenenfalls, ein lokales Bereitstellungstool wie minikube oder kind zu installieren. Mit diesen Tools können Sie einen Amazon-EKS-Cluster auf Ihrem lokalen Rechner einrichten, um Anwendungen zu testen.
  - Paket-Manager: – helm ist ein beliebter Paket-Manager für Kubernetes, der die Installation und Verwaltung komplexer Pakete vereinfacht. Mit Helm können Pakete wie AWS-Load-Balancer-Controller in Ihrem Amazon-EKS-Cluster einfacher installiert und verwaltet werden.

Nächste Schritte

- AWS-CLI einrichten
- kubectl und eksctl einrichten
- Schnellstart: Bereitstellen einer Web-App und Speichern von Daten

AWS-CLI einrichten

Die AWS-CLI ist ein Befehlszeilen-Tool für die Arbeit mit AWS-Services (einschließlich Amazon EKS). Es wird auch verwendet, um IAM-Benutzer oder -Rollen für den Zugriff auf den Amazon-EKS-Cluster sowie auf andere AWS-Ressourcen von Ihrem lokalen Computer zu authentifizieren. Um Ressourcen über die Befehlszeile in AWS bereitstellen zu können, benötigen Sie eine AWS-Zugriffsschlüssel-ID und einen geheimen Schlüssel für die Verwendung in der Befehlszeile.

Anschließend müssen Sie diese Anmeldeinformationen in der AWS-CLI konfigurieren. Wenn Sie die AWS-CLI noch nicht installiert haben, lesen Sie den Abschnitt Installieren oder Aktualisieren der neuesten Version der AWS-CLI im Benutzerhandbuch zur AWS-Befehlszeilenschnittstelle.

So erstellen Sie einen Zugriffsschlüssel

1. Melden Sie sich beim AWS-Managementkonsole an.
2. Für Einzelbenutzer- oder Mehrbenutzerkonten:
   - Einzelbenutzerkonto –:: Wählen Sie oben rechts Ihren AWS-Benutzernamen, um das Navigationsmenü zu öffnen. Wählen Sie zum Beispiel **`webadmin`** aus.
   - Mehrbenutzerkonto –:: Wählen Sie IAM aus der Liste der Services. Wählen Sie im IAM-Dashboard Benutzer und dann den Namen des Benutzers aus.
3. Wählen Sie Sicherheitsanmeldeinformationen aus.
4. Wählen Sie unter Zugriffsschlüssel die Option Zugriffsschlüssel erstellen aus.
5. Wählen Sie Befehlszeilenschnittstelle (CLI) und anschließend Weiter aus.
6. Wählen Sie Zugriffsschlüssel erstellen aus.
7. Wählen Sie CSV-Datei herunterladen aus.

Konfigurieren der AWS-CLI

Führen Sie nach dem Installieren der AWS-CLI die folgenden Schritte aus, um sie zu konfigurieren. Weitere Informationen finden Sie unter Konfiguration der AWS-CLI im Benutzerhandbuch zur AWS-Befehlszeilenschnittstelle.

1. Geben Sie in einem Terminal-Fenster den folgenden Befehl ein:

   ```
   aws configure
   ```

   Optional können Sie ein benanntes Profil konfigurieren, z. B. `--profile cluster-admin`. Wenn Sie ein benanntes Profil in der AWS-CLI konfigurieren, muss in nachfolgenden Befehlen immer dieses Flag übergeben werden.
2. Geben Sie Ihre AWS-Anmeldeinformationen ein. Zum Beispiel:

   ```
   Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
   Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
   ```

```
Default region name [None]: region-code
Default output format [None]: json
```

So rufen Sie ein Sicherheitstoken ab

Führen Sie bei Bedarf den folgenden Befehl aus, um ein neues Sicherheits-Token für die AWS-CLI abzurufen. Weitere Informationen finden Sie unter get-session-token in der AWS-CLI-Befehlsreferenz.

Das Token ist standardmäßig 15 Minuten lang gültig. Das standardmäßige Sitzungs-Timeout kann durch Übergeben des Flags `--duration-seconds` geändert werden. Zum Beispiel:

```
aws sts get-session-token --duration-seconds 3600
```

Dieser Befehl gibt die temporären Sicherheits-Anmeldeinformationen für eine AWS-CLI-Sitzung zurück. Die ausgegebene Antwort sollte wie folgt aussehen:

```
{
    "Credentials": {
        "AccessKeyId": "ASIA5FTRU3LOEXAMPLE",
        "SecretAccessKey": "JnKgvwfqUD9mNsPoi9IbxAYEXAMPLE",
        "SessionToken": "VERYLONGSESSIONTOKENSTRING",
        "Expiration": "2023-02-17T03:14:24+00:00"
    }
}
```

So überprüfen Sie die Benutzeridentität

Führen Sie bei Bedarf den folgenden Befehl aus, um die AWS-Anmeldeinformationen für Ihre IAM-Benutzeridentität (z. B. *`ClusterAdmin`*) für die Terminalsitzung zu überprüfen.

```
aws sts get-caller-identity
```

Dieser Befehl gibt den Amazon-Ressourcennamen (ARN) der IAM-Entität zurück, die für die AWS-CLI konfiguriert ist. Die ausgegebene Antwort sollte beispielsweise wie folgt aussehen:

```
{
```

```
    "UserId": "AKIAIOSFODNN7EXAMPLE",
    "Account": "01234567890",
    "Arn": "arn:aws:iam::01234567890:user/ClusterAdmin"
}
```

Nächste Schritte

- kubectl und eksctl einrichten
- Schnellstart: Bereitstellen einer Web-App und Speichern von Daten

kubectl und **eksctl** einrichten

> **Tip**
>
> Melden Sie sich für bevorstehende Amazon EKS-Workshops an.

Sobald die AWS CLI installiert ist, sollten Sie zwei weitere Tools installieren, um Ihre Kubernetes-Cluster zu erstellen und zu verwalten:

- `kubectl`: Das `kubectl`-Befehlszeilen-Tool ist das Haupt-Tool, das Sie zum Verwalten von Ressourcen in Ihrem Kubernetes-Cluster verwenden. Auf dieser Seite wird beschrieben, wie Sie die `kubectl`-Binärdatei herunterladen und einrichten, die der Version Ihres Kubernetes-Clusters entspricht. Weitere Informationen finden Sie unter Installieren oder Aktualisieren von kubectl.
- `eksctl`: Das `eksctl` Befehlszeilentool dient zum Erstellen von EKS-Clustern in der AWS Cloud oder vor Ort (mit EKS Anywhere) sowie zum Ändern und Löschen dieser Cluster. Lesen Sie Installieren von eksctl.

Installieren oder Aktualisieren von **kubectl**

In diesem Thema finden Sie Hinweise zum Herunterladen und Installieren der `kubectl`-Binärdatei auf Ihrem Gerät. Die Binärdatei ist identisch mit den Upstream-Gemeinschaftsversionen. Die Binärdatei gilt nicht nur für Amazon EKS oder AWS. Verwenden Sie die folgenden Schritte, um die spezifische Version von `kubectl` abzurufen, die Sie benötigen. Viele Entwickler führen einfach `brew install kubectl` aus, um es zu installieren.

> **Note**
>
> Sie müssen eine `kubectl`-Version verwenden, die nur in der Minor-Version von Ihrer Amazon–EKS-Cluster-Steuerebene abweicht. Beispielsweise sollte ein `1.32`-`kubectl`-Client mit Kubernetes `1.31`-, `1.32`- und `1.33`-Clustern funktionieren.

Schritt 1: Überprüfen, ob **`kubectl`** installiert ist

Stellen Sie fest, ob Sie `kubectl` bereits auf Ihrem Gerät installiert haben.

```
kubectl version --client
```

Wenn `kubectl` im Pfad Ihres Geräts installiert ist, enthält die Beispielausgabe ähnliche Informationen wie die folgende. Wenn Sie die Version, die Sie derzeit installiert haben, mit einer neueren Version aktualisieren möchten, führen Sie den nächsten Schritt aus und stellen Sie sicher, dass Sie die neue Version an demselben Speicherort installieren, an dem sich Ihre aktuelle Version befindet.

```
Client Version: v1.31.X-eks-1234567
```

Wenn Sie keine Ausgabe erhalten, ist `kubectl` entweder nicht installiert oder nicht an einem Ort installiert, der im Pfad Ihres Geräts enthalten ist.

Step 2: **`kubectl`** installieren oder aktualisieren

Installieren oder aktualisieren Sie `kubectl` auf einem der folgenden Betriebssysteme:

- the section called “macOS”
- the section called “Linux (amd64)”
- the section called “Linux (arm64)”
- the section called “Windows”

> **Note**
>
> Wenn die Downloads aus den in diesem Abschnitt verwendeten AWS Regionen in Ihre AWS Region langsam sind, sollten Sie erwägen, den Inhalt so einzurichten, CloudFront dass der

Inhalt zuerst angezeigt wird. Weitere Informationen finden Sie unter Erste Schritte mit einer CloudFront Basisdistribution.

macOS

Führen Sie die nachfolgenden Schritte aus, um die Installation von `kubectl` unter macOS durchzuführen. Zu diesen Schritten gehören:

- Auswählen und Herunterladen der Binärdatei für die gewünschte Kubernetes-Version.
- Optionales Überprüfen der Prüfsumme der Binärdatei.
- Hinzufügen der Ausführungsberechtigungen zur Binärdatei.
- Kopieren der Binärdatei in einen Ordner in Ihrem PATH.
- Hinzufügen des Verzeichnisses der Binärdatei zu Ihrem PATH.

Verfahren:

1. Laden Sie die Binärdatei für die Kubernetes-Version Ihres Clusters von Amazon S3 herunter.
   - Kubernetes `1.34`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.34.2/2025-11-13/bin/darwin/amd64/kubectl
     ```

   - Kubernetes `1.33`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.33.5/2025-11-13/bin/darwin/amd64/kubectl
     ```

   - Kubernetes `1.32`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.32.9/2025-11-13/bin/darwin/amd64/kubectl
     ```

   - Kubernetes `1.31`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.31.13/2025-11-13/bin/darwin/amd64/kubectl
     ```

   - Kubernetes `1.30`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.14/2025-11-13/bin/
darwin/amd64/kubectl
```

- Kubernetes `1.29`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.15/2025-11-13/bin/
darwin/amd64/kubectl
```

- Kubernetes `1.28`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.15/2025-11-13/bin/
darwin/amd64/kubectl
```

- Kubernetes `1.27`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.16/2025-01-10/bin/
darwin/amd64/kubectl
```

- Kubernetes `1.26`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-12-12/bin/
darwin/amd64/kubectl
```

2. (Optional) Überprüfen Sie die heruntergeladene Binärdatei mit der `SHA-256`-Prüfsumme für Ihre Binärdatei.

   a. Laden Sie die `SHA-256`-Prüfsumme für die Kubernetes-Version Ihres Clusters herunter.

   - Kubernetes `1.34`

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.34.2/2025-11-13/bin/
   darwin/amd64/kubectl.sha256
   ```

   - Kubernetes `1.33`

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.33.5/2025-11-13/bin/
   darwin/amd64/kubectl.sha256
   ```

   - Kubernetes `1.32`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.32.9/2025-11-13/bin/
darwin/amd64/kubectl.sha256
```

- Kubernetes `1.31`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.31.13/2025-11-13/bin/
darwin/amd64/kubectl.sha256
```

- Kubernetes `1.30`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.14/2025-11-13/bin/
darwin/amd64/kubectl.sha256
```

- Kubernetes `1.29`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.15/2025-11-13/bin/
darwin/amd64/kubectl.sha256
```

- Kubernetes `1.28`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.15/2025-11-13/bin/
darwin/amd64/kubectl.sha256
```

- Kubernetes `1.27`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.16/2025-01-10/bin/
darwin/amd64/kubectl.sha256
```

- Kubernetes `1.26`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-12-12/bin/
darwin/amd64/kubectl.sha256
```

b. Überprüfen Sie die `SHA-256`-Prüfsumme für Ihre heruntergeladene Binärdatei.

```
openssl sha1 -sha256 kubectl
```

c. Stellen Sie sicher, dass die generierte Prüfsumme in der Ausgabe mit der Prüfsumme in der heruntergeladenen `kubectl.sha256`-Datei übereinstimmt.

3. Wenden Sie Ausführungsberechtigungen auf die Binärdatei an.

```
chmod +x ./kubectl
```

4. Kopieren Sie die Binärdatei in ein Verzeichnis in Ihrer PATH-Variable. Wenn Sie bereits eine Version von kubectl installiert haben, empfehlen wir, eine $HOME/bin/kubectl-Datei zu erstellen und sicherzustellen, dass $HOME/bin in der $PATH-Variablen zuerst vorkommt.

```
mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$HOME/bin:$PATH
```

5. (Optional) Fügen Sie den Pfad $HOME/bin zu Ihrer Shell-Initialisierungsdatei hinzu, um den Pfad bereits beim Öffnen einer Shell zu konfigurieren.

```
echo 'export PATH=$HOME/bin:$PATH' >> ~/.bash_profile
```

Linux (amd64)

Befolgen Sie die folgenden Schritte, um die Installation von kubectl unter Linux (amd64) durchzuführen. Zu diesen Schritten gehören:

- Auswählen und Herunterladen der Binärdatei für die gewünschte Kubernetes-Version.
- Optionales Überprüfen der Prüfsumme der Binärdatei.
- Hinzufügen der Ausführungsberechtigungen zur Binärdatei.
- Kopieren der Binärdatei in einen Ordner in Ihrem PATH.
- Hinzufügen des Verzeichnisses der Binärdatei zu Ihrem PATH.

Verfahren:

1. Laden Sie die kubectl-Binärdatei für die Kubernetes-Version Ihres Clusters von Amazon S3 herunter.
   - Kubernetes 1.34

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.34.2/2025-11-13/bin/linux/amd64/kubectl
     ```

   - Kubernetes 1.33

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.33.5/2025-11-13/bin/linux/amd64/kubectl
```

- Kubernetes `1.32`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.32.9/2025-11-13/bin/linux/amd64/kubectl
```

- Kubernetes `1.31`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.31.13/2025-11-13/bin/linux/amd64/kubectl
```

- Kubernetes `1.30`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.14/2025-11-13/bin/linux/amd64/kubectl
```

- Kubernetes `1.29`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.15/2025-11-13/bin/linux/amd64/kubectl
```

- Kubernetes `1.28`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.15/2025-11-13/bin/linux/amd64/kubectl
```

- Kubernetes `1.27`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.16/2024-12-12/bin/linux/amd64/kubectl
```

- Kubernetes `1.26`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-12-12/bin/linux/amd64/kubectl
```

2. (Optional) Überprüfen Sie die heruntergeladene Binärdatei mit der `SHA-256`-Prüfsumme für Ihre Binärdatei.

a. Laden Sie die SHA-256-Prüfsumme für die Kubernetes-Version Ihres Clusters von Amazon S3 herunter, indem Sie den Befehl für die Hardware-Plattform Ihres Geräts verwenden.

- Kubernetes `1.34`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.34.2/2025-11-13/bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.33`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.33.5/2025-11-13/bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.32`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.32.9/2025-11-13/bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.31`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.31.13/2025-11-13/bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.30`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.14/2025-11-13/bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.29`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.15/2025-11-13/bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.28`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.15/2025-11-13/bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.27`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.16/2024-12-12/bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.26`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-12-12/bin/linux/amd64/kubectl.sha256
```

b. Überprüfen Sie die SHA-256-Prüfsumme für die heruntergeladene Binärdatei mit einem der folgenden Befehle.

```
sha256sum -c kubectl.sha256
```

oder

```
openssl sha1 -sha256 kubectl
```

c. Im ersten Fall sollte `kubectl: OK` angezeigt werden, im zweiten Fall können Sie überprüfen, ob die in der Ausgabe generierte Prüfsumme mit der Prüfsumme in der heruntergeladenen `kubectl.sha256`-Datei übereinstimmt.

3. Wenden Sie Ausführungsberechtigungen auf die Binärdatei an.

```
chmod +x ./kubectl
```

4. Kopieren Sie die Binärdatei in ein Verzeichnis in Ihrer PATH-Variable. Wenn Sie bereits eine Version von `kubectl` installiert haben, empfehlen wir, eine `$HOME/bin/kubectl`-Datei zu erstellen und sicherzustellen, dass `$HOME/bin` in der $PATH-Variablen zuerst vorkommt.

```
mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$HOME/bin:$PATH
```

5. (Optional) Fügen Sie den Pfad `$HOME/bin` zu Ihrer Shell-Initialisierungsdatei hinzu, um den Pfad bereits beim Öffnen einer Shell zu konfigurieren.

> **Note**
>
> Bei diesem Schritt wird davon ausgegangen, dass Sie das Bash-Shell verwenden. Wenn Sie eine andere Shell nutzen, ändern Sie den Befehl zur Angabe der spezifischen Shell-Initialisierungsdatei.

```
echo 'export PATH=$HOME/bin:$PATH' >> ~/.bashrc
```

Linux (arm64)

Befolgen Sie die folgenden Schritte, um die Installation von `kubectl` unter Linux (arm64) durchzuführen. Zu diesen Schritten gehören:

- Auswählen und Herunterladen der Binärdatei für die gewünschte Kubernetes-Version.
- Optionales Überprüfen der Prüfsumme der Binärdatei.
- Hinzufügen der Ausführungsberechtigungen zur Binärdatei.
- Kopieren der Binärdatei in einen Ordner in Ihrem PATH.
- Hinzufügen des Verzeichnisses der Binärdatei zu Ihrem PATH.

Verfahren:

1. Laden Sie die `kubectl`-Binärdatei für die Kubernetes-Version Ihres Clusters von Amazon S3 herunter.
   - Kubernetes `1.34`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.34.2/2025-11-13/bin/linux/arm64/kubectl
     ```

   - Kubernetes `1.33`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.33.5/2025-11-13/bin/linux/arm64/kubectl
     ```

   - Kubernetes `1.32`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.32.9/2025-11-13/bin/linux/arm64/kubectl
     ```

   - Kubernetes `1.31`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.31.13/2025-11-13/bin/linux/arm64/kubectl
```

- Kubernetes `1.30`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.14/2025-11-13/bin/linux/arm64/kubectl
```

- Kubernetes `1.29`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.15/2025-11-13/bin/linux/arm64/kubectl
```

- Kubernetes `1.28`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.15/2025-11-13/bin/linux/arm64/kubectl
```

- Kubernetes `1.27`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.16/2024-12-12/bin/linux/arm64/kubectl
```

- Kubernetes `1.26`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-12-12/bin/linux/arm64/kubectl
```

2. (Optional) Überprüfen Sie die heruntergeladene Binärdatei mit der `SHA-256`-Prüfsumme für Ihre Binärdatei.

   a. Laden Sie die `SHA-256`-Prüfsumme für die Kubernetes-Version Ihres Clusters von Amazon S3 herunter, indem Sie den Befehl für die Hardware-Plattform Ihres Geräts verwenden.

   - Kubernetes `1.34`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.34.2/2025-11-13/bin/linux/arm64/kubectl.sha256
     ```

   - Kubernetes `1.33`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.33.5/2025-11-13/bin/
linux/arm64/kubectl.sha256
```

- Kubernetes `1.32`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.32.9/2025-11-13/bin/
linux/arm64/kubectl.sha256
```

- Kubernetes `1.31`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.31.13/2025-11-13/bin/
linux/arm64/kubectl.sha256
```

- Kubernetes `1.30`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.14/2025-11-13/bin/
linux/arm64/kubectl.sha256
```

- Kubernetes `1.29`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.15/2025-11-13/bin/
linux/arm64/kubectl.sha256
```

- Kubernetes `1.28`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.15/2025-11-13/bin/
linux/arm64/kubectl.sha256
```

- Kubernetes `1.27`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.16/2024-12-12/bin/
linux/arm64/kubectl.sha256
```

- Kubernetes `1.26`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-12-12/bin/
linux/arm64/kubectl.sha256
```

b. Überprüfen Sie die `SHA-256`-Prüfsumme für die heruntergeladene Binärdatei mit einem der folgenden Befehle.

```
sha256sum -c kubectl.sha256
```

oder

```
openssl sha1 -sha256 kubectl
```

c. Im ersten Fall sollte `kubectl: OK` angezeigt werden, im zweiten Fall können Sie überprüfen, ob die in der Ausgabe generierte Prüfsumme mit der Prüfsumme in der heruntergeladenen `kubectl.sha256`-Datei übereinstimmt.

3. Wenden Sie Ausführungsberechtigungen auf die Binärdatei an.

```
chmod +x ./kubectl
```

4. Kopieren Sie die Binärdatei in ein Verzeichnis in Ihrer `PATH`-Variable. Wenn Sie bereits eine Version von `kubectl` installiert haben, empfehlen wir, eine `$HOME/bin/kubectl`-Datei zu erstellen und sicherzustellen, dass `$HOME/bin` in der `$PATH`-Variablen zuerst vorkommt.

```
mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$HOME/bin:$PATH
```

5. (Optional) Fügen Sie den Pfad `$HOME/bin` zu Ihrer Shell-Initialisierungsdatei hinzu, um den Pfad bereits beim Öffnen einer Shell zu konfigurieren.

> **Note**
>
> Bei diesem Schritt wird davon ausgegangen, dass Sie das Bash-Shell verwenden. Wenn Sie eine andere Shell nutzen, ändern Sie den Befehl zur Angabe der spezifischen Shell-Initialisierungsdatei.

```
echo 'export PATH=$HOME/bin:$PATH' >> ~/.bashrc
```

Windows

Befolgen Sie zur Installation von `kubectl` unter Windows die folgenden Schritte. Zu diesen Schritten gehören:

- Auswählen und Herunterladen der Binärdatei für die gewünschte Kubernetes-Version.

- Optionales Überprüfen der Prüfsumme der Binärdatei.
- Kopieren der Binärdatei in einen Ordner in Ihrem PATH.
- Hinzufügen des Verzeichnisses der Binärdatei zu Ihrem PATH.

Verfahren:

1. Öffnen Sie ein PowerShell Terminal.
2. Laden Sie die `kubectl`-Binärdatei für die Kubernetes-Version Ihres Clusters von Amazon S3 herunter.
   - Kubernetes `1.34`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.34.2/2025-11-13/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.33`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.33.5/2025-11-13/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.32`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.32.9/2025-11-13/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.31`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.31.13/2025-11-13/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.30`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.14/2025-11-13/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.29`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.15/2025-11-13/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.28`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.15/2025-11-13/bin/
windows/amd64/kubectl.exe
```

- Kubernetes `1.27`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.16/2024-12-12/bin/
windows/amd64/kubectl.exe
```

- Kubernetes `1.26`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-12-12/bin/
windows/amd64/kubectl.exe
```

3. (Optional) Überprüfen Sie die heruntergeladene Binärdatei mit der `SHA-256`-Prüfsumme für Ihre Binärdatei.

   a. Laden Sie die `SHA-256`-Prüfsumme für die Kubernetes-Version Ihres Clusters für Windows herunter.

   - Kubernetes `1.34`

   ```
   curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.34.2/2025-11-13/bin/
   windows/amd64/kubectl.exe.sha256
   ```

   - Kubernetes `1.33`

   ```
   curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.33.5/2025-11-13/bin/
   windows/amd64/kubectl.exe.sha256
   ```

   - Kubernetes `1.32`

   ```
   curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.32.9/2025-11-13/bin/
   windows/amd64/kubectl.exe.sha256
   ```

   - Kubernetes `1.31`

   ```
   curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.31.13/2025-11-13/
   bin/windows/amd64/kubectl.exe.sha256
   ```

   - Kubernetes `1.30`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.14/2025-11-13/
bin/windows/amd64/kubectl.exe.sha256
```

- Kubernetes `1.29`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.15/2025-11-13/
bin/windows/amd64/kubectl.exe.sha256
```

- Kubernetes `1.28`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.15/2025-11-13/
bin/windows/amd64/kubectl.exe.sha256
```

- Kubernetes `1.27`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.16/2024-12-12/
bin/windows/amd64/kubectl.exe.sha256
```

- Kubernetes `1.26`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-12-12/
bin/windows/amd64/kubectl.exe.sha256
```

b. Überprüfen Sie die `SHA-256`-Prüfsumme für Ihre heruntergeladene Binärdatei.

```
Get-FileHash kubectl.exe
```

c. Stellen Sie sicher, dass die generierte Prüfsumme in der Ausgabe mit der Prüfsumme in der heruntergeladenen `kubectl.sha256`-Datei übereinstimmt. Die PowerShell Ausgabe sollte eine entsprechende Zeichenfolge in Großbuchstaben sein.

4. Kopieren Sie die Binärdatei in ein Verzeichnis in Ihrer `PATH`-Variable. Wenn Sie in Ihrem `PATH` ein Verzeichnis vorhanden ist, das Sie für Befehlszeilen-Dienstprogramme verwenden, kopieren Sie die Binärdatei in dieses Verzeichnis. Führen Sie andernfalls die folgenden Schritte aus.

   a. Erstellen Sie ein neues Verzeichnis für Ihre Befehlszeilen-Binärdateien, z. B. `C:\bin`.

   b. Kopieren Sie die `kubectl.exe`-Binärdatei in Ihr neues Verzeichnis.

   c. Bearbeiten Sie Ihre Benutzer- oder System-`PATH`-Umgebungsvariable, um das neue Verzeichnis zu Ihrem `PATH` hinzuzufügen.

d. Schließen Sie Ihr PowerShell Terminal und öffnen Sie ein neues, um die neue `PATH` Variable aufzurufen.

5. Nach der Installation von `kubectl` können Sie die Version prüfen:

```
kubectl version --client
```

6. Bei der Erstinstallation von `kubectl` ist es noch nicht für die Kommunikation mit einem Server konfiguriert. Wir werden diese Konfiguration bei Bedarf in anderen Verfahren behandeln. Wenn Sie jemals die Konfiguration aktualisieren müssen, um mit einem bestimmten Cluster zu kommunizieren, können Sie den folgenden Befehl ausführen. *`region-code`*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet. Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters.

```
aws eks update-kubeconfig --region region-code --name my-cluster
```

7. Erwägen Sie die Konfiguration der automatischen Vervollständigung, mit der Sie die Tabulatortaste verwenden können, um `kubectl`-Unterbefehle nach der Eingabe der ersten Buchstaben zu vervollständigen. Weitere Informationen finden Sie unter Kubectl-Autovervollständigung in der Kubernetes-Dokumentation.

Installieren **`eksctl`**

Die `eksctl`-CLI wird zum Arbeiten mit EKS-Clustern verwendet. Sie automatisiert viele einzelne Aufgaben. Anweisungen zur Installation von `eksctl` finden Sie unter Installation in der `eksctl`-Dokumentation. Verwenden Sie für Linux die UNIX-Anweisungen.

Wenn Sie `eksctl` den von Ihnen verwendeten IAM-Sicherheitsprinzipal verwenden, müssen Sie über Berechtigungen für die Arbeit mit Amazon EKS IAM-Rollen, serviceverknüpften Rollen AWS CloudFormation, einer VPC und verwandten Ressourcen verfügen. Weitere Informationen finden Sie unter Aktionen und Verwendung von serviceverknüpften Rollen im IAM-Benutzerhandbuch. Sie müssen alle Schritte in diesem Handbuch als derselbe Benutzer ausführen. Führen Sie den folgenden Befehl aus, um den aktuellen Benutzer zu überprüfen:

```
aws sts get-caller-identity
```

Nächste Schritte

- Schnellstart: Bereitstellen einer Web-App und Speichern von Daten

Schnellstart: Bereitstellen einer Web-App und Speichern von Daten

Dieses Schnellstart-Tutorial führt Sie durch die Schritte, um die Beispielanwendung 2048 für das Spiel bereitzustellen und ihre Daten mithilfe von eksctl auf einem Amazon EKS Auto Mode-Cluster zu speichern.

Amazon EKS Auto Mode vereinfacht die Clusterverwaltung, indem er Routineaufgaben wie Blockspeicher, Netzwerke, Lastenausgleich und automatische Rechenskalierung automatisiert. Während der Einrichtung übernimmt es die Erstellung von Knoten mit EC2 verwalteten Instances, Anwendungs-Loadbalancern und EBS-Volumes.

Zusammenfassend stellen Sie einen Beispiel-Workload mit den benutzerdefinierten Anmerkungen bereit, die für eine nahtlose Integration mit Services erforderlich sind. AWS

In diesem Tutorial

Mit der folgenden `eksctl`-Cluster-Vorlage erstellen Sie einen Cluster mit EKS Auto Mode für die automatisierte Bereitstellung von Knoten.

- VPC-Konfiguration: Bei Verwendung der folgenden eksctl-Clustervorlage erstellt eksctl automatisch eine IPv4 Virtual Private Cloud (VPC) für den Cluster. Standardmäßig konfiguriert eksctl eine VPC, die alle Netzwerkanforderungen erfüllt und sowohl öffentliche als auch private Endpunkte erstellt.
- Instanzverwaltung: Der automatische Modus von EKS fügt je nach den Anforderungen Ihrer Kubernetes-Anwendungen dynamisch Knoten in Ihrem EKS-Cluster hinzu oder entfernt sie.
- Datenpersistenz: Verwenden Sie die Blockspeicherfunktion von EKS Auto Mode, um die Persistenz von Anwendungsdaten sicherzustellen, auch in Szenarien mit Pod-Neustarts oder -Ausfällen.
- Externer App-Zugriff: Verwenden Sie die Lastausgleichsfunktion von EKS Auto Mode, um einen Application Load Balancer (ALB) dynamisch bereitzustellen.

Voraussetzungen

Stellen Sie zunächst sicher, dass Sie die folgenden Aufgaben durchgeführt haben:

- Einrichtung Ihrer Umgebung für Amazon EKS
- Installation der neuesten Version von eksctl

Konfigurieren des Clusters

In diesem Abschnitt erstellen Sie einen Cluster unter Verwendung von EKS Auto Mode für die dynamische Bereitstellung von Knoten.

Erstellen Sie eine `cluster-config.yaml`-Datei und fügen Sie den folgenden Inhalt ein. Durch eine gültige Region `region-code` ersetzen (z. B.`us-east-1`).

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: web-quickstart
  region: <region-code>

autoModeConfig:
  enabled: true
```

Nun können wir den Cluster erstellen.

Erstellen Sie den EKS-Cluster mit dem`cluster-config.yaml`:

```
eksctl create cluster -f cluster-config.yaml
```

> **Important**
>
> Wenn Sie den Cluster nicht mit eksctl erstellen, müssen Sie die VPC-Subnetze manuell kennzeichnen.

Erstellen IngressClass

Erstellen Sie ein Kubernetes `IngressClass` für EKS Auto Mode. Das IngressClass definiert, wie EKS Auto Mode mit Ingress-Ressourcen umgeht. In diesem Schritt wird die Load-Balancing-Funktion von EKS Auto Mode konfiguriert. Wenn Sie Ingress-Ressourcen für Ihre Anwendungen erstellen, verwendet EKS Auto Mode diese, IngressClass um automatisch Load Balancer bereitzustellen und zu verwalten und Ihre Kubernetes-Anwendungen mit Lastausgleichsdiensten zu integrieren. AWS

Speichern Sie die folgende YAML-Datei unter dem Namen `ingressclass.yaml`:

```
apiVersion: networking.k8s.io/v1
kind: IngressClass
metadata:
  name: alb
  annotations:
    ingressclass.kubernetes.io/is-default-class: "true"
spec:
  controller: eks.amazonaws.com/alb
```

Wenden Sie das auf Ihren Cluster an IngressClass :

```
kubectl apply -f ingressclass.yaml
```

Bereitstellung der Beispielanwendung für das Game 2048

In diesem Abschnitt führen wir Sie durch die Schritte zum Bereitstellen des beliebten „Spiel 2048“ als Beispielanwendung innerhalb des Clusters. Das bereitgestellte Manifest enthält benutzerdefinierte Annotationen für den Application Load Balancer (ALB). Diese Anmerkungen sind in EKS integriert und weisen EKS an, eingehenden HTTP-Verkehr als „mit dem Internet verbunden“ zu behandeln und ihn unter Verwendung des Zieltyps „ip“ an den entsprechenden Dienst im `game-2048` Namespace weiterzuleiten.

> **Note**
>
> Das `docker-2048`-Image im Beispiel ist ein `x86_64`-Container-Image und funktioniert nicht auf anderen Architekturen.

1. Erstellen Sie mit dem `--save-config`-Flag einen Kubernetes-Namespace mit dem Namen `game-2048`.

   ```
   kubectl create namespace game-2048 --save-config
   ```

   Die ausgegebene Antwort sollte wie folgt aussehen:

   ```
   namespace/game-2048 created
   ```

2. Stellen Sie die Beispielanwendung „Spiel 2048“ bereit.

```
kubectl apply -n game-2048 -f https://raw.githubusercontent.com/kubernetes-sigs/aws-
load-balancer-controller/v2.8.0/docs/examples/2048/2048_full.yaml
```

Dieses Manifest richtet eine Kubernetes-Bereitstellung, einen Service und einen Ingress für den `game-2048`-Namespace ein und erstellt die erforderlichen Ressourcen, um die `game-2048`-Anwendung innerhalb des Clusters bereitzustellen und verfügbar zu machen. Es umfasst die Erstellung eines Service mit dem Name `service-2048`, der die Bereitstellung auf Port `80` verfügbar macht, sowie einer Ingress-Ressource mit dem Namen `ingress-2048`, die Routing-Regeln für eingehenden HTTP-Datenverkehr und Annotationen für einen mit dem Internet verbundenen Application Load Balancer (ALB) definiert. Die ausgegebene Antwort sollte wie folgt aussehen:

```
namespace/game-2048 configured
deployment.apps/deployment-2048 created
service/service-2048 created
ingress.networking.k8s.io/ingress-2048 created
```

3. Führen Sie den folgenden Befehl aus, um die Ingress-Ressource für den `game-2048`-Namespace abzurufen.

```
kubectl get ingress -n game-2048
```

Die ausgegebene Antwort sollte wie folgt aussehen:

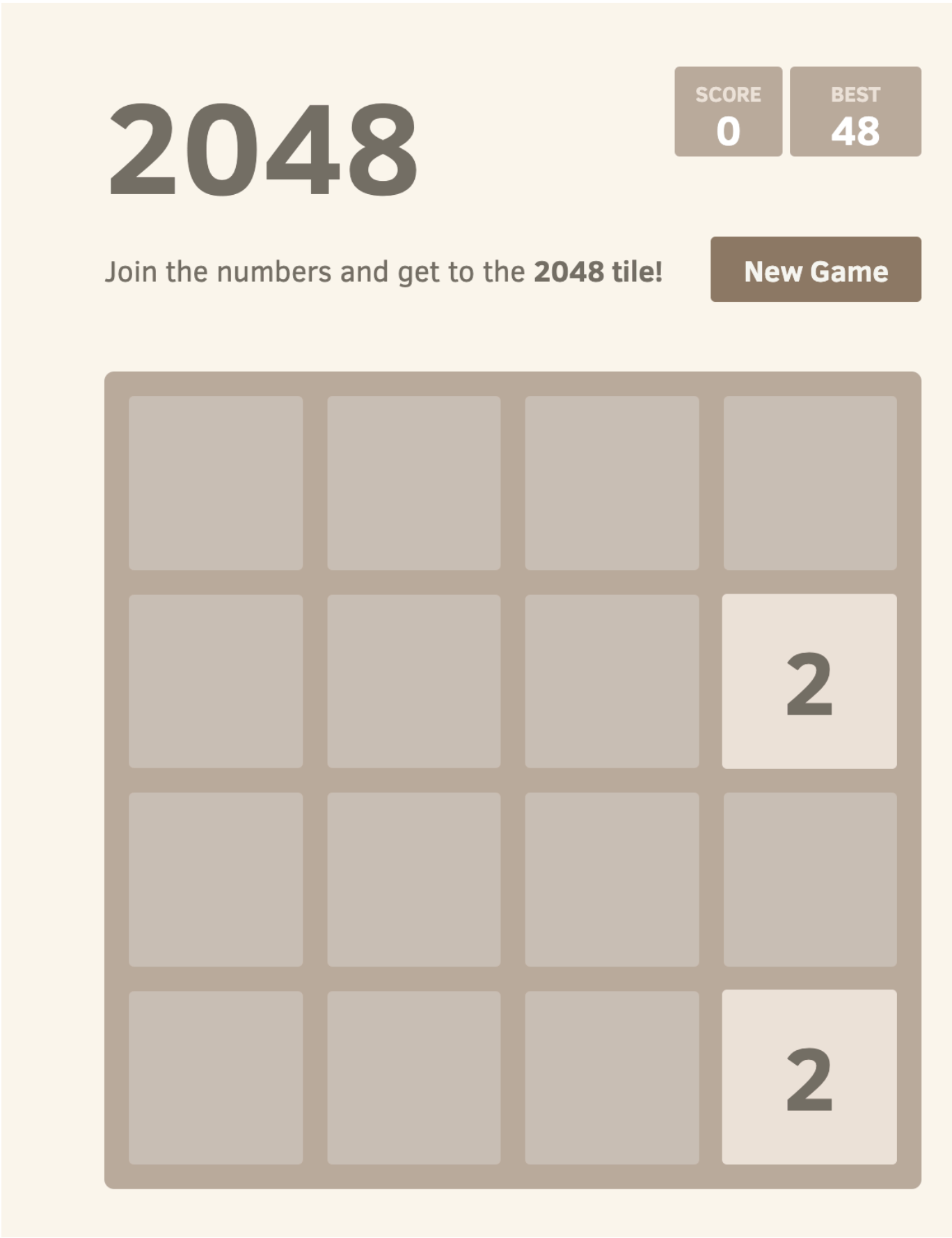
```
NAME           CLASS   HOSTS   ADDRESS
                      PORTS   AGE
ingress-2048   alb     *       k8s-game2048-ingress2-eb379a0f83-378466616.region-
code.elb.amazonaws.com   80      31s
```

Sie müssen einige Minuten warten, bis der Application Load Balancer (ALB) bereitgestellt ist, bevor Sie mit den folgenden Schritten beginnen.

4. Öffnen Sie einen Web-Browser und geben Sie `ADDRESS` aus dem vorherigen Schritt ein, um auf die Web-Anwendung zuzugreifen. Beispiel:

```
k8s-game2048-ingress2-eb379a0f83-378466616.region-code.elb.amazonaws.com
```

Das Spiel 2048 sollte in Ihrem Browser angezeigt werden. Spielen!


2048
SCORE
0
BEST
48
Join the numbers and get to the 2048 tile!
New Game
2
2

Persistente Daten unter Verwendung von Amazon EKS Auto Mode

Nachdem das Spiel 2048 nun in Ihrem Amazon-EKS-Cluster ausgeführt wird, ist es an der Zeit, sicherzustellen, dass Ihre Spieldaten mithilfe der Blockspeicherfunktion von Amazon EKS Auto Mode sicher gespeichert werden.

1. Erstellen Sie eine Datei namens `storage-class.yaml`:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: auto-ebs-sc
  annotations:
    storageclass.kubernetes.io/is-default-class: "true"
provisioner: ebs.csi.eks.amazonaws.com
volumeBindingMode: WaitForFirstConsumer
parameters:
  type: gp3
  encrypted: "true"
```

2. Wenden Sie `StorageClass` an:

```
kubectl apply -f storage-class.yaml
```

3. Erstellen Sie einen Persistent Volume Claim (PVC), um Speicherplatz für Ihre Spieldaten anzufordern. Erstellen Sie eine Datei mit dem Namen `ebs-pvc.yaml` und fügen Sie den folgenden Inhalt hinzu:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: game-data-pvc
  namespace: game-2048
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi
  storageClassName: auto-ebs-sc
```

4. Wenden Sie das PVC auf Ihren Cluster an:

```
kubectl apply -f ebs-pvc.yaml
```

Die ausgegebene Antwort sollte wie folgt aussehen:

```
persistentvolumeclaim/game-data-pvc created
```

5. Nun müssen Sie Ihre 2048-Spielbereitstellung aktualisieren, um diesen PVC für die Speicherung von Daten zu verwenden. Die folgende Bereitstellung ist so konfiguriert, dass sie den PVC für die Speicherung von Spieldaten verwendet. Erstellen Sie eine Datei mit dem Namen `ebs-deployment.yaml` und fügen Sie ihr den folgenden Inhalt hinzu:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  namespace: game-2048
  name: deployment-2048
spec:
  replicas: 3  # Adjust the number of replicas as needed
  selector:
    matchLabels:
      app.kubernetes.io/name: app-2048
  template:
    metadata:
      labels:
        app.kubernetes.io/name: app-2048
    spec:
      containers:
        - name: app-2048
          image: public.ecr.aws/l6m2t8p7/docker-2048:latest
          imagePullPolicy: Always
          ports:
            - containerPort: 80
          volumeMounts:
            - name: game-data
              mountPath: /var/lib/2048
      volumes:
        - name: game-data
          persistentVolumeClaim:
            claimName: game-data-pvc
```

6. Wenden Sie die aktualisierte Bereitstellung an:

```
kubectl apply -f ebs-deployment.yaml
```

Die ausgegebene Antwort sollte wie folgt aussehen:

```
deployment.apps/deployment-2048 configured
```

Mit diesen Schritten ist Ihr Spiel 2048 im Cluster nun so eingerichtet, dass Daten mithilfe der Blockspeicherfunktion von Amazon EKS Auto Mode dauerhaft gespeichert werden. Dadurch wird sichergestellt, dass Ihr Spielfortschritt und Ihre Daten auch bei Pod- oder Knotenausfällen sicher sind.

Wenn Ihnen dieses Tutorial gefallen hat, teilen Sie uns dies mit, damit wir Ihnen weitere anwendungsspezifische Schnellstart-Tutorials wie dieses anbieten können.

Bereinigen

Um future Gebühren zu vermeiden, müssen Sie den zugehörigen CloudFormation Stack manuell löschen, um alle in diesem Handbuch erstellten Ressourcen, einschließlich des VPC-Netzwerks, zu löschen.

Löschen Sie den Stack: CloudFormation

```
eksctl delete cluster -f ./cluster-config.yaml
```

Amazon EKS anhand von Beispielen kennenlernen

Übersicht

Dieses Amazon-EKS-Benutzerhandbuch enthält allgemeine Verfahren zum Erstellen Ihres ersten EKS-Clusters über die Befehlszeile oder AWS-Managementkonsole sowie eine solide Referenz für alle wichtigen Amazon-EKS-Komponenten. Als Amazon-EKS-Cluster-Administrator oder -Entwickler können Sie jedoch ein tieferes Verständnis von Amazon EKS erlangen, indem Sie Lernpfade auf Websites außerhalb dieses Handbuchs befolgen. Diese Websites können Ihnen helfen:

- Richten Sie bestimmte Cluster-Typen ein. Bestimmte Cluster-Typen können auf Ihren Workload-Typen oder Sicherheitsanforderungen basieren. Beispielsweise möchten Sie möglicherweise einen Cluster so optimieren, dass er Batch-, Machine Learning- oder rechenintensive Workloads ausführen kann.
- Verbessern Sie Ihre Cluster. Sie können Ihrem Cluster erweiterte Features hinzufügen, um beispielsweise Beobachtbarkeit, flexiblen Speicher, automatische Skalierung oder spezialisierte Cluster-Netzwerke bereitzustellen.
- Automatisieren Sie Updates. Mithilfe von Features wie GitOps können Sie die automatische Bereitstellung von Cluster-Infrastruktur und Workloads einrichten können, basierend auf Änderungen, die an diesen Komponenten in Ihren Git-Repositorys vorgenommen werden.
- Verwenden Sie erweiterte Tools zur Cluster-Einrichtung. Während `eksctl` eine schnelle Möglichkeit zum Erstellen eines Clusters bietet, gibt es andere Tools, welche die Konfiguration und Aktualisierung komplexerer Cluster erleichtern können. Dazu gehören Tools wie Terraform und CloudFormation.

Um mit Ihrem Amazon-EKS-Lernpfad zu beginnen, empfehlen wir Ihnen, einige der auf dieser Seite beschriebenen Websites zu besuchen. Sollten Sie unterwegs auf Probleme stoßen, stehen Ihnen auch Ressourcen zur Verfügung, die Ihnen bei der Lösung helfen. Im Re:post Knowledge Center können Sie beispielsweise die Support-Datenbank nach Amazon-EKS-bezogenen Support-Problemen durchsuchen. Der Leitfaden zu bewährten Methoden für Amazon EKS bietet Tipps zur optimalen Einrichtung Ihrer Produktions-Cluster.

Workshop zu Amazon EKS

Der Amazon-EKS-Workshop vermittelt grundlegende Kenntnisse über Kubernetes und Container und dient als Lernplattform, auf der Cluster-Administratoren mit den wichtigsten Features von Amazon EKS vertraut gemacht werden. Sie können auf folgende Weise am Amazon-EKS-Workshop teilnehmen:

- Grundlagen zu Amazon EKS: Sehen Sie sich das Video auf der Seite Einführung an, um zu erfahren, wie Amazon-EKS-Kubernetes-Feature in der AWS Cloud implementiert. Wenn Sie ein noch grundlegenderes Verständnis von Kubernetes benötigen, sehen Sie sich das Video Was ist Kubernetes an.
- Amazon-EKS-Einrichtung: Wenn Sie über ein AWS-Konto verfügen, hilft Ihnen der Abschnitt Einrichtung beim Einrichten einer CloudShell-Umgebung zum Erstellen eines Clusters. Es bietet die Wahl zwischen eksctl (einer einfachen Befehlszeile zur Cluster-Erstellung) und Terraform (einem eher Infrastruktur-als-Code-Ansatz zur Cluster-Erstellung) zum Erstellen Ihres Amazon-EKS-Clusters.
- Amazon EKS – Erste Schritte: Probieren Sie einen einfachen Webshop aus dem Abschnitt Beispielanwendung aus. Diesen können Sie in den anderen Übungen verwenden. In diesem Abschnitt erfahren Sie außerdem mehr über das Verpacken von Container-Images und die Verwaltung von Microservices mit Kubernetes-Pods, -Bereitstellungen, -Services, -StatefulSets und -Namespaces. Verwenden Sie anschließend Kustomize, um Änderungen an Kubernetes-Manifesten bereitzustellen.
- Amazon-EKS-Grundlagen: Mithilfe von AWS-Features wie dem AWS Load Balancer Controller zeigt Ihnen der Workshop, wie Sie Ihre Anwendungen für die Öffentlichkeit zugänglich machen. Für die Speicherung zeigt der Workshop, wie Amazon EBS für Blockspeicher, Amazon EFS für Dateisystemspeicher und Amazon FSx für NetApp ONTAP zum Verwalten von ONTAP-Dateisystemen in AWS verwendet werden. Im Bereich Knotenverwaltung unterstützt Sie der Workshop beim Einrichten verwalteter Knotengruppen.
- Erweiterte Features von Amazon EKS: Zu den erweiterten Features, die im Amazon-EKS-Workshop angeboten werden, gehören Übungen zum Einrichten von:
  - Automatische Skalierung: Dazu gehören die automatische Skalierung von Knoten (mit Cluster Autoscaler oder Karpenter) und die automatische Skalierung der Workloads (mit Horizontal Pod Autoscaler und Cluster Proportional Autoscaler).
  - Beobachtbarkeit: Erfahren Sie in einer Reihe von Beobachtbarkeitsübungen mehr über Protokollierung, OpenSearch, Container Insights in Amazon EKS und Kostentransparenz mit Kubecost.

- Sicherheit: Mit dieser Reihe von Sicherheitsübungen können Sie sich mit Secrets Management, Amazon GuardDuty, Pod-Sicherheitsstandards und Kyverno-Richtlinienverwaltung vertraut machen.
- Netzwerke: Lernen Sie in den Netzwerk-Übungen die Netzwerk-Features für Amazon EKS kennen, darunter Amazon VPC CNI (zur Unterstützung von Netzwerk-Plugins) und Amazon VPC Lattice (zur Konfiguration von Clustern über VC- und Benutzerkonten hinweg).
- Automatisierung: In den Übungen zur Automatisierung werden Sie Schritt für Schritt durch GitOps-Methoden zur Verwaltung Ihrer Cluster und Projekte wie AWS Controllers für Kubernetes und Crossplane für die Verwaltung von Amazon-EKS-Steuerebenen geführt.

Amazon-EKS-Tutorials zur praktischen Einrichtung von Clustern

Eine Reihe von Tutorials zur Einrichtung von Amazon-EKS-Clustern auf der Seite der AWS-Community kann Ihnen dabei helfen, spezielle Amazon-EKS-Cluster zu erstellen und diese Cluster auf verschiedene Weise zu verbessern. Die Tutorials sind in drei verschiedene Typen unterteilt:

Entwicklung von Clustern

Diese Tutorials helfen Ihnen bei der Entwicklung von Clustern, die für spezielle Zwecke verwendet werden können. Zu diesen speziellen Zwecken gehört die Möglichkeit, Folgendes auszuführen:

- Global skalierbare Anwendungen auf Basis von IPv6
- Asynchrone Batch-Aufgaben
- Microservices mit hohem Datenverkehr
- Automatische Skalierung mit Karpenter in Fargate
- Finanzielle Workloads
- Von Windows verwaltete Knotengruppen

Cluster-Optimierung

Sobald Sie über einen vorhandenen Cluster verfügen, können Sie diesen so erweitern und optimieren, dass er spezielle Workloads ausführen und die Cluster anderweitig verbessern kann. Diese Tutorials umfassen Möglichkeiten zum:

- Bereitstellen von Speicherlösungen mit EFS CSI
- Bereitstellen von dynamischem Datenbankspeicher mit EBS CSI

- Anwendungen in IPv4-Clustern mit dem AWS Load Balancer Controller bereitstellen
- Anwendungen in IPv6-Clustern mit dem AWS Load Balancer Controller bereitstellen

Optimierung von AWS-Services

Mithilfe dieser Tutorials können Sie Ihre Cluster besser mit AWS-Services integrieren. Diese Tutorials helfen Ihnen unter anderem bei Folgendem:

- Verwalten von DNS-Einträgen für Microservices mit ExternalDNS
- Überwachung von Anwendungen mit CloudWatch
- Verwaltung asynchroner Aufgaben mit SQS und EFS-Speicher
- Verbrauch von AWS-Secrets-Manager-Geheimnissen aus Ihrer Workload
- Einrichtung von mTLS mit Fargate, NGINX und ACM PCA

Amazon-EKS-Beispiele

Das Repository für Amazon-EKS-Beispiele speichert Manifeste zur Verwendung mit Amazon EKS. Diese Manifeste bieten Ihnen die Möglichkeit, verschiedene Arten von Anwendungen in Amazon EKS auszuprobieren oder bestimmte Arten von Amazon-EKS-Clustern zu erstellen. Beispiele enthalten Manifeste für:

- Erstellung eines AWS-Amazon-EKS-Clusters
- Erstellung eines Clusters mit einer vorhandenen IAM-Rolle
- Hinzufügen einer Ubuntu-verwalteten Knoten-Gruppe zu einem Cluster
- Sicherung und Wiederherstellung von Pod-Speicher mit Volume-Snapshots
- Wiederherstellen von EBS-Volumes, die als PVCs mit mehreren Konten eingebunden sind
- Aktivierung des Proxy-Protokolls für NGINX Ingress Controller mit Classic Load Balancer
- Konfiguration der Protokollierung in Fargate für AWS OpenSearch
- Ausführung des Python SDK mit einem Web-Verbund-Identitätsanbieter
- Bereitstellung einer Beispielanwendung in einem NFS-CSI-Controller
- Verwendung von Volume-Snapshots für StatefulSets
- Bereitstellung von Pods über Knoten in verschiedenen Availability Zones

Beachten Sie, dass diese Beispiele ausschließlich zu Lern- und Testzwecken dienen und nicht in der Produktion verwendet werden sollen.

AWS-Tutorials

Auf der Website AWS Tutorials werden einige Amazon-EKS-Tutorials veröffentlicht, aber es gibt auch ein Such-Tool für die Suche nach weiteren Tutorials, die auf anderen AWS-Websites (z. B. der AWS-Community-Website) veröffentlicht wurden. Zu den direkt auf dieser Site veröffentlichten Amazon-EKS-Tutorials gehören:

- Bereitstellung einer Container-Web-App in Amazon EKS
- Kostengünstige Ausführung von Kubernetes-Cluster (Amazon EKS und Spot Instances)
- So optimieren Sie die Kosten für Jenkins-Aufträge in Kubernetes

Workshop für Entwickler

Wenn Sie Software-Entwickler sind und Anwendungen erstellen oder umgestalten möchten, die in Amazon EKS ausgeführt werden sollen, ist der Amazon-EKS-Entwickler-Workshop ein guter Ausgangspunkt. Der Workshop unterstützt Sie nicht nur beim Entwickeln containerisierter Anwendungen, sondern auch beim Bereitstellen dieser Container in einer Container-Registry (ECR) und von dort aus in einem Amazon-EKS-Cluster.

Beginnen Sie mit dem Amazon-EKS-Python-Workshop, um den Prozess der Umgestaltung einer Python-Anwendung durchzugehen, und richten Sie dann Ihre Entwicklungsumgebung ein, um die Bereitstellung der Anwendung vorzubereiten. Gehen Sie die Abschnitte zu Containern, Kubernetes und Amazon EKS durch, um die Ausführung Ihrer containerisierten Anwendungen in diesen Umgebungen vorzubereiten.

Terraform-Workshop

Während `eksctl` ein einfaches Tool zum Erstellen eines Clusters ist, stellt Terraform ein beliebtes Tool zum Erstellen und Verwalten von Amazon-EKS-Clustern für komplexere Infrastructure-as-Code-Typen von Amazon-EKS-Bereitstellungen dar. Im Terraform-Amazon-EKS-Workshop lernen Sie, wie Sie Terraform zur Entwicklung einer AWS-VPC, zum Erstellen von Amazon-EKS-Clustern und zum Hinzufügen optionaler Erweiterungen an Ihrem Cluster verwenden. Insbesondere gibt es einen Abschnitt zum Erstellen eines privaten Amazon-EKS-Clusters

AWS-Amazon-EKS-Training

AWS bietet formelles Training für das Erlernen von Amazon EKS an. Der dreitägige Trainingskurs Ausführung von Containern in Amazon Elastic Kubernetes Service vermittelt folgende Inhalte:

- Grundlagen von Kubernetes und Amazon EKS
- Erstellung von Amazon-EKS-Clustern
- Sicherung von Amazon EKS mit AWS-IAM- und Kubernetes-RBAC-Autorisierung
- GitOps-Automatisierungstools
- Überwachungstools
- Techniken zur Verbesserung von Kosten, Effizienz und Ausfallsicherheit

Erste Schritte mit Amazon EKS

Vergewissern Sie sich, dass Sie für die Verwendung von Amazon EKS bereit sind, bevor Sie sich mit den Anleitungen für die ersten Schritte beschäftigen. Weitere Informationen finden Sie unter Einrichten.

Es gibt zwei Handbücher „Erste Schritte“ für das Erstellen eines neuen Kubernetes-Clusters mit Knoten in Amazon EKS:

- Erste Schritte mit Amazon EKS – eksctl – Dieses Handbuch für den Einstieg unterstützt Sie bei der Installation aller erforderlichen Ressourcen für den Einstieg in Amazon EKS mithilfe von `eksctl`, einem einfachen Befehlszeilenprogramm zum Erstellen und Verwalten von Kubernetes-Clustern auf Amazon EKS. Am Ende des Tutorials haben Sie einen laufenden Amazon-EKS-Cluster, auf dem Sie Anwendungen bereitstellen können. Dies ist die schnellste und einfachste Möglichkeit zum Einstieg in Amazon EKS.
- Erste Schritte mit Amazon EKS – AWS-Managementkonsole und AWS-CLI – Diese Anleitung für den Einstieg hilft Ihnen, alle erforderlichen Ressourcen zu erstellen, um mit Amazon EKS unter Verwendung von AWS-Managementkonsole und AWS-CLI zu beginnen. Am Ende des Tutorials haben Sie einen laufenden Amazon-EKS-Cluster, auf dem Sie Anwendungen bereitstellen können. In diesem Handbuch erstellen Sie manuell jede Ressource, die für einen Amazon-EKS-Cluster erforderlich ist. Die Verfahren geben Ihnen einen Überblick darüber, wie jede Ressource erstellt wird und wie die Ressourcen miteinander interagieren.

Wir bieten außerdem die folgenden Referenzen:

- Eine Sammlung praktischer Tutorials finden Sie unter EKS-Cluster-Einrichtung in der AWS-Community.
- Code-Beispiele finden Sie unter Code-Beispiele für Amazon EKS unter Verwendung von AWS-SDKs.

Erste Schritte mit Amazon EKS – EKS Auto Mode

Wie bei anderen EKS-Einführungserfahrungen wird auch beim Erstellen Ihres ersten Clusters mit dem EKS Auto Mode die Verwaltung des Clusters selbst an AWS delegiert. EKS Auto Mode erweitert jedoch die EKS-Automatisierung, indem es die Verantwortung für viele wichtige Services übernimmt, die für die Einrichtung der Workload-Infrastruktur (Knoten, Netzwerke und verschiedene Services)

erforderlich sind. Dadurch wird die Verwaltung von Knoten und die Hochskalierung zur Erfüllung der Workload-Anforderungen vereinfacht.

Wählen Sie eine der folgenden Möglichkeiten, um einen Cluster mit dem EKS Auto Mode zu erstellen:

- the section called " AWS CLI": Verwenden Sie die `aws`-Befehlszeilenschnittstelle, um einen Cluster zu erstellen.
- the section called "Management-Konsole": Verwenden Sie die AWS-Managementkonsole zum Erstellen eines Clusters.
- the section called "eksctl CLI": Verwenden Sie die `eksctl`-Befehlszeilenschnittstelle, um einen Cluster zu erstellen.

Wenn Sie verschiedene Ansätze zum Erstellen Ihres ersten EKS-Clusters vergleichen, sollten Sie wissen, dass AWS im EKS Auto Mode zusätzliche Aufgaben der Cluster-Verwaltung übernimmt, darunter das Einrichten von Komponenten für:

- Starten und skalieren Sie Knoten entsprechend der steigenden und sinkenden Workload.
- Aktualisieren Sie regelmäßig den Cluster selbst (Steuerebene), die Knotenbetriebssysteme und die auf den Knoten ausgeführten Services.
- Wählen Sie Standardeinstellungen, die beispielsweise die Größe und Geschwindigkeit des Knotenspeichers und die Pod-Netzwerkkonfiguration bestimmen.

Weitere Informationen zu den Vorteilen von EKS-Auto-Mode-Clustern finden Sie unter EKS Auto Mode.

Erste Schritte mit Amazon EKS – **`eksctl`**

> **Note**
>
> Dieses Thema behandelt die ersten Schritte ohne EKS Auto Mode.
> EKS Auto Mode automatisiert Routineaufgaben für Cluster-Computing, -Speicherung und -Netzwerke. Informationen zu den ersten Schritten mit Amazon EKS Auto Mode.

Dieses Handbuch hilft Ihnen beim Erstellen aller erforderlichen Ressourcen für die ersten Schritte mit Amazon Elastic Kubernetes Service (Amazon EKS) mithilfe von `eksctl`, einem einfachen Befehlszeilendienstprogramm zum Erstellen und Verwalten von Kubernetes-Clustern auf Amazon EKS. Am Ende dieses Tutorials haben Sie einen laufenden Amazon-EKS-Cluster, auf dem Sie Anwendungen bereitstellen können.

Die Verfahren in diesem Handbuch erstellen automatisch mehrere Ressourcen für Sie, die Sie manuell erstellen müssen, wenn Sie Ihren Cluster mit dem AWS-Managementkonsole. Wenn Sie die meisten Ressourcen lieber manuell erstellen möchten, um besser zu verstehen, wie sie miteinander interagieren, verwenden Sie das AWS-Managementkonsole, um Ihren Cluster zu erstellen und zu berechnen. Weitere Informationen finden Sie unter the section called “Cluster erstellen (Konsole und CLI)”.

Voraussetzungen

Bevor Sie mit diesem Tutorial beginnen, müssen Sie die AWS-CLI-, kubectl- und eksctl-Tools wie unter Einrichtung zur Verwendung von Amazon EKS beschrieben installieren und konfigurieren.

Schritt 1: Erstellen Sie Ihre Amazon-EKS-Cluster und -Knoten

> **Important**
>
> Für einen möglichst einfachen und schnellen Einstieg enthält dieses Thema Schritte zum Erstellen eines Clusters und von Knoten mit Standardeinstellungen. Bevor Sie einen Cluster und Knoten für den Produktionseinsatz erstellen, empfehlen wir Ihnen, sich mit allen Einstellungen vertraut zu machen und einen Cluster und Knoten mit den Einstellungen bereitzustellen, die Ihren Anforderungen entsprechen. Weitere Informationen erhalten Sie unter the section called “Erstellen eines Clusters” und Verwaltung von Rechenleistung. Einige Einstellungen können nur aktiviert werden, wenn Sie Ihren Cluster und Ihre Knoten erstellen.

Sie können einen Cluster mit einem der folgenden Knotentypen erstellen. Weitere Informationen zu den einzelnen Typen finden Sie unter Verwaltung von Rechenleistung. Nachdem Ihr Cluster bereitgestellt wurde, können Sie andere Knotentypen hinzufügen.

- Fargate – Linux – Wählen Sie diesen Knotentyp, wenn Sie Linux-Anwendungen in the section called “ AWS-Fargate” ausführen möchten. Fargate ist eine Serverless-Computing-Engine, mit der Sie Kubernetes Pods bereitstellen können, ohne Amazon-EC2-Instances zu verwalten.

- Verwaltete Knoten – Linux – Wählen Sie diesen Knotentyp aus, wenn Sie Amazon-Linux-Anwendungen auf Amazon-EC2-Instances ausführen möchten. Obwohl in diesem Handbuch nicht behandelt, können Sie Ihrem Cluster auch selbstverwaltete Windows- und Bottlerocket-Knoten hinzufügen.

Erstellen Sie Ihren Amazon-EKS-Cluster mit dem folgenden Befehl. Sie können *my-cluster* durch Ihren eigenen Wert ersetzen. Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS-Region und des AWS-Kontos eindeutig sein, in der/dem Sie den Cluster erstellen. Ersetzen Sie *region-code* mit jeder AWS-Region, die von Amazon EKS unterstützt wird. Eine Liste der AWS-Regionen finden Sie unter Endpunkte von Amazon EKS und Kontingente im Allgemeinem AWS-Handbuch.

Example

Fargate - Linux

```
eksctl create cluster --name my-cluster --region region-code --fargate
```

Managed nodes - Linux

```
eksctl create cluster --name my-cluster --region region-code
```

Die Clustererstellung dauert mehrere Minuten. Während der Erstellung werden mehrere Ausgabezeilen angezeigt. Die letzte Ausgabezeile ähnelt der folgenden Beispielzeile.

```
[...]
[#]  EKS cluster "my-cluster" in "region-code" region is ready
```

`eksctl` hat eine `kubectl`-config-Datei in `~/.kube/config` erstellt oder hat die Konfiguration des neuen Clusters innerhalb einer vorhandenen config-Datei in `~/.kube/config` auf Ihrem Rechner hinzugefügt.

Zeigen Sie nach Abschluss der Cluster-Erstellung den AWS-CloudFormation-Stack mit dem Namen `eksctl-`*my-cluster*`-cluster` in der AWS-CloudFormation-Konsole an, um alle erstellten Ressourcen anzuzeigen.

Schritt 2: Anzeigen von Kubernetes-Ressourcen

1. Zeigen Sie Ihre Cluster-Knoten an.

```
kubectl get nodes -o wide
```

Eine Beispielausgabe sieht wie folgt aus.

Example

Fargate - Linux

```
NAME                                                  STATUS   ROLES    AGE
 VERSION              INTERNAL-IP   EXTERNAL-IP   OS-IMAGE         KERNEL-VERSION
                CONTAINER-RUNTIME
fargate-ip-192-0-2-0.region-code.compute.internal   Ready    <none>
 8m3s    v1.2.3-eks-1234567   192.0.2.0     <none>        Amazon Linux 2
 1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
fargate-ip-192-0-2-1.region-code.compute.internal   Ready    <none>
 7m30s   v1.2.3-eks-1234567   192-0-2-1     <none>        Amazon Linux 2
 1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
```

Managed nodes - Linux

```
NAME                                        STATUS   ROLES    AGE    VERSION
        INTERNAL-IP   EXTERNAL-IP   OS-IMAGE         KERNEL-VERSION
   CONTAINER-RUNTIME
ip-192-0-2-0.region-code.compute.internal   Ready    <none>   6m7s
  v1.2.3-eks-1234567   192.0.2.0     192.0.2.2     Amazon Linux 2
 1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
ip-192-0-2-1.region-code.compute.internal   Ready    <none>   6m4s
  v1.2.3-eks-1234567   192.0.2.1     192.0.2.3     Amazon Linux 2
 1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
```

Weitere Informationen dazu, was in der Ausgabe angezeigt wird, finden Sie unter the section called “Zugriff auf Cluster-Ressourcen”.

2. Zeigen Sie die Workloads an, die auf Ihrem Cluster ausgeführt werden.

```
kubectl get pods -A -o wide
```

Eine Beispielausgabe sieht wie folgt aus.

Example

Fargate - Linux

```
NAMESPACE     NAME                          READY   STATUS    RESTARTS   AGE   IP
          NODE                                                NOMINATED NODE
 READINESS GATES
kube-system   coredns-1234567890-abcde   1/1     Running   0          18m
 192.0.2.0   fargate-ip-192-0-2-0.region-code.compute.internal   <none>
 <none>
kube-system   coredns-1234567890-12345   1/1     Running   0          18m
 192.0.2.1   fargate-ip-192-0-2-1.region-code.compute.internal   <none>
 <none>
```

Managed nodes - Linux

```
NAMESPACE     NAME                          READY   STATUS    RESTARTS   AGE     IP
          NODE                                          NOMINATED NODE   READINESS
 GATES
kube-system   aws-node-12345             1/1     Running   0          7m43s
 192.0.2.1   ip-192-0-2-1.region-code.compute.internal   <none>           <none>
kube-system   aws-node-67890             1/1     Running   0          7m46s
 192.0.2.0   ip-192-0-2-0.region-code.compute.internal   <none>           <none>
kube-system   coredns-1234567890-abcde   1/1     Running   0          14m
 192.0.2.3   ip-192-0-2-3.region-code.compute.internal   <none>           <none>
kube-system   coredns-1234567890-12345   1/1     Running   0          14m
 192.0.2.4   ip-192-0-2-4.region-code.compute.internal   <none>           <none>
kube-system   kube-proxy-12345           1/1     Running   0          7m46s
 192.0.2.0   ip-192-0-2-0.region-code.compute.internal   <none>           <none>
kube-system   kube-proxy-67890           1/1     Running   0          7m43s
 192.0.2.1   ip-192-0-2-1.region-code.compute.internal   <none>           <none>
```

Weitere Informationen dazu, was in der Ausgabe angezeigt wird, finden Sie unter the section called “Zugriff auf Cluster-Ressourcen”.

Schritt 3: Löschen Ihrer Cluster und Knoten

Nachdem Sie mit dem Cluster und den Knoten fertig sind, die Sie für dieses Tutorial erstellt haben, sollten Sie den Cluster und die Knoten mit dem folgenden Befehl löschen. Wenn Sie vor dem Bereinigen mehr mit diesem Cluster tun möchten, lesen Sie the section called “Nächste Schritte”.

```
eksctl delete cluster --name my-cluster --region region-code
```

Nächste Schritte

Die folgenden Dokumentationsthemen helfen Ihnen bei der Erweiterung der Funktionalität Ihres Clusters.

- Stellen Sie eine Beispielanwendung für Ihr Cluster bereit.
- Der IAM-Prinzipal, der den Cluster erstellt hat, ist der einzige Prinzipal, der Aufrufe an den Kubernetes-API-Server mit `kubectl` oder AWS-Managementkonsole tätigen kann. Wenn Sie möchten, dass andere IAM-Prinzipale Zugriff auf Ihren Cluster haben, müssen Sie sie hinzufügen. Weitere Informationen erhalten Sie unter the section called “Kubernetes-API-Zugriff” und the section called “Erforderliche Berechtigungen”.
- Bevor Sie einen Cluster für die Produktion bereitstellen, empfehlen wir Ihnen, sich mit allen Einstellungen für Cluster und Knoten vertraut zu machen. Einige Einstellungen, (z. B. das Aktivieren des SSH-Zugriffs auf Amazon-EC2-Knoten) müssen vorgenommen werden, wenn der Cluster erstellt wird.
- Um die Sicherheit für Ihren Cluster zu erhöhen, konfigurieren Sie das Amazon VPC Container Networking Interface-Plugin für die Verwendung von IAM-Rollen für Servicekonten.

Erste Schritte mit Amazon EKS — AWS-Managementkonsole und AWS CLI

> **Note**
>
> Dieses Thema behandelt die ersten Schritte ohne EKS Auto Mode. Es verwendet verwaltete Knotengruppen zum Bereitstellen von Knoten.

EKS Auto Mode automatisiert Routineaufgaben für Cluster-Computing, -Speicherung und -Netzwerke. Informationen zu den ersten Schritten mit Amazon EKS Auto Mode. EKS Auto Mode ist die bevorzugte Methode zum Bereitstellen von Knoten.

Dieses Handbuch hilft Ihnen dabei, alle erforderlichen Ressourcen für die ersten Schritte mit Amazon Elastic Kubernetes Service (Amazon EKS) mithilfe der AWS-Managementkonsole und der AWS CLI zu erstellen. In diesem Leitfaden erstellen Sie jede Ressource manuell. Am Ende dieses Tutorials haben Sie einen laufenden Amazon-EKS-Cluster, auf dem Sie Anwendungen bereitstellen können.

Die Verfahren in diesem Handbuch geben Ihnen vollständige Einsichten in die Erstellung der einzelnen Ressourcen und in die Interaktionen der Ressourcen. Wenn die Mehrzahl der Ressourcen automatisch für Sie erstellt werden soll, verwenden Sie die `eksctl`-CLI, um Ihre Cluster- und Knoten zu erstellen. Weitere Informationen finden Sie unter the section called “Cluster erstellen (eksctl)”.

Voraussetzungen

Bevor Sie mit diesem Tutorial beginnen, müssen Sie die folgenden Werkzeuge und Ressourcen installieren und konfigurieren, die Sie zum Erstellen und Verwalten eines Amazon-EKS-Clusters benötigen.

- AWS CLI — Ein Befehlszeilentool für die Arbeit mit AWS Services, einschließlich Amazon EKS. Weitere Informationen finden Sie unter Installation im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Wir empfehlen, dass Sie die AWS CLI nach der Installation ebenfalls konfigurieren. Weitere Informationen finden Sie unter Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Beachten Sie, dass AWS CLI v2 erforderlich ist, um die auf dieser Seite gezeigte Option update-kubeconfig zu verwenden.
- **`kubectl`** – Ein Befehlszeilentool für die Arbeit mit Kubernetes-Clustern. Weitere Informationen finden Sie unter the section called “kubectl und eksctl einrichten”.
- Erforderliche IAM-Berechtigungen — Der von Ihnen verwendete IAM-Sicherheitsprinzipal muss über Berechtigungen verfügen, um mit Amazon EKS-IAM-Rollen, serviceverknüpften Rollen AWS CloudFormation, einer VPC und verwandten Ressourcen zu arbeiten. Weitere Informationen finden Sie unter Aktionen und Verwendung von serviceverknüpften Rollen im IAM-Benutzerhandbuch. Sie müssen alle Schritte in diesem Handbuch als derselbe Benutzer ausführen. Führen Sie den folgenden Befehl aus, um den aktuellen Benutzer zu überprüfen:

```
aws sts get-caller-identity
```

Wir empfehlen, dass Sie die Schritte in diesem Thema in einer Bash-Shell ausführen. Wenn Sie keine Bash-Shell verwenden, erfordern einige Skript-Befehle wie Zeilenfortsetzungszeichen und die Art und Weise, wie Variablen gesetzt und verwendet werden, eine Anpassung für Ihre Shell. Darüber hinaus können die Zitier- und Escape-Regeln für Ihre Shell unterschiedlich sein. Weitere Informationen finden Sie unter Verwenden von Anführungszeichen mit Zeichenfolgen in der AWS CLI im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle.

Schritt 1: Erstellen Sie Ihre Amazon-EKS-Cluster

> **Important**
>
> Für einen möglichst einfachen und schnellen Einstieg enthält dieses Thema Schritte zum Erstellen eines Clusters mit Standardeinstellungen. Bevor Sie einen Cluster für den Produktionseinsatz erstellen, empfehlen wir Ihnen, sich mit allen Einstellungen vertraut zu machen und einen Cluster mit den Einstellungen bereitzustellen, die Ihren Anforderungen entsprechen. Weitere Informationen finden Sie unter the section called “Erstellen eines Clusters”. Einige Einstellungen können nur aktiviert werden, wenn Sie Ihren Cluster erstellen.

1. Erstellen Sie eine Amazon VPC mit öffentlichen und privaten Subnetzen, die die Amazon-EKS-Anforderungen erfüllt. *region-code*Ersetzen Sie durch eine beliebige AWS Region, die von Amazon EKS unterstützt wird. Eine Liste der AWS Regionen finden Sie unter Amazon EKS-Endpunkte und Kontingente im AWS Allgemeinen Referenzhandbuch. Sie können *my-eks-vpc-stack* mit einem beliebigen Namen ersetzen, den Sie wählen.

```
aws cloudformation create-stack \
  --region region-code \
  --stack-name my-eks-vpc-stack \
  --template-url https://s3.us-west-2.amazonaws.com/amazon-eks/
cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml
```

> **Tip**
>
> Eine Liste aller Ressourcen, die der vorherige Befehl erstellt hat, finden Sie in der AWS CloudFormation Konsole unter https://console.aws.amazon.com/cloudformation/. Wählen Sie das *my-eks-vpc-stack*-Stack, und wählen Sie dann die Registerkarte Ressourcen.

2. Erstellen Sie eine Cluster-IAM-Rolle und fügen Sie ihr die erforderliche von Amazon EKS IAM verwaltete Richtlinie hinzu. Von Amazon EKS verwaltete Kubernetes-Cluster rufen in Ihrem Namen andere AWS Services auf, um die Ressourcen zu verwalten, die Sie mit dem Service verwenden.

   a. Kopieren Sie den folgenden Inhalt in eine Datei namens *eks-cluster-role-trust-policy.json*.

   ```
   {
     "Version":"2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
           "Service": "eks.amazonaws.com"
         },
         "Action": "sts:AssumeRole"
       }
     ]
   }
   ```

   b. Erstellen Sie die -Rolle.

   ```
   aws iam create-role \
     --role-name myAmazonEKSClusterRole \
     --assume-role-policy-document file://"eks-cluster-role-trust-policy.json"
   ```

   c. Hängen Sie die erforderliche von Amazon EKS verwaltete IAM-Richtlinie an die Rolle an.

   ```
   aws iam attach-role-policy \
     --policy-arn arn:aws: iam::aws:policy/AmazonEKSClusterPolicy \
     --role-name myAmazonEKSClusterRole
   ```

3. Öffnen Sie die Amazon EKS-Konsole unter https://console.aws.amazon.com/eks/home#/clusters.

   Stellen Sie sicher, dass die AWS Region, die oben rechts auf Ihrer Konsole angezeigt wird, die AWS Region ist, in der Sie Ihren Cluster erstellen möchten. Ist dies nicht der Fall, wählen Sie das Drop-down-Menü neben dem Namen der AWS Region aus und wählen Sie die AWS Region aus, die Sie verwenden möchten.

4. Wählen Sie Cluster erstellen. Wenn diese Option nicht angezeigt wird, wählen Sie zunächst im linken Navigationsbereich Cluster aus.

5. Führen Sie auf der Seite Configure cluster (Cluster konfigurieren) die folgenden Schritte aus:

a. Wählen Sie Benutzerdefinierte Konfiguration und deaktivieren Sie EKS Auto Mode verwenden. (Wenn Sie einen EKS-Auto-Mode-Cluster bevorzugen, beachten Sie stattdessen the section called “Management-Konsole”.)

b. Geben Sie einen Namen für Ihren Cluster ein, z. B. *my-cluster*. Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein.

c. Wählen Sie für Cluster Service Role die Option*myAmazonEKSClusterRole*.

d. Belassen Sie die restlichen Einstellungen auf ihren Standardwerten und wählen Sie Weiter aus.

6. Gehen Sie auf der Seite Netzwerk angeben wie folgt vor:

a. Wählen Sie in der Dropdown-Liste VPC (VPC) die ID der VPC aus, die Sie in einem vorherigen Schritt erstellt haben. Es ist so etwas wie** | my-eks-vpc-stack-VPC*.

b. Wählen Sie die im vorherigen Schritt erstellten Subnetze aus der Dropdown-Liste Subnetze aus. Die Subnetze werden so etwas wie ** | my-eks-vpc-stack-** sein.

c. Wählen Sie die im vorherigen Schritt erstellte Sicherheitsgruppe aus der Dropdown-Liste Zusätzliche Sicherheitsgruppen aus. Es ist so etwas wie** | my-eks-vpc-stack-ControlPlaneSecurityGroup-**.

d. Belassen Sie die restlichen Einstellungen auf ihren Standardwerten und wählen Sie Weiter aus.

7. Wählen Sie auf der Seite Beobachtbarkeit konfigurieren die Option Weiter aus.

8. Wählen Sie auf der Seite Add-Ons auswählen die Option Weiter aus.

   Weitere Informationen zu Add-Ons finden Sie unter the section called “Amazon-EKS-Add-ons”.

9. Wählen Sie auf der Seite Konfigurieren ausgewählter Add-Ons-Einstellungen die Option Weiter aus.

10. Wählen Sie auf der Seite Überprüfen und erstellen die Option Erstellen aus.

    Rechts neben dem Cluster-Namen lautet der Cluster-Status für einige Minuten Erstellen, bis der Cluster-Bereitstellungsprozess abgeschlossen ist. Fahren Sie nicht mit dem nächsten Schritt fort, bis der Status Aktiv lautet.

> **Note**
>
> Sie erhalten möglicherweise eine Fehlermeldung, dass eine der Availability Zones in Ihrer Anfrage nicht über genügend Kapazität zum Erstellen eines Amazon-EKS-Clusters verfügt.

Wenn dies der Fall ist, enthält die Fehlerausgabe die Availability Zones, die einen neuen Cluster unterstützen können. Versuchen Sie, Ihren Cluster mit mindestens zwei Subnetzen erneut zu erstellen, die sich in den unterstützten Availability Zones für Ihr Konto befinden. Weitere Informationen finden Sie unter the section called “Unzureichende Kapazität”.

Schritt 2: Konfigurieren Sie Ihren Computer für die Kommunikation mit Ihrem Cluster

In diesem Abschnitt erstellen Sie eine `kubeconfig`-Datei für Ihren Cluster. Die Einstellungen in dieser Datei ermöglichen der `kubectl`-CLI die Kommunikation mit Ihrem Cluster.

Bevor Sie fortfahren, vergewissern Sie sich, dass die Erstellung Ihres Clusters in Schritt 1 erfolgreich abgeschlossen wurde.

1. Erstellen oder aktualisieren Sie eine `kubeconfig`-Datei für Ihren Cluster. *`region-code`*Ersetzen Sie es durch die AWS Region, in der Sie Ihren Cluster erstellt haben. Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters.

   ```
   aws eks update-kubeconfig --region region-code --name my-cluster
   ```

   Standardmäßig wird die `config`-Datei in `~/.kube` erstellt oder die Konfiguration des neuen Clusters wird einer vorhandenen `config`-Datei in `~/.kube` hinzugefügt.

2. Testen Sie Ihre Konfiguration.

   ```
   kubectl get svc
   ```

   Note

   Wenn Sie Autorisierungs- oder Ressourcenfehler erhalten, finden Sie weitere Informationen unter the section called “Nicht autorisiert oder Zugriff verweigert (`kubectl`)” im Thema zur Fehlerbehebung.

   Eine Beispielausgabe sieht wie folgt aus.

```
NAME             TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
svc/kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   1m
```

Schritt 3: Erstellen von Knoten

> **Important**
>
> Um den Einstieg so einfach und schnell wie möglich zu gestalten, enthält dieses Thema Schritte zum Erstellen von Knoten mit überwiegend Standardeinstellungen. Bevor Sie Knoten für den Produktionseinsatz erstellen, empfehlen wir Ihnen, sich mit allen Einstellungen vertraut zu machen und Knoten mit den Einstellungen bereitzustellen, die Ihren Anforderungen entsprechen. Weitere Informationen finden Sie unter Verwaltung von Rechenleistung. Einige Einstellungen können nur aktiviert werden, wenn Sie Ihre Knoten erstellen.

Mit diesem Verfahren konfigurieren Sie Ihren Cluster für die Verwendung von verwalteten Knotengruppen zum Erstellen von Knoten und legen dabei die Subnetze und die IAM-Rolle für Knoten fest, die Sie in den vorherigen Schritten erstellt haben. Damit können Sie Amazon Linux-Anwendungen auf EC2 Amazon-Instances ausführen.

Weitere Informationen zu den verschiedenen Möglichkeiten der Konfiguration von Knoten in EKS finden Sie unter Verwaltung von Rechenleistung. Nachdem Ihr Cluster bereitgestellt wurde, können Sie andere Knotentypen hinzufügen. Obwohl in diesem Handbuch nicht behandelt, können Sie Ihrem Cluster auch selbstverwaltete Windows- und Bottlerocket-Knoten hinzufügen.

Um Ihre verwaltete EC2 Linux-Knotengruppe zu erstellen

1. Erstellen Sie eine Knoten-IAM-Rolle und fügen Sie ihr die erforderliche von Amazon EKS IAM verwaltete Richtlinie hinzu. Der Amazon EKS Node `kubelet` Daemon ruft in Ihrem Namen AWS APIs an. Knoten erhalten über ein IAM-Instance-Profil und zugehörige Richtlinien Berechtigungen für diese API-Aufrufe.
   a. Kopieren Sie den folgenden Inhalt in eine Datei namens `node-role-trust-policy.json`.

```
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

b. Erstellen Sie die Knoten-IAM-Rolle.

```
aws iam create-role \
  --role-name myAmazonEKSNodeRole \
  --assume-role-policy-document file://"node-role-trust-policy.json"
```

c. Hängen Sie die erforderlichen verwalteten IAM-Richtlinien an die Rolle an.

```
aws iam attach-role-policy \
  --policy-arn arn:aws: iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name myAmazonEKSNodeRole
aws iam attach-role-policy \
  --policy-arn arn:aws: iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name myAmazonEKSNodeRole
aws iam attach-role-policy \
  --policy-arn arn:aws: iam::aws:policy/AmazonEKS_CNI_Policy \
  --role-name myAmazonEKSNodeRole
```

d. Öffnen Sie die Amazon EKS-Konsole unter https://console.aws.amazon.com/eks/home#/clusters.

e. Wählen Sie den Namen des Clusters aus, den Sie in Schritt 1: Amazon EKS-Cluster erstellen erstellt haben, z. *my-cluster* B.

f. Gehen Sie auf der ***my-cluster***Seite wie folgt vor:

g. Wählen Sie die Registerkarte Compute (Datenverarbeitung) aus.

h. Wählen Sie Add Node Group (Knotengruppe hinzufügen) aus.

2. Führen Sie auf der Seite Configure Node Group (Knotengruppe konfigurieren) die folgenden Schritte aus:

a. Für Name geben Sie einen eindeutigen Namen für die verwaltete Knotengruppe ein, wie z. B. *my-nodegroup*. Der Name der Knotengruppe darf nicht länger als 63 Zeichen sein. Er muss

mit einem Buchstaben oder einer Ziffer beginnen, kann danach aber auch Bindestriche und Unterstriche enthalten.

b. Wählen Sie als Node IAM-Rollenname die *myAmazonEKSNodeRole* Rolle aus, die Sie in einem vorherigen Schritt erstellt haben. Wir empfehlen, dass jede Knotengruppe ihre eigene eindeutige IAM-Rolle verwendet.

c. Wählen Sie Weiter aus.

3. Akzeptieren Sie auf der Seite Computing- und Skalierungskonfiguration festlegen die Standardwerte und wählen Sie Weiter aus.
4. Akzeptieren Sie auf der Seite Netzwerk angeben die Standardwerte und wählen Sie Weiter aus.
5. Überprüfen Sie auf der Seite Review and create (Überprüfen und Erstellen) die Konfiguration der verwalteten Knoten, und wählen Sie Create (Erstellen).
6. Nach einigen Minuten ändert sich der Status im Abschnitt Konfiguration der Knotengruppe von Wird erstellt auf Aktiv. Fahren Sie nicht mit dem nächsten Schritt fort, bis der Status Aktiv lautet.

Schritt 4: Ressourcen anzeigen

Sie können Ihre Knoten und Kubernetes-Workloads anzeigen.

1. Wählen Sie im linken Navigationsbereich Cluster aus. Wählen Sie in der Liste der Cluster den Namen des Clusters aus, den Sie erstellt haben, z. B. *my-cluster*
2. Wählen Sie auf der ***my-cluster***Seite Folgendes aus:

a. Registerkarte Compute (Datenverarbeitung) – Es wird eine Liste der Knoten angezeigt, die für den Cluster bereitgestellt wurden. Sie können den Namen eines Knotens auswählen, um weitere Informationen darüber anzuzeigen.

b. Registerkarte Resourcen – Es werden alle Kubernetes-Ressourcen angezeigt, die standardmäßig in einem Amazon-EKS-Cluster bereitgestellt werden. Wählen Sie einen Ressourcentyp in der Konsole aus, um mehr über ihn zu erfahren.

Schritt 5: Löschen von Ressourcen

Nachdem Sie mit dem Cluster und den Knoten fertig sind, die Sie für dieses Tutorial erstellt haben, sollten Sie die Ressourcen, die Sie erstellt haben, löschen. Wenn Sie vor dem Löschen der Ressourcen mehr mit diesem Cluster tun möchten, lesen Sie the section called “Nächste Schritte”.

1. Löschen Sie alle von Ihnen erstellten Knotengruppen-Profile.

a. Öffnen Sie die Amazon EKS-Konsole unter https://console.aws.amazon.com/eks/home#/clusters.

b. Wählen Sie im linken Navigationsbereich Cluster aus. Wählen Sie in der Liste der Cluster *my-cluster*.

c. Wählen Sie die Registerkarte Compute (Datenverarbeitung) aus.

d. Wenn Sie eine Knotengruppe erstellt haben, wählen Sie die ***my-nodegroup*** Knotengruppe und dann Löschen aus. Geben Sie *my-nodegroup* ein und klicken Sie auf Delete (Löschen).

e. Fahren Sie nicht fort, bis die Knotengruppenprofile gelöscht sind.

2. Löschen Sie den -Cluster.

a. Wählen Sie im linken Navigationsbereich die Option Cluster aus. Wählen Sie in der Liste der Cluster *my-cluster*.

b. Wählen Sie Delete cluster (Cluster löschen) aus.

c. Geben Sie *my-cluster* ein und wählen Sie Delete (Löschen). Fahren Sie nicht fort, bis der Cluster gelöscht wurde.

3. Löschen Sie den AWS CloudFormation VPC-Stack, den Sie erstellt haben.

a. Öffnen Sie die CloudFormation Konsole unter. https://console.aws.amazon.com/cloudformation/

b. Wählen Sie den *my-eks-vpc-stack*-Stack und anschließend Delete Löschen aus.

c. Wählen Sie im *my-eks-vpc-stack* Bestätigungsdialogfeld für das Löschen die Option Stapel löschen aus.

4. Löschen Sie die IAM-Rollen, die Sie erstellt haben.

a. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

b. Wählen Sie im linken Navigationsbereich Roles aus.

c. Wählen Sie jede Rolle, die Sie erstellt haben ***myAmazonEKSClusterRole***, aus der Liste aus (sowie*myAmazonEKSNodeRole*) aus. Klicken Sie auf Delete (Löschen), geben Sie den angeforderten Bestätigungstext ein und wählen Sie Delete (Löschen) aus.

Nächste Schritte

Die folgenden Dokumentationsthemen helfen Ihnen bei der Erweiterung der Funktionalität Ihres Clusters.

- Der IAM-Prinzipal, der den Cluster erstellt hat, ist der einzige Prinzipal, der Aufrufe an den Kubernetes-API-Server mit `kubectl` oder AWS-Managementkonsole tätigen kann. Wenn Sie

möchten, dass andere IAM-Prinzipale Zugriff auf Ihren Cluster haben, müssen Sie sie hinzufügen. Weitere Informationen erhalten Sie unter the section called “Kubernetes-API-Zugriff” und the section called “Erforderliche Berechtigungen”.

- Stellen Sie eine Beispielanwendung für Ihr Cluster bereit.
- Bevor Sie einen Cluster für die Produktion bereitstellen, empfehlen wir Ihnen, sich mit allen Einstellungen für Cluster und Knoten vertraut zu machen. Einige Einstellungen (z. B. die Aktivierung des SSH-Zugriffs auf EC2 Amazon-Knoten) müssen bei der Erstellung des Clusters vorgenommen werden.
- Um die Sicherheit für Ihren Cluster zu erhöhen, konfigurieren Sie das Amazon VPC Container Networking Interface-Plugin für die Verwendung von IAM-Rollen für Servicekonten.

Automatisieren der Cluster-Infrastruktur mit dem EKS-Automatikmodus

> **Tip**
>
> Melden Sie sich für bevorstehende Amazon EKS Auto Mode-Workshops an.

EKS Auto Mode erweitert die AWS Verwaltung von Kubernetes-Clustern über den Cluster selbst hinaus, sodass AWS Sie auch die Infrastruktur einrichten und verwalten können, die den reibungslosen Betrieb Ihrer Workloads ermöglicht. Sie können wichtige Infrastrukturentscheidungen delegieren und das Fachwissen von for Operations nutzen. AWS day-to-day Die Cluster-Infrastruktur, die von verwaltet wird, AWS umfasst viele Kubernetes-Funktionen als Kernkomponenten, im Gegensatz zu Add-Ons wie Compute-Autoscaling, Pod- und Service-Networking, Anwendungslastausgleich, Cluster-DNS, Blockspeicher und GPU-Unterstützung.

Für den Einstieg können Sie einen neuen EKS-Auto-Mode-Cluster bereitstellen oder EKS Auto Mode für einen vorhandenen Cluster aktivieren. Sie können Ihre EKS-Auto-Mode-Cluster mithilfe von eksctl, der AWS CLI, dem, EKS oder Ihren bevorzugten AWS-Managementkonsole infrastructure-as-code Tools bereitstellen APIs, aktualisieren oder ändern.

Mit EKS Auto Mode können Sie weiterhin Ihre bevorzugten Kubernetes-kompatiblen Tools verwenden. Der EKS Auto Mode lässt sich in AWS Dienste wie Amazon EC2, Amazon EBS und ELB integrieren und nutzt AWS Cloud-Ressourcen, die bewährten Methoden folgen. Diese Ressourcen werden automatisch skaliert, kostenoptimiert und regelmäßig aktualisiert, um die Betriebskosten und den Aufwand zu minimieren.

Features

EKS Auto Mode bietet die folgenden allgemeinen Features:

Optimierung der Kubernetes-Cluster-Verwaltung:: EKS Auto Mode optimiert die EKS-Verwaltung, indem er produktionsreife Cluster mit minimalem Betriebsaufwand bereitstellt. Mit EKS Auto Mode können Sie anspruchsvolle, dynamische Workloads zuverlässig ausführen, ohne dass Sie über fundierte EKS-Kenntnisse verfügen müssen.

Anwendungsverfügbarkeit: EKS Auto Mode fügt basierend auf den Anforderungen Ihrer Kubernetes-Anwendungen dynamisch Knoten in Ihrem EKS-Cluster hinzu oder entfernt sie. Dies minimiert den Bedarf an manueller Kapazitätsplanung und stellt die Anwendungsverfügbarkeit sicher

Effizienz: Der automatische Modus von EKS wurde entwickelt, um die Rechenkosten zu optimieren und gleichzeitig die Flexibilität zu wahren, die sich aus Ihren Anforderungen NodePool und den Workload-Anforderungen ergibt. Darüber hinaus werden nicht genutzte Instances beendet und Workloads auf anderen Knoten konsolidiert, um die Kosteneffizienz zu verbessern.

Sicherheit: Der automatische Modus von EKS verwendet für Ihre Knoten, AMIs die als unveränderlich behandelt werden. Diese AMIs setzen gesperrte Software durch, ermöglichen SELinux obligatorische Zugriffskontrollen und bieten schreibgeschützte Root-Dateisysteme. Darüber hinaus haben von EKS Auto Mode gestartete Knoten eine maximale Lebensdauer von 21 Tagen (die Sie reduzieren können), nach deren Ablauf sie automatisch durch neue Knoten ersetzt werden. Dieser Ansatz verbessert Ihre Sicherheitslage durch regelmäßiges Knotenwechsel und entspricht den bewährten Methoden, die bereits von vielen Kunden übernommen wurden.

Automatisierte Upgrades: Der automatische Modus von EKS hält Ihren Kubernetes-Cluster, Ihre Knoten und die zugehörigen Komponenten auf dem neuesten Stand und berücksichtigt gleichzeitig Ihre konfigurierten Pod-Disruption-Budgets () und Disruption-Budgets (PDBs). NodePool NDBs Bis zur maximalen Lebensdauer von 21 Tagen kann ein Eingreifen erforderlich sein, wenn Blockierungen PDBs oder andere Konfigurationen Updates verhindern.

Verwaltete Komponenten: Der automatische Modus von EKS umfasst Kubernetes- und AWS Cloud-Funktionen als Kernkomponenten, die andernfalls als Add-Ons verwaltet werden müssten. Dazu gehören integrierter Support für die Zuweisung von Pod-IP-Adressen, Pod-Netzwerkrichtlinien, lokale DNS-Services, GPU-Plug-ins, Zustandsprüfungen und EBS-CSI-Speicher.

Anpassbar NodePools und NodeClasses: Wenn Ihr Workload Änderungen an Speicher-, Rechen- oder Netzwerkkonfigurationen erfordert, können Sie benutzerdefinierte Konfigurationen erstellen NodePools und den EKS Auto Mode NodeClasses verwenden. Sie sollten die NodePools Standardkonfigurationen zwar nicht bearbeiten NodeClasses, aber Sie können neue benutzerdefinierte Konfigurationen NodePools oder NodeClasses zusätzlich zu den Standardkonfigurationen hinzufügen, um Ihren spezifischen Anforderungen gerecht zu werden.

Automatisierte Komponenten

EKS Auto Mode optimiert den Betrieb Ihrer Amazon-EKS-Cluster durch die Automatisierung wichtiger Infrastruktur-Komponenten. Durch die Aktivierung von EKS Auto Mode werden die Aufgaben zur Verwaltung Ihrer EKS-Cluster weiter reduziert.

Nachfolgend finden Sie eine Liste der automatisierten Datenebenen-Komponenten:

- Rechenleistung: Bei vielen Workloads können Sie mit EKS Auto Mode zahlreiche Aspekte der Rechenleistung für Ihre EKS-Cluster außer Acht lassen. Dazu zählen:
  - Knoten: Knoten in EKS Auto Mode sind so konzipiert, dass sie wie Geräte behandelt werden. EKS Auto Mode führt Folgendes durch:
    - Wählt ein geeignetes AMI, das mit vielen Services konfiguriert ist, die zum Ausführen Ihrer Workloads ohne Eingriff erforderlich sind.
    - Sperrt den Zugriff auf Dateien im AMI mithilfe des SELinux Erzwingungsmodus und eines schreibgeschützten Root-Dateisystems.
    - Verhindert den direkten Zugriff auf die Knoten, indem SSH- oder SSM-Zugriffe untersagt werden.
    - Beinhaltet GPU-Unterstützung mit separaten Kernel-Treibern und Plug-ins für NVIDIA und Neuron GPUs, was Hochleistungs-Workloads ermöglicht.
    - Verarbeitet automatisch Benachrichtigungen über Unterbrechungen der EC2 EC2 Spot-Instance und Instance-Integritätsereignisse
  - Auto Scaling: EKS Auto Mode basiert auf Auto Scaling von Karpenter, überwacht nicht planbare Pods und ermöglicht die Bereitstellung neuer Knoten zum Ausführen dieser Pods. Wenn Workloads beendet werden, unterbricht EKS Auto Mode dynamisch die Knoten und beendet sie, wenn sie nicht mehr benötigt werden, wodurch die Ressourcennutzung optimiert wird.
  - Upgrades: Durch die Übernahme der Kontrolle über Ihre Knoten kann EKS Auto Mode bei Bedarf Sicherheits-Patches sowie Betriebssystem- und Komponenten-Upgrades bereitstellen. Diese Upgrades sind so konzipiert, dass sie Ihre Workloads nur minimal beeinträchtigen. Der automatische EKS-Modus erzwingt eine maximale Knotenlebensdauer von 21 Tagen, um sicherzustellen, dass up-to-date Software und. APIs
- Load Balancing: EKS Auto Mode optimiert das Load Balancing durch die Integration mit dem Amazon-Service Elastic Load Balancing und automatisiert die Bereitstellung und Konfiguration von Load-Balancing-Modulen für Kubernetes-Services und Ingress-Ressourcen. Es unterstützt erweiterte Feature für Anwendungs- und Netzwerk-Load-Balancer, verwaltet deren Lebenszyklus

und skaliert sie entsprechend den Cluster-Anforderungen. Diese Integration bietet eine einsatzbereite Load-Balancing-Lösung, die den AWS Best Practices entspricht, sodass Sie sich auf Anwendungen statt auf das Infrastrukturmanagement konzentrieren können.

- Speicher: EKS Auto Mode konfiguriert flüchtigen Speicher für Sie, indem er Volumentypen, Volumengrößen, Verschlüsselungsrichtlinien und Löschrichtlinien bei Beendigung des Knotens einrichtet.
- Netzwerk: EKS Auto Mode automatisiert wichtige Netzwerkaufgaben für die Pod- und Service-Konnektivität. Dazu gehören IPv4 IPv6 /-Unterstützung und die Verwendung sekundärer CIDR-Blöcke zur Erweiterung von IP-Adressräumen.
- Identitäts- und Zugriffsverwaltung: Es ist nicht erforderlich, den EKS Pod Identity Agent in Clustern in EKS Auto Mode zu installieren.

Weitere Informationen zu diesen Komponenten finden Sie unter the section called “Funktionsweise”.

Konfiguration

Obwohl EKS Auto Mode die meisten Ihrer Datenebenen-Services ohne Ihr Zutun effektiv verwaltet, kann es vorkommen, dass Sie das Verhalten einiger dieser Services ändern möchten. Sie können die Konfiguration Ihrer Cluster in EKS Auto Mode auf folgende Weise ändern:

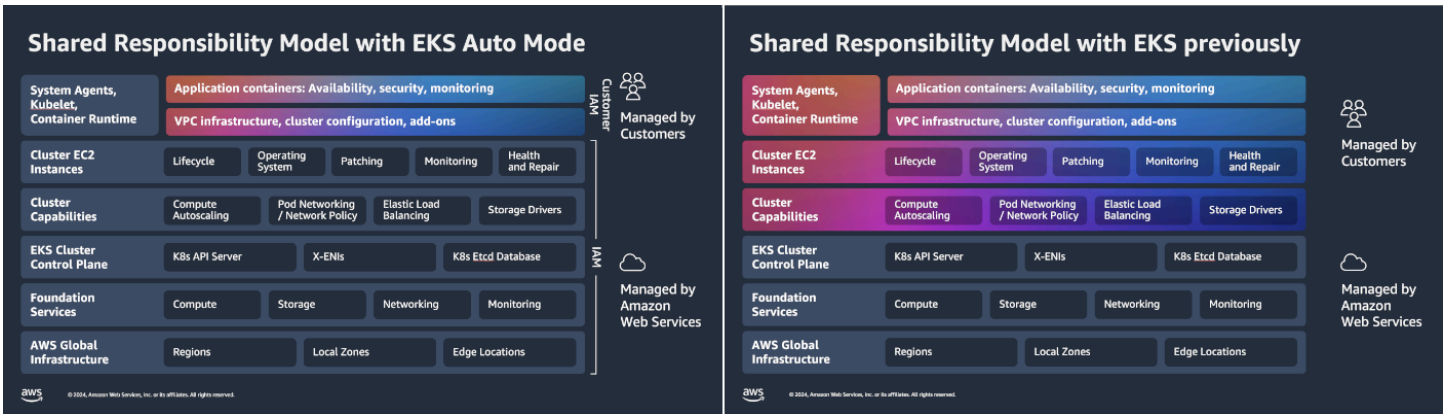
- Kubernetes DaemonSets: Anstatt die auf Ihren Knoten installierten Dienste zu ändern, können Sie stattdessen Kubernetes-Daemonsets verwenden. DaemonSets sind so konzipiert, dass sie von Kubernetes verwaltet werden, jedoch auf jedem Knoten im Cluster ausgeführt werden. Auf diese Weise können Sie spezielle Services für die Überwachung oder sonstige Beobachtung Ihrer Knoten hinzufügen.
- Benutzerdefiniert NodePools und NodeClasses: Standard NodePools und NodeClasses werden von EKS Auto Mode konfiguriert und sollten nicht bearbeitet werden. Um das Verhalten von Knoten anzupassen, können Sie zusätzliche NodePools oder NodeClasses für folgende Anwendungsfälle erstellen:
  - Auswahl bestimmter Instance-Typen (z. B. beschleunigte Prozessoren oder EC2 Spot-Instances).
  - Isolierung von Workloads aus Sicherheits- oder Kostenverfolgungsgründen.
  - Konfiguration von Einstellungen für flüchtigen Speicher wie IOPS, Größe und Durchsatz.

- Load Balancing: Bestimmte Services, wie beispielsweise Load Balancing, die EKS Auto Mode als Kubernetes-Objekte ausführen, können direkt auf Ihren EKS-Auto-Mode-Clustern konfiguriert werden.

Weitere Informationen zu den Optionen für die Konfiguration von EKS Auto Mode finden Sie unter the section called “Konfiguration”.

Modell der geteilten Verantwortung

Das Modell der AWS gemeinsamen Verantwortung definiert die Sicherheits- und Compliance-Verantwortlichkeiten zwischen Kunden AWS und Kunden. Die Bilder und der Text unten vergleichen und verdeutlichen, wie sich Kunden und AWS Zuständigkeiten zwischen dem EKS-Automodus und dem EKS-Standardmodus unterscheiden.



EKS Auto Mode verlagert einen Großteil der gemeinsamen Verantwortung für die Kubernetes-Infrastruktur von den Kunden auf AWS. Mit EKS Auto Mode AWS übernimmt EKS mehr Verantwortung für die Cloud-Sicherheit, die früher in der Verantwortung des Kunden lag und nun gemeinsam genutzt wird. Kunden können sich jetzt mehr auf ihre Anwendungen konzentrieren und gleichzeitig die zugrunde liegende Infrastruktur AWS verwalten.

Kundenverantwortung

In EKS Auto Mode behalten Kunden weiterhin die Verantwortung für die Anwendungs-Container, einschließlich Verfügbarkeit, Sicherheit und Überwachung. Sie behalten auch die Kontrolle über die VPC-Infrastruktur und die EKS-Cluster-Konfiguration. Dieses Modell ermöglicht es Kunden, sich auf anwendungsspezifische Belange zu konzentrieren und gleichzeitig die Verwaltung der Cluster-Infrastruktur an diese zu delegieren. AWS Optionale Funktionen pro Knoten können über Add-Ons in Cluster integriert werden. AWS

AWS -Verantwortung

Mit dem automatischen Modus von EKS AWS erweitert sich seine Verantwortung um die Verwaltung mehrerer zusätzlicher kritischer Komponenten im Vergleich zu den Komponenten, die bereits in EKS-Clustern verwaltet werden, die den automatischen Modus nicht verwenden. Der automatische Modus von EKS übernimmt insbesondere die Konfiguration, Verwaltung, Sicherheit und Skalierung der gestarteten EC2 Instances sowie die Clusterfunktionen für Lastenausgleich, IP-Adressverwaltung, Netzwerkrichtlinien und Blockspeicher. Die folgenden Komponenten werden AWS im EKS Auto Mode verwaltet:

- Automatisch im Modus gestartete EC2 Instances: AWS wickelt den gesamten Lebenszyklus von Knoten ab, indem von Amazon EC2 verwaltete Instances genutzt werden. EC2 verwaltete Instances übernehmen die Verantwortung für die Konfiguration, das Patchen, die Überwachung und die Aufrechterhaltung des Zustands des Betriebssystems. In diesem Modell liegen sowohl die Instance selbst als auch das darauf ausgeführte Gastbetriebssystem in der Verantwortung von AWS. Die Knoten verwenden Varianten von Bottlerocket, AMIs die für den Betrieb von Containern optimiert sind. Die Bottlerocket AMIs verfügen über gesperrte Software, unveränderliche Root-Dateisysteme und sicheren Netzwerkzugriff (um direkte Kommunikation über SSH oder SSM zu verhindern).
- Cluster-Funktionen: AWS verwaltet automatische Rechenskalierung, Pod-Netzwerke mit Durchsetzung von Netzwerkrichtlinien, Elastic Load Balancing Balancing-Integration und Speichertreiberkonfiguration.
- Cluster Control Plane: AWS Verwaltet weiterhin den Kubernetes-API-Server, Cros-Accounts ENIs und die etcd-Datenbank wie bei Standard-EKS.
- Foundation Services und globale Infrastruktur: AWS behält die Verantwortung für die zugrunde liegenden Rechen-, Speicher-, Netzwerk- und Überwachungsdienste sowie für die globale Infrastruktur von Regionen, lokalen Zonen und Edge-Standorten.

Einen Cluster mit Amazon EKS Auto Mode erstellen

In diesem Kapitel wird erläutert, wie Sie mithilfe verschiedener Tools und Schnittstellen einen Amazon-EKS-Cluster mit Auto Mode erstellen. Auto Mode vereinfacht die Cluster-Erstellung durch die automatische Konfiguration und Verwaltung der Rechen-, Netzwerk- und Speicherinfrastruktur des Clusters. Sie erfahren, wie Sie mithilfe der AWS-CLI, AWS-Managementkonsole oder des Befehlszeilentools eksctl einen Auto-Mode-Cluster erstellen.

> **Note**
>
> EKS Auto Mode erfordert Kubernetes Version 1.29 oder höher.

Wählen Sie Ihr bevorzugtes Tool entsprechend Ihren Anforderungen: Die AWS-Managementkonsole bietet eine visuelle Benutzeroberfläche, die sich ideal zum Kennenlernen der Feature von EKS Auto Mode und zum Erstellen einzelner Cluster eignet. Die AWS-CLI eignet sich am besten für Skripting- und Automatisierungsaufgaben, insbesondere bei der Integration der Cluster-Erstellung in vorhandene Workflows oder CI/CD-Pipelines. Die eksctl-CLI bietet eine Kubernetes-native Erfahrung und wird Benutzern empfohlen, die mit Kubernetes-Tools vertraut sind und vereinfachte Befehlszeilenvorgänge mit sinnvollen Standardeinstellungen wünschen.

Stellen Sie vor Beginn sicher, dass Sie über die erforderlichen Voraussetzungen verfügen, einschließlich der entsprechenden IAM-Berechtigungen zum Erstellen von EKS-Clustern. Informationen zur Installation von CLI-Tools wie `kubectl`, `aws` und `eksctl` finden Sie unter Einrichten.

Sie können die AWS-CLI, AWS-Managementkonsole, oder die eksctl CLI verwenden, um einen Cluster mit Amazon EKS Auto Mode zu erstellen.

Themen

- Erstellung eines EKS-Auto-Mode-Clusters mit der eksctl-Befehlszeilenschnittstelle
- Erstellen Sie einen EKS-Auto-Mode-Cluster mit der AWS CLI
- Erstellung eines EKS-Auto-Mode-Clusters mit AWS-Managementkonsole

Erstellung eines EKS-Auto-Mode-Clusters mit der eksctl-Befehlszeilenschnittstelle

In diesem Abschnitt wird erläutert, wie Sie einen EKS-Auto-Mode-Cluster mithilfe der Befehlszeilenschnittstelle (CLI) eksctl erstellen. Sie können einen EKS-Auto-Mode-Cluster entweder durch Ausführen eines einzelnen CLI-Befehls oder durch Anwenden einer YAML-Konfigurationsdatei erstellen. Beide Methoden bieten die gleiche Funktionalität, wobei der YAML-Ansatz eine detailliertere Kontrolle über die Cluster-Einstellungen ermöglicht.

Die eksctl CLI vereinfacht den Prozess der Erstellung und Verwaltung von EKS-Auto-Mode-Clustern, indem sie die Erstellung und Konfiguration der zugrunde liegenden AWS-Ressourcen übernimmt.

Stellen Sie vor dem Fortfahren sicher, dass Sie die erforderlichen AWS-Anmeldeinformationen und Berechtigungen auf Ihrem lokalen Rechner konfiguriert haben. Dieser Leitfaden setzt voraus, dass Sie mit den grundlegenden Konzepten von Amazon EKS vertraut sind und die erforderlichen CLI-Tools bereits installiert haben.

> **Note**
>
> Sie müssen die Version `0.195.0` oder höher von eksctl installieren. Weitere Informationen finden Sie unter eksctl-Versionen auf GitHub.

Erstellung eines EKS-Auto-Mode-Clusters mit einem CLI-Befehl

Sie müssen die `aws`- und `eksctl`-Tools installiert haben. Sie müssen bei der AWS-CLI mit ausreichenden Berechtigungen angemeldet sein, um AWS-Ressourcen zu verwalten, darunter: EC2-Instances, EC2-Netzwerke, EKS-Cluster und IAM-Rollen. Weitere Informationen finden Sie unter Einrichten.

Führen Sie den folgenden Befehl aus, um einen neuen EKS-Auto-Mode-Cluster zu erstellen

```
eksctl create cluster --name=<cluster-name> --enable-auto-mode
```

Erstellung eines EKS-Auto-Mode-Clusters mit einer YAML-Datei

Sie müssen die `aws`- und `eksctl`-Tools installiert haben. Sie müssen bei der AWS-CLI mit ausreichenden Berechtigungen angemeldet sein, um AWS-Ressourcen zu verwalten, darunter: EC2-Instances, EC2-Netzwerke, EKS-Cluster und IAM-Rollen. Weitere Informationen finden Sie unter Einrichten.

Überprüfen Sie die Konfigurationsoptionen für EKS Auto Mode in der folgenden Beispiel-ClusterConfig-Ressource. Die vollständige ClusterConfig-Spezifikation finden Sie in der eksctl-Dokumentation.

AWS empfiehlt, EKS Auto Mode zu aktivieren. Sollten Sie zum ersten Mal einen EKS-Auto-Mode-Cluster erstellen, lassen Sie `nodeRoleARN` unverändert, um eine Knoten-IAM-Rolle für EKS Auto Mode zu erstellen. Wenn Sie bereits über eine Knoten-IAM-Rolle in Ihrem AWS-Konto verfügen, empfiehlt AWS, diese wiederzuverwenden.

AWS empfiehlt, keinen Wert für `nodePools` anzugeben. EKS Auto Mode erstellt Standard-Knoten-Pools. Sie können die Kubernetes-API verwenden, um zusätzliche Knoten-Pools zu erstellen.

```
# cluster.yaml
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: <cluster-name>
  region: <aws-region>

iam:
  # ARN of the Cluster IAM Role
  # optional, eksctl creates a new role if not supplied
  # suggested to use one Cluster IAM Role per account
  serviceRoleARN: <arn-cluster-iam-role>

autoModeConfig:
  # defaults to false
  enabled: boolean
  # optional, defaults to [general-purpose, system].
  # suggested to leave unspecified
  # To disable creation of nodePools, set it to the empty array ([]).
  nodePools: []string
  # optional, eksctl creates a new role if this is not supplied
  # and nodePools are present.
  nodeRoleARN: string
```

Speichern Sie die `ClusterConfig`-Datei als `cluster.yaml` und verwenden Sie den folgenden Befehl, um den Cluster zu erstellen:

```
eksctl create cluster -f cluster.yaml
```

Erstellen Sie einen EKS-Auto-Mode-Cluster mit der AWS CLI

EKS-Auto-Mode-Cluster automatisieren routinemäßige Cluster-Verwaltungsaufgaben für Rechenleistung, Speicher und Netzwerke. EKS-Auto-Mode-Cluster erkennen beispielsweise automatisch, wenn zusätzliche Knoten erforderlich sind, und stellen neue EC2 Instanzen bereit, um den Workload-Anforderungen gerecht zu werden.

Dieses Thema führt Sie durch die Erstellung eines neuen EKS-Auto-Mode-Clusters mithilfe der AWS CLI und optional durch die Bereitstellung eines Beispiel-Workloads.

Voraussetzungen

- Die neueste Version der AWS Befehlszeilenschnittstelle (AWS CLI) ist auf Ihrem Gerät installiert und konfiguriert. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version`. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle.
  - Melden Sie sich mit ausreichenden IAM-Berechtigungen bei der CLI an, um AWS Ressourcen wie IAM-Richtlinien, IAM-Rollen und EKS-Cluster zu erstellen.
- Das kubectl-Befehlszeilentool, das auf Ihrem Gerät installiert ist. AWS schlägt vor, dass Sie dieselbe kubectl-Version wie die Kubernetes-Version Ihres EKS-Clusters verwenden. Informationen zum Installieren von kubectl finden Sie unter the section called “kubectl und eksctl einrichten”.

Angabe von VPC-Subnetzen

Amazon EKS Auto Mode stellt Knoten in VPC-Subnetzen bereit. Beim Erstellen eines EKS-Clusters müssen Sie die VPC-Subnetze angeben, in denen die Knoten bereitgestellt werden sollen. Sie können die Standard-VPC-Subnetze in Ihrem AWS Konto verwenden oder eine dedizierte VPC für kritische Workloads erstellen.

- AWS schlägt vor, eine dedizierte VPC für Ihren Cluster zu erstellen. Weitere Informationen erhalten Sie unter the section called “Erstellen einer VPC”.
- Die EKS-Konsole unterstützt Sie bei der Erstellung einer neuen VPC. Weitere Informationen erhalten Sie unter the section called “Management-Konsole”.
- Alternativ können Sie die Standard-VPC Ihres AWS Kontos verwenden. Verwenden Sie die folgenden Anweisungen, um das IDs Subnetz zu finden.

So finden Sie das Subnetz IDs Ihrer Standard-VPC

Verwenden der AWS CLI:

1. Führen Sie den folgenden Befehl aus, um die Standard-VPC und ihre Subnetze aufzulisten:

```
aws ec2 describe-subnets --filters "Name=vpc-id,Values=$(aws ec2 describe-vpcs
 --query 'Vpcs[?IsDefault==`true`].VpcId' --output text)" --query 'Subnets[*].
{ID:SubnetId,AZ:AvailabilityZone}' --output table
```

2. Speichern Sie die Ausgabe und notieren Sie sich das Subnetz IDs.

   Beispielausgabe:

```
----------------------------------------
|             DescribeSubnets          |
----------------------------------------
|   SubnetId        |   AvailabilityZone  |
|-------------------|---------------------|
|   subnet-012345678 |   us-west-2a        |
|   subnet-234567890 |   us-west-2b        |
|   subnet-345678901 |   us-west-2c        |
----------------------------------------
```

IAM-Rollen für Cluster in EKS Auto Mode

Cluster-IAM-Rolle

Der automatische EKS-Modus erfordert eine Cluster-IAM-Rolle, um Aktionen in Ihrem AWS Konto auszuführen, z. B. die Bereitstellung neuer Instanzen. EC2 Sie müssen diese Rolle erstellen, um EKS die erforderlichen Berechtigungen zu erteilen. AWS empfiehlt, die folgenden AWS verwalteten Richtlinien an die Cluster-IAM-Rolle anzuhängen:

- EKSComputeAmazon-Richtlinie
- AmazonEKSBlockStoragePolicy
- AmazonEKSLoadBalancingPolicy
- EKSNetworkingAmazon-Richtlinie
- EKSClusterAmazon-Richtlinie

Knoten-IAM-Rolle

Wenn Sie einen EKS-Auto-Mode-Cluster erstellen, geben Sie eine IAM-Rolle für den Knoten an. Wenn der automatische Modus von EKS Knoten zur Verarbeitung ausstehender Workloads erstellt, wird jedem neuen EC2 Instance-Knoten die Node-IAM-Rolle zugewiesen. Diese Rolle ermöglicht es dem Knoten, mit EKS zu kommunizieren, wird jedoch in der Regel nicht von den auf dem Knoten ausgeführten Workloads aufgerufen.

Wenn Sie Workloads, die auf einem Knoten ausgeführt werden, Berechtigungen gewähren möchten, verwenden Sie EKS Pod Identity. Weitere Informationen finden Sie unter the section called “Pod Identity”.

Sie müssen diese Rolle erstellen und die folgende AWS verwaltete Richtlinie anhängen:

- AmazonEKSWorkerNodeMinimalPolicy
- AmazonEC2ContainerRegistryPullOnly

Serviceverknüpfte Rolle

EKS Auto Mode erfordert außerdem eine serviceverknüpfte Rolle, die automatisch von AWS erstellt und konfiguriert wird. Weitere Informationen finden Sie unter AWSServiceRoleForAmazonEKS.

Erstellung einer IAM-Rolle für einen Cluster in EKS Auto Mode

Schritt 1: Vertrauensrichtlinie erstellen

Erstellen Sie eine Vertrauensrichtlinie, mit welcher der Amazon-EKS-Service die Rolle übernehmen kann. Speichern Sie die Richtlinie als `trust-policy.json`:

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
```

Schritt 2: Erstellen der IAM-Rolle

Verwenden Sie die Vertrauensrichtlinie, um die Cluster-IAM-Rolle zu erstellen:

```
aws iam create-role \
```

```
    --role-name AmazonEKSAutoClusterRole \
    --assume-role-policy-document file://trust-policy.json
```

Schritt 3: Rollen-ARN notieren

Rufen Sie die ARN der neuen Rolle ab und speichern Sie sie für die Verwendung in den folgenden Schritten:

```
aws iam get-role --role-name AmazonEKSAutoClusterRole --query "Role.Arn" --output text
```

Schritt 4: Erforderliche Richtlinien hinzufügen

Fügen Sie der Cluster-IAM-Rolle die folgenden AWS verwalteten Richtlinien hinzu, um die erforderlichen Berechtigungen zu gewähren:

EKSClusterAmazon-Richtlinie:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoClusterRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSClusterPolicy
```

EKSComputeAmazon-Richtlinie:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoClusterRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSComputePolicy
```

Amazon EKSBlock StoragePolicy:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoClusterRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSBlockStoragePolicy
```

Amazon EKSLoad BalancingPolicy:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoClusterRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSLoadBalancingPolicy
```

EKSNetworkingAmazon-Richtlinie:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoClusterRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSNetworkingPolicy
```

Erstellung einer Knoten-IAM-Rolle in EKS Auto Mode

Schritt 1: Vertrauensrichtlinie erstellen

Erstellen Sie eine Vertrauensrichtlinie, mit welcher der Amazon-EKS-Service die Rolle übernehmen kann. Speichern Sie die Richtlinie als `node-trust-policy.json`:

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Schritt 2: Knoten-IAM-Rolle erstellen

Verwenden Sie die node-trust-policyJSON-Datei aus dem vorherigen Schritt, um zu definieren, welche Entitäten die Rolle übernehmen können. Führen Sie den folgenden Befehl aus, um die Knoten-IAM-Rolle zu erstellen:

```
aws iam create-role \
    --role-name AmazonEKSAutoNodeRole \
    --assume-role-policy-document file://node-trust-policy.json
```

Schritt 3: Rollen-ARN notieren

Rufen Sie nach dem Erstellen der Rolle die ARN der Knoten-IAM-Rolle ab und speichern Sie sie. Sie benötigen diese ARN in den nachfolgenden Schritten. Um die ARN abzurufen, verwenden Sie den folgenden Befehl.

```
aws iam get-role --role-name AmazonEKSAutoNodeRole --query "Role.Arn" --output text
```

Schritt 4: Erforderliche Richtlinien hinzufügen

Fügen Sie der Node-IAM-Rolle die folgenden AWS verwalteten Richtlinien hinzu, um die erforderlichen Berechtigungen bereitzustellen:

Amazon EKSWorker NodeMinimalPolicy:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoNodeRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSWorkerNodeMinimalPolicy
```

Amazon EC2 ContainerRegistryPullOnly:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoNodeRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEC2ContainerRegistryPullOnly
```

Erstellung eines Clusters in EKS Auto Mode

-Übersicht

Um einen EKS-Auto-Mode-Cluster mit der AWS CLI zu erstellen, benötigen Sie die folgenden Parameter:

- `cluster-name`: Der Name des Clusters.
- `k8s-version`: Die Kubernetes-Version (z. B. 1.31).
- `subnet-ids`: In den vorherigen Schritten IDs identifiziertes Subnetz.
- `cluster-role-arn`: ARN der Cluster-IAM-Rolle.
- `node-role-arn`: ARN der Knoten-IAM-Rolle.

Standard-Cluster-Konfigurationen

Überprüfen Sie diese Standardwerte und Feature, bevor Sie den Cluster erstellen:

- `nodePools`: EKS Auto Mode beinhaltet universelle und systemseitige Standard-Knoten-Pools. Weitere Informationen zu Knoten-Pools

Hinweis: Knoten-Pools in EKS Auto Mode unterscheiden sich von Amazon EKS verwalteten Knotengruppen, können jedoch im selben Cluster koexistieren.

- `computeConfig.enabled`: Automatisiert routinemäßige Rechenaufgaben wie das Erstellen und Löschen EC2 von Instanzen.
- `kubernetesNetworkConfig.elasticLoadBalancing.enabled`: Automatisiert Load-Balancing-Aufgaben, einschließlich des Erstellens und Löschens von Elastic Load Balancers.
- `storageConfig.blockStorage.enabled`: Automatisiert Speicheraufgaben, wie das Erstellen und Löschen von Amazon-EBS-Volumes.
- `accessConfig.authenticationMode`: Erfordert EKS-Zugriffseinträge. Weitere Informationen zu EKS-Authentifizierungsmodi.

Ausführen des Befehls

Verwenden Sie den folgenden Befehl, um den Cluster zu erstellen:

```
aws eks create-cluster \
  --region ${AWS_REGION} \
  --cli-input-json \
  "{
      \"name\": \"${CLUSTER_NAME}\",
      \"version\": \"${K8S_VERSION}\",
      \"roleArn\": \"${CLUSTER_ROLE_ARN}\",
      \"resourcesVpcConfig\": {
        \"subnetIds\": ${SUBNETS_JSON},
        \"endpointPublicAccess\": true,
        \"endpointPrivateAccess\": true
      },
      \"computeConfig\": {
        \"enabled\": true,
        \"nodeRoleArn\":\"${NODE_ROLE_ARN}\",
        \"nodePools\": [\"general-purpose\", \"system\"]
      },
      \"kubernetesNetworkConfig\": {
        \"elasticLoadBalancing\": {
          \"enabled\": true
        }
      },
      \"storageConfig\": {
        \"blockStorage\": {
          \"enabled\": true
        }
      },
      \"accessConfig\": {
```

```
        \"authenticationMode\": \"API\"
      }
    }
```

Cluster-Status überprüfen

Schritt 1: Cluster-Erstellung überprüfen

Führen Sie den folgenden Befehl aus, um den Status Ihres Clusters zu überprüfen. Die Cluster-Erstellung dauert in der Regel 15 Minuten.

```
aws eks describe-cluster --name "${CLUSTER_NAME}" --output json
```

Schritt 2: kubeconfig aktualisieren

Sobald der Cluster bereit ist, aktualisieren Sie Ihre lokale kubeconfig-Datei, um die Kommunikation zwischen `kubectl` und dem Cluster zu ermöglichen. Diese Konfiguration verwendet die AWS CLI für die Authentifizierung.

```
aws eks update-kubeconfig --name "${CLUSTER_NAME}"
```

Schritt 3: Knoten-Pools überprüfen

Listen Sie die Knoten-Pools in Ihrem Cluster mit dem folgenden Befehl auf:

```
kubectl get nodepools
```

Nächste Schritte

- Erfahren Sie, wie Sie einen Beispiel-Workload in Ihrem neuen Cluster in EKS Auto Mode bereitstellen.

Erstellung eines EKS-Auto-Mode-Clusters mit AWS-Managementkonsole

Die Erstellung eines EKS-Auto-Mode-Clusters in AWS-Managementkonsole erfordert weniger Konfigurationsaufwand als andere Optionen. EKS lässt sich in AWS IAM- und VPC-Netzwerke integrieren, um Sie bei der Erstellung der einem EKS-Cluster zugeordneten Ressourcen zu unterstützen.

Sie haben zwei Möglichkeiten, einen Cluster in der Konsole zu erstellen:

- Schnellkonfiguration (mit EKS Auto Mode)
- Benutzerdefinierte Konfiguration

In diesem Abschnitt erfahren Sie, wie Sie einen EKS-Auto-Mode-Cluster mithilfe der Schnellkonfigurations-Option erstellen.

Erstellung von EKS Auto Mode mithilfe der Schnellkonfigurations-Option

Sie müssen mit ausreichenden Berechtigungen bei AWS-Managementkonsole angemeldet sein, um AWS-Ressourcen verwalten zu können, darunter: EC2-Instances, EC2-Netzwerke, EKS-Cluster und IAM-Rollen.

1. Navigieren Sie zur EKS-Konsole
2. Klicken Sie auf Cluster erstellen
3. Bestätigen Sie, dass die Option Schnellkonfiguration ausgewählt ist
4. Ermitteln Sie die folgenden Werte oder verwenden Sie die Standardwerte für einen Test-Cluster.
   - Cluster-Name
   - Kubernetes-Version
5. Wählen Sie die Cluster-IAM-Rolle aus. Wenn Sie zum ersten Mal einen EKS-Cluster im Automatikmodus erstellen, verwenden Sie die Option Empfohlene Rolle erstellen.
   - Optional können Sie eine einzelne Cluster-IAM-Rolle in Ihrem AWS-Konto für alle EKS-Auto-Mode-Cluster wiederverwenden.
   - Die Cluster-IAM-Rolle enthält die erforderlichen Berechtigungen für EKS Auto Mode, um Ressourcen wie EC2-Instances, EBS-Volumes und EC2 Load-Balancer zu verwalten.
   - Die Option Empfohlene Rolle erstellen füllt alle Felder mit empfohlenen Werten vorab aus. Wählen Sie Weiter und dann Erstellen aus. Die Rolle wird den vorgeschlagenen `AmazonEKSAutoClusterRole`-Namen verwenden.
   - Wenn Sie kürzlich eine neue Rolle erstellt haben, verwenden Sie das Symbol Aktualisieren, um die Dropdown-Liste für die Rollenauswahl neu zu laden.
6. Wählen Sie die Knoten-IAM-Rolle aus. Wenn Sie zum ersten Mal einen EKS-Cluster im Automatikmodus erstellen, verwenden Sie die Option Empfohlene Rolle erstellen.
   - Optional können Sie eine einzelne Knoten-IAM-Rolle in Ihrem AWS-Konto für alle EKS-Auto-Mode-Cluster wiederverwenden.

- Die Knoten-IAM-Rolle enthält die erforderlichen Berechtigungen für Knoten im Automatikmodus, um eine Verbindung zum Cluster herzustellen. Die Knoten-IAM-Rolle muss Berechtigungen zum Abrufen von ECR-Images für Ihre Container enthalten.
- Die Option Empfohlene Rolle erstellen füllt alle Felder mit empfohlenen Werten vorab aus. Wählen Sie Weiter und dann Erstellen aus. Die Rolle wird den vorgeschlagenen `AmazonEKSAutoNodeRole`-Namen verwenden.
- Wenn Sie kürzlich eine neue Rolle erstellt haben, verwenden Sie das Symbol Aktualisieren, um die Dropdown-Liste für die Rollenauswahl neu zu laden.

7. Wählen Sie die VPC für Ihren EKS-Auto-Mode-Cluster aus. Wählen Sie VPC erstellen, um eine neue VPC für EKS zu erstellen, oder wählen Sie eine VPC aus, die Sie zuvor für EKS erstellt haben.
   - Wenn Sie die VPC-Konsole zum Erstellen einer neuen VPC verwenden, empfiehlt AWS, mindestens ein NAT-Gateway pro Availability Zone zu erstellen. Andernfalls können Sie alle anderen Standardeinstellungen verwenden.
   - Weitere Informationen und Details zu den Anforderungen für IPv6-Cluster finden Sie unter the section called “Erstellen einer VPC”.
8. (Optional) EKS Auto Mode füllt automatisch die privaten Subnetze für die von Ihnen ausgewählte VPC. Sie können unerwünschte Subnetze entfernen.
   - EKS wählt automatisch private Subnetze aus der VPC gemäß bewährten Methoden aus. Optional können Sie zusätzliche Subnetze aus der VPC auswählen, beispielsweise öffentliche Subnetze.
9. (optional) Wählen Sie Standardeinstellungen für die Schnellkonfiguration anzeigen aus, um alle Konfigurationseinstellungen für den neuen Cluster zu überprüfen. Die Tabelle zeigt, dass einige Werte nach der Erstellung des Clusters nicht mehr bearbeitet werden können.
10. Wählen Sie Cluster erstellen aus. Beachten Sie, dass die Erstellung des Clusters bis zu fünfzehn Minuten dauern kann.

Nächste Schritte

- Erfahren Sie, wie Sie eine Beispiel-Workload in Ihrem Cluster in EKS Auto Mode bereitstellen.

Aktivierung von EKS Auto Mode in vorhandenen EKS-Clustern

Sie können EKS Auto Mode in vorhandenen EKS-Clustern aktivieren.

AWS unterstützt die folgenden Migrationen:

- Migration von Karpenter zu Knoten in EKS Auto Mode. Weitere Informationen finden Sie unter the section called “Von Karpenter migrieren”.
- Migration von EKS-verwalteten Knotengruppen zu Knoten in EKS Auto Mode. Weitere Informationen finden Sie unter the section called “Von MNGs migrieren”.
- Migration von EKS Fargate zu EKS Auto Mode. Weitere Informationen finden Sie unter the section called “Von Fargate migrieren”.

AWS unterstützt die folgenden Migrationen nicht:

- Migration von Volumes vom EBS-CSI-Controller (mithilfe des Amazon-EKS-Add-Ons) zum EBS-CSI-Controller in EKS Auto Mode (verwaltet durch EKS Auto Mode). Mit dem einen Controller erstellte PVCs können nicht vom anderen Controller eingebunden werden, da sie zwei verschiedene Kubernetes-Volume-Bereitsteller verwenden.
  - The `eks-auto-mode-ebs-migration-tool` (AWS-Labs-Projekt) ermöglicht die Migration zwischen Standard EBS CSI StorageClass (`ebs.csi.aws.com`) und EKS Auto EBS CSI StorageClass (`ebs.csi.eks.amazonaws.com`). Beachten Sie, dass für die Migration das Löschen und Neuerstellen vorhandener PersistentVolumeClaim/PersistentVolume-Ressourcen erforderlich ist. Daher ist vor der Implementierung eine Validierung in einer Nicht-Produktionsumgebung unerlässlich.
- Migration von Load Balancern vom AWS Load Balancer Controller zu EKS Auto Mode

  Sie können den AWS Load Balancer Controller auf einem Cluster in Amazon EKS Auto Mode installieren. Verwenden Sie die Optionen `IngressClass` oder `loadBalancerClass`, um Service- und Ingress-Ressourcen entweder dem Load Balancer Controller oder EKS Auto Mode zuzuordnen.
- Migration von EKS-Clustern mit alternativen CNIs oder anderen nicht unterstützten Netzwerk-Konfigurationen

Migrationsreferenz

Verwenden Sie die folgende Migrationsreferenz, um Kubernetes-Ressourcen so zu konfigurieren, dass sie entweder selbstverwalteten Controllern oder EKS Auto Mode zugeordnet sind.

Funktion	Ressource	Feld	Selbstverwaltet	EKS Auto Mode
Blockspeicher	`StorageClass`	`provisioner`	`ebs.csi.aws.com`	`ebs.csi.eks.amazonaws.com`
Load Balancing	`Service`	`loadBalancerClass`	`service.k8s.aws/nlb`	`eks.amazonaws.com/nlb`
Load Balancing	`IngressClass`	`controller`	`ingress.k8s.aws/alb`	`eks.amazonaws.com/alb`
Load Balancing	`IngressClassParams`	`apiversion`	`elbv2.k8s.aws/v1beta1`	`eks.amazonaws.com/v1`
Load Balancing	`TargetGroupBinding`	`apiversion`	`elbv2.k8s.aws/v1beta1`	`eks.amazonaws.com/v1`
Computing	`NodeClass`	`apiVersion`	`karpenter.sh/v1`	`eks.amazonaws.com/v1`

Migration von EBS-Volumes

Bei der Migration von Workloads in EKS Auto Mode ist es aufgrund unterschiedlicher CSI-Treiber-Bereitsteller erforderlich, die Migration von EBS-Volumes zu berücksichtigen:

- EKS-Auto-Mode-Bereitsteller: `ebs.csi.eks.amazonaws.com`
- Open-Source-EBS-CSI-Bereitsteller: `ebs.csi.aws.com`

Führen Sie die folgenden Schritte aus, um Ihre persistenten Volumes zu migrieren:

1. Volume-Aufbewahrungsrichtlinie ändern: Ändern Sie die vorhandenen Plattformversionen (PVs) `persistentVolumeReclaimPolicy` in `Retain`, um sicherzustellen, dass das zugrunde liegende EBS-Volume nicht gelöscht wird.
2. PV aus Kubernetes entfernen: Löschen Sie die alte PV-Ressource, während das aktuelle EBS-Volume erhalten bleibt.
3. Neue PV mit statischer Bereitstellung erstellen: Erstellen Sie eine neue PV, die auf dasselbe EBS-Volume verweist, jedoch mit dem Ziel-CSI-Treiber funktioniert.
4. An ein neues PVC binden: Erstellen Sie ein neues PVC, die mithilfe des Feldes `volumeName` speziell auf Ihre PV verweist.

Überlegungen

- Stellen Sie sicher, dass Ihre Anwendungen beendet sind, bevor Sie mit dieser Migration beginnen.
- Sichern Sie Ihre Daten, bevor Sie mit dem Migrationsprozess beginnen.
- Dieser Vorgang muss für jedes persistente Volume durchgeführt werden.
- Die Workload muss aktualisiert werden, um das neue PVC zu verwenden.

Migration von Load Balancern

Sie können vorhandene Load Balancer nicht direkt vom selbstverwalteten AWS Load Balancer Controller in EKS Auto Mode übertragen. Stattdessen müssen Sie eine Blau-Grün-Bereitstellungsstrategie implementieren. Dabei wird die vorhandene Load-Balancer-Konfiguration beibehalten, während neue Load Balancer unter dem verwalteten Controller erstellt werden.

Um Service Unterbrechungen zu minimieren, empfehlen wir einen DNS-basierten Ansatz zur Datenverkehrsverlagerung. Erstellen Sie zunächst neue Load Balancer in EKS Auto Mode, während Ihre vorhandene Konfiguration betriebsbereit bleibt. Verwenden Sie anschließend DNS-Routing (z. B. Route 53), um den Datenverkehr schrittweise von den alten Load Balancern auf die neuen umzuleiten. Sobald der Datenverkehr erfolgreich migriert wurde und Sie die neue Konfiguration überprüft haben, können Sie die alten Load Balancer und den selbstverwalteten Controller außer Betrieb nehmen.

Aktivierung von EKS Auto Mode in einem vorhandenen Cluster

In diesem Thema wird beschrieben, wie Sie Amazon EKS Auto Mode für Ihre vorhandenen Amazon-EKS-Cluster aktivieren. Um den Automatikmodus in einem vorhandenen Cluster zu aktivieren,

müssen die IAM-Berechtigungen aktualisiert und die zentralen EKS-Auto-Mode-Einstellungen konfiguriert werden. Nach der Aktivierung können Sie mit der Migration Ihrer vorhandenen Rechen-Workloads beginnen, um die vereinfachten Vorgänge und die automatisierte Infrastrukturverwaltung des Automatikmodus zu nutzen.

> **Important**
>
> Stellen Sie sicher, dass Sie die erforderliche Mindestversion bestimmter Amazon-EKS-Add-Ons installiert haben, bevor Sie EKS Auto Mode aktivieren. Weitere Informationen finden Sie unter the section called “Erforderliche Add-On-Versionen”.

Stellen Sie vor Beginn sicher, dass Sie über Administratorzugriff auf Ihren Amazon-EKS-Cluster und Berechtigungen zum Ändern von IAM-Rollen verfügen. Die Schritte in diesem Thema führen Sie durch die Aktivierung des automatischen Modus mit der AWS-Managementkonsole oder AWS CLI.

AWS-Managementkonsole

Sie müssen mit der Berechtigung zur Verwaltung von IAM, EKS und EC2 Ressourcen an der AWS Konsole angemeldet sein.

> **Note**
>
> Die Cluster-IAM-Rolle eines EKS-Clusters kann nach der Erstellung des Clusters nicht mehr geändert werden. EKS Auto Mode erfordert zusätzliche Berechtigungen für diese Rolle. Sie müssen der aktuellen Rolle zusätzliche Richtlinien anfügen.

Cluster-IAM-Rolle aktualisieren

1. Öffnen Sie Ihre Cluster-Übersichtsseite in der AWS-Managementkonsole.
2. Wählen Sie unter ARN der Cluster-IAM-Rolle die Option In IAM anzeigen aus.
3. Wählen Sie in der Dropdown-Liste Berechtigungen hinzufügen die Option Richtlinien hinzufügen aus.
4. Verwenden Sie das Feld Suche, um die folgenden Richtlinien zu suchen und auszuwählen:
   - `AmazonEKSComputePolicy`
   - `AmazonEKSBlockStoragePolicy`
   - `AmazonEKSLoadBalancingPolicy`

- `AmazonEKSNetworkingPolicy`
- `AmazonEKSClusterPolicy`

5. Wählen Sie Berechtigungen hinzufügen aus.
6. Wählen Sie auf der Registerkarte Vertrauensbeziehungen die Option Vertrauensrichtlinie bearbeiten aus
7. Fügen Sie die folgende Cluster-IAM-Rollen-Vertrauensrichtlinie ein und wählen Sie Richtlinie aktualisieren aus

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
```

EKS Auto Mode aktivieren

1. Öffnen Sie Ihre Cluster-Übersichtsseite in der AWS-Managementkonsole.
2. Wählen Sie unter EKS Auto Mode die Option Verwalten aus.
3. Aktivieren Sie EKS Auto Mode.
4. Wählen Sie aus der Dropdown-Liste EKS-Knoten-Pool die Standard-Knoten-Pools aus, die Sie erstellen möchten.
   - Weitere Informationen zu Knoten-Pools in EKS Auto Mode. Weitere Informationen finden Sie unter the section called “Knoten-Pool erstellen”.
5. Wenn Sie für dieses AWS Konto bereits eine EKS Auto Mode Node IAM-Rolle erstellt haben, wählen Sie sie in der Dropdownliste Node IAM Role aus. Falls Sie diese Rolle noch nicht erstellt haben, wählen Sie Empfohlene Rolle erstellen und befolgen Sie die Schritte.

AWS CLI

Voraussetzungen

- Die Cluster-IAM-Rolle des vorhandenen EKS-Clusters muss über ausreichende Berechtigungen für EKS Auto Mode verfügen, beispielsweise die folgenden Richtlinien:
  - `AmazonEKSComputePolicy`
  - `AmazonEKSBlockStoragePolicy`
  - `AmazonEKSLoadBalancingPolicy`
  - `AmazonEKSNetworkingPolicy`
  - `AmazonEKSClusterPolicy`
- Die Cluster-IAM-Rolle muss über eine aktualisierte Vertrauensrichtlinie verfügen, welche die Aktion `sts:TagSession` enthält. Weitere Informationen zum Erstellen einer Cluster-IAM-Rolle finden Sie unter the section called “ AWS CLI”.
- `aws`-CLI installiert, angemeldet und eine ausreichende Version vorhanden. Sie benötigen die Erlaubnis, IAM, EKS und EC2 Ressourcen zu verwalten. Weitere Informationen finden Sie unter Einrichten.

Verfahren

Verwenden Sie die folgenden Befehle, um EKS Auto Mode in einem vorhandenen Cluster zu aktivieren.

> **Note**
>
> Die Funktionen für Rechenleistung, Blockspeicher und Lastenausgleich müssen alle in derselben Anfrage aktiviert oder deaktiviert werden.

```
aws eks update-cluster-config \
 --name $CLUSTER_NAME \
 --compute-config enabled=true \
 --kubernetes-network-config '{"elasticLoadBalancing":{"enabled": true}}' \
 --storage-config '{"blockStorage":{"enabled": true}}'
```

Erforderliche Add-On-Versionen

Wenn Sie EKS Auto Mode in einem vorhandenen Cluster aktivieren möchten, müssen Sie möglicherweise bestimmte Add-Ons aktualisieren. Bitte beachten Sie Folgendes:

- Dies gilt nur für vorhandene Cluster, die in EKS Auto Mode umgestellt werden.
- Neue Cluster, die mit aktiviertem EKS Auto Mode erstellt wurden, erfordern diese Aktualisierungen nicht.

Wenn Sie eines der folgenden Add-Ons installiert haben, stellen Sie sicher, dass es mindestens die angegebene Mindestversion aufweist:

Add-On-Name	Erforderliche Mindestversion
Amazon-VPC-CNI-Plugin für Kubernetes	v1.19.0-eksbuild.1
Kube-proxy	• v1.26.15-eksbuild.19 • v1.27.16-eksbuild.14 • v1.28.15-eksbuild.4 • v1.29.10-eksbuild.3 • v1.30.6-eksbuild.3 • v1.31.2-eksbuild.3
Amazon-EBS-CSI-Treiber	v1.37.0-eksbuild.1
CSI-Snapshot-Controller	v8.1.0-eksbuild.2
EKS Pod Identity Agent	v1.3.4-eksbuild.1

Weitere Informationen finden Sie unter the section called “Add-On aktualisieren”.

Nächste Schritte

- Informationen zum Migrieren von Workloads der Knotengruppe finden Sie unter the section called “Von MNGs migrieren”.
- Informationen zur Migration von selbstverwaltetem Karpenter finden Sie unter the section called “Von Karpenter migrieren”.

Migration von Karpenter zu EKS Auto Mode mithilfe von kubectl

Dieses Thema führt Sie durch den Prozess der Migration von Workloads von Karpenter zu Amazon EKS Auto Mode mithilfe von kubectl. Die Migration kann schrittweise durchgeführt werden, sodass Sie Workloads in Ihrem eigenen Tempo verschieben können, während die Cluster-Stabilität und die Verfügbarkeit der Anwendungen während der gesamten Übergangsphase gewährleistet bleiben.

Der unten beschriebene step-by-step Ansatz ermöglicht es Ihnen, Karpenter und EKS Auto Mode während der Migrationsphase parallel auszuführen. Diese duale Betriebsstrategie gewährleistet einen nahtlosen Übergang, da Sie das Workload-Verhalten in EKS Auto Mode überprüfen können, bevor Sie Karpenter vollständig außer Betrieb nehmen. Sie können Anwendungen einzeln oder in Gruppen migrieren, was Ihnen Flexibilität bietet, um Ihren spezifischen betrieblichen Anforderungen und Ihrer Risikotoleranz gerecht zu werden.

Voraussetzungen

Stellen Sie vor Beginn der Migration sicher, dass Sie über Folgendes verfügen:

- Karpenter v1.1 oder höher muss auf Ihrem Cluster installiert sein. Weitere Informationen finden Sie unter Upgrade auf 1.1.0+ in der Karpenter-Dokumentation.
- `kubectl` installiert und mit Ihrem Cluster verbunden. Weitere Informationen finden Sie unter Einrichten.

Bei diesem Thema wird davon ausgegangen, dass Sie mit Karpenter und vertraut sind. NodePools Weitere Informationen finden Sie in der Dokumentation zu Karpenter.

Schritt 1: EKS Auto Mode auf dem Cluster aktivieren

Aktivieren Sie den EKS-Automatikmodus auf Ihrem vorhandenen Cluster mithilfe der AWS CLI oder der Managementkonsole. Weitere Informationen finden Sie unter the section called “Cluster aktivieren”.

> **Note**
>
> Aktivieren Sie den `general purpose`-Knoten-Pool in dieser Phase des Übergangs nicht, während Sie EKS Auto Mode aktivieren. Dieser Knoten-Pool ist nicht selektiv. Weitere Informationen finden Sie unter the section called “Integrierte Knoten-Pool überprüfen”.

Schritt 2: Erstellen Sie einen fehlerhaften EKS-Automatikmodus NodePool

Erstellen Sie einen neuen NodePool für EKS Auto Mode mit einem Makel. Dadurch wird sichergestellt, dass vorhandene Pods nicht automatisch auf den neuen EKS-Auto-Mode-Knoten geplant werden. Dieser Knoten-Pool verwendet den in EKS Auto Mode integrierten `default` `NodeClass`. Weitere Informationen finden Sie unter the section called “Knotenklasse erstellen”.

Beispiel für einen Knoten-Pool mit Taint:

```
apiVersion: karpenter.sh/v1
kind: NodePool
metadata:
  name: eks-auto-mode
spec:
  template:
    spec:
      requirements:
        - key: "eks.amazonaws.com/instance-category"
          operator: In
          values: ["c", "m", "r"]
      nodeClassRef:
        group: eks.amazonaws.com
        kind: NodeClass
        name: default
      taints:
        - key: "eks-auto-mode"
          effect: "NoSchedule"
```

Aktualisieren Sie die Anforderungen für den Knoten-Pool, damit sie mit der Karpenter-Konfiguration übereinstimmen, von der Sie migrieren. Sie benötigen mindestens eine Anforderung.

Schritt 3: Workloads für die Migration aktualisieren

Identifizieren und aktualisieren Sie die Workloads, die Sie in EKS Auto Mode migrieren möchten. Fügen Sie diesen Workloads sowohl Toleranzen als auch Knoten-Selektoren hinzu:

```
apiVersion: apps/v1
kind: Deployment
spec:
  template:
    spec:
      tolerations:
```

```
      - key: "eks-auto-mode"
        effect: "NoSchedule"
      nodeSelector:
        eks.amazonaws.com/compute-type: auto
```

Durch diese Änderung kann die Workload auf den neuen EKS-Auto-Mode-Knoten geplant werden.

EKS Auto Mode verwendet andere Bezeichnungen als Karpenter. Beschriftungen, die sich auf EC2 verwaltete Instanzen beziehen, beginnen mit`eks.amazonaws.com`. Weitere Informationen finden Sie unter the section called “Knoten-Pool erstellen”.

Schritt 4: Workloads schrittweise migrieren

Wiederholen Sie Schritt 3 für jede Workload, die Sie migrieren möchten. Auf diese Weise können Sie Workloads einzeln oder in Gruppen verschieben, je nach Ihren Anforderungen und Ihrer Risikotoleranz.

Schritt 5: Entfernen Sie den ursprünglichen Karpenter NodePool

Sobald alle Workloads migriert wurden, können Sie den ursprünglichen Karpenter entfernen: NodePool

```
kubectl delete nodepool <original-nodepool-name>
```

Schritt 6: Entfernen Sie den Makel aus dem EKS-Automatikmodus (optional) NodePool

Wenn Sie möchten, dass der EKS-Automodus zum Standard für neue Workloads wird, können Sie den Makel aus dem EKS-Automatikmodus entfernen: NodePool

```
apiVersion: karpenter.sh/v1
kind: NodePool
metadata:
  name: eks-auto-mode
spec:
  template:
    spec:
      nodeClassRef:
        group: eks.amazonaws.com
        kind: NodeClass
        name: default
      # Remove the taints section
```

Schritt 7: Knoten-Selektoren aus Workloads entfernen (optional)

Wenn Sie den Makel aus dem EKS-Automatikmodus entfernt haben NodePool, können Sie optional die Node-Selectors aus Ihren Workloads entfernen, da der EKS-Automodus jetzt der Standard ist:

```
apiVersion: apps/v1
kind: Deployment
spec:
  template:
    spec:
      # Remove the nodeSelector section
      tolerations:
      - key: "eks-auto-mode"
        effect: "NoSchedule"
```

Schritt 8: Karpenter von Ihrem Cluster deinstallieren

Die Schritte zum Entfernen von Karpenter hängen davon ab, wie Sie es installiert haben. Weitere Informationen finden Sie in den Installationsanweisungen für Karpenter.

Migration von EKS-verwalteten Knotengruppen zu EKS Auto Mode

Bei der Umstellung Ihres Amazon-EKS-Clusters auf den EKS Auto Mode können Sie Ihre vorhandenen Workloads mithilfe des eksctl-CLI-Tools nahtlos aus verwalteten Knotengruppen (MNGs) migrieren. Dieser Vorgang gewährleistet die kontinuierliche Verfügbarkeit Ihrer Anwendungen, während EKS Auto Mode Ihre Rechenressourcen optimiert. Die Migration kann mit minimalen Unterbrechungen Ihrer ausgeführten Anwendungen durchgeführt werden.

Dieses Thema führt Sie durch die Schritte, mit denen Sie Pods sicher aus Ihren vorhandenen verwalteten Knotengruppen entfernen und EKS Auto Mode diese auf neu bereitgestellten Instances neu planen lassen können. Durch Befolgen dieses Verfahrens können Sie die Vorteile der intelligenten Workload-Konsolidierung von EKS Auto Mode nutzen und gleichzeitig die Verfügbarkeit Ihrer Anwendung während der gesamten Migration aufrechterhalten.

Voraussetzungen

- Cluster mit aktiviertem EKS Auto Mode
- `eksctl`-CLI installiert und mit Ihrem Cluster verbunden. Weitere Informationen finden Sie unter Einrichten.

- Karpenter ist auf dem Cluster nicht installiert.

Verfahren

Verwenden Sie den folgenden `eksctl`-CLI-Befehl, um das Entleeren von Pods aus den vorhandenen verwalteten Knotengruppen-Instances zu initiieren. EKS Auto Mode erstellt neue Knoten, um die verschobenen Pods zu sichern.

```
eksctl delete nodegroup --cluster=<clusterName> --name=<nodegroupName>
```

Sie müssen diesen Befehl für jede verwaltete Knotengruppe in Ihrem Cluster ausführen.

Weitere Informationen zu diesem Befehl finden Sie unter Knotengruppen löschen und leeren in der eksctl-Dokumentation.

Migration von EKS Fargate zu EKS Auto Mode

In diesem Thema wird der Prozess der Migration von Workloads von EKS Fargate zu Amazon EKS Auto Mode mithilfe von `kubectl` erläutert Die Migration kann schrittweise durchgeführt werden, sodass Sie Workloads in Ihrem eigenen Tempo verschieben können, während die Cluster-Stabilität und die Verfügbarkeit der Anwendungen während der gesamten Übergangsphase gewährleistet bleiben.

Der unten beschriebene step-by-step Ansatz ermöglicht es Ihnen, EKS Fargate und EKS Auto Mode während des Migrationszeitraums parallel auszuführen. Diese Dual-Betriebsstrategie trägt zu einem nahtlosen Übergang bei, indem sie Ihnen die Möglichkeit bietet, das Workload-Verhalten in EKS Auto Mode zu validieren, bevor Sie EKS Fargate vollständig außer Betrieb nehmen. Sie können Anwendungen einzeln oder in Gruppen migrieren, was Ihnen Flexibilität bietet, um Ihren spezifischen betrieblichen Anforderungen und Ihrer Risikotoleranz gerecht zu werden.

Vergleich von Amazon EKS Auto Mode und EKS mit AWS Fargate

Amazon EKS mit AWS Fargate bleibt eine Option für Kunden, die EKS ausführen möchten, aber Amazon EKS Auto Mode ist in Zukunft der empfohlene Ansatz. EKS Auto Mode ist vollständig Kubernetes-konform und unterstützt alle Upstream-Kubernetes-Primitive und Plattform-Tools wie Istio, die Fargate nicht unterstützen kann. Der automatische Modus von EKS unterstützt außerdem vollständig alle EC2 Runtime-Kaufoptionen, einschließlich GPU- und Spot-Instances, sodass Kunden ausgehandelte EC2 Rabatte und andere Sparmechanismen nutzen können. Diese Funktionen sind bei der Verwendung von EKS mit Fargate nicht verfügbar.

Darüber hinaus ermöglicht EKS Auto Mode Kunden, dasselbe Isolationsmodell wie Fargate zu erreichen und dabei die standardmäßigen Kubernetes-Planungsfunktionen zu verwenden, um sicherzustellen, dass auf jeder EC2 Instanz ein einziger Anwendungscontainer ausgeführt wird. Durch die Einführung von Amazon EKS Auto Mode können Kunden alle Vorteile der Ausführung von Kubernetes nutzen AWS — eine vollständig Kubernetes-konforme Plattform, die die Flexibilität bietet, die gesamte Bandbreite an Kaufoptionen zu nutzen und gleichzeitig die Benutzerfreundlichkeit EC2 und Abstraktion von der Infrastrukturverwaltung beizubehalten, die Fargate bietet.

Voraussetzungen

Stellen Sie vor Beginn der Migration sicher, dass Sie über Folgendes verfügen:

- Richten Sie einen Cluster mit Fargate ein. Weitere Informationen finden Sie unter the section called “Erste Schritte”.
- Installiert und `kubectl` mit Ihrem Cluster verbunden. Weitere Informationen finden Sie unter Einrichten.

Schritt 1: Fargate-Cluster überprüfen

1. Ausführung der EKS-Cluster mit Fargate überprüfen:

```
kubectl get node
```

```
NAME STATUS ROLES AGE VERSION
fargate-ip-192-168-92-52.ec2.internal Ready <none> 25m v1.30.8-eks-2d5f260
fargate-ip-192-168-98-196.ec2.internal Ready <none> 24m v1.30.8-eks-2d5f260
```

2. Ausgeführte Pods überprüfen:

```
kubectl get pod -A
```

```
NAMESPACE NAME READY STATUS RESTARTS AGE
kube-system coredns-6659cb98f6-gxpjz 1/1 Running 0 26m
kube-system coredns-6659cb98f6-gzzsx 1/1 Running 0 26m
```

3. Bereitstellung in einer Datei mit dem Namen `deployment_fargate.yaml` erstellen:

```
apiVersion: apps/v1
```

```
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
      annotations:
        eks.amazonaws.com/compute-type: fargate
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
```

4. Bereitstellung durchführen:

```
kubectl apply -f deployment_fargate.yaml
```

```
deployment.apps/nginx-deployment created
```

5. Pods und Bereitstellungen überprüfen:

```
kubectl get pod,deploy
```

```
NAME                                    READY   STATUS    RESTARTS   AGE
pod/nginx-deployment-5c7479459b-6trtm   1/1     Running   0          61s
pod/nginx-deployment-5c7479459b-g8ssb   1/1     Running   0          61s
pod/nginx-deployment-5c7479459b-mq4mf   1/1     Running   0          61s

NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/nginx-deployment   3/3     3            3           61s
```

6. Knoten überprüfen:

```
kubectl get node -owide
```

```
NAME                                       STATUS  ROLES  AGE VERSION
   INTERNAL-IP     EXTERNAL-IP OS-IMAGE       KERNEL-VERSION
 CONTAINER-RUNTIME
fargate-ip-192-168-111-43.ec2.internal  Ready   <none> 31s v1.30.8-eks-2d5f260
 192.168.111.43  <none>      Amazon Linux 2 5.10.234-225.910.amzn2.x86_64
 containerd://1.7.25
fargate-ip-192-168-117-130.ec2.internal Ready   <none> 36s v1.30.8-eks-2d5f260
 192.168.117.130 <none>      Amazon Linux 2 5.10.234-225.910.amzn2.x86_64
 containerd://1.7.25
fargate-ip-192-168-74-140.ec2.internal  Ready   <none> 36s v1.30.8-eks-2d5f260
 192.168.74.140  <none>      Amazon Linux 2 5.10.234-225.910.amzn2.x86_64
 containerd://1.7.25
```

Schritt 2: EKS Auto Mode auf dem Cluster aktivieren

1. Aktivieren Sie den EKS-Automatikmodus auf Ihrem vorhandenen Cluster mithilfe der AWS CLI oder der Managementkonsole. Weitere Informationen finden Sie unter the section called “Cluster aktivieren”.
2. Knoten-Pool überprüfen:

```
kubectl get nodepool
```

```
NAME              NODECLASS   NODES   READY   AGE
general-purpose   default     1       True    6m58s
system            default     0       True    3d14h
```

Schritt 3: Workloads für die Migration aktualisieren

Identifizieren und aktualisieren Sie die Workloads, die Sie in EKS Auto Mode migrieren möchten.

Um eine Workload von Fargate nach EKS Auto Mode zu migrieren, wenden Sie die Annotation `eks.amazonaws.com/compute-type: ec2` an. Dadurch wird sichergestellt, dass die Arbeitslast trotz des Fargate-Profils nicht von Fargate geplant wird, sondern vom EKS-Automatikmodus

übernommen wird. NodePool Weitere Informationen finden Sie unter the section called “Knoten-Pool erstellen”.

1. Ändern Sie Ihre Bereitstellungen (beispielsweise die `deployment_fargate.yaml`-Datei), um den Rechentyp auf `ec2` umzustellen:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
      annotations:
        eks.amazonaws.com/compute-type: ec2
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
```

2. Wenden Sie die Bereitstellung an. Durch diese Änderung kann die Workload auf den neuen EKS-Auto–Mode-Knoten geplant werden:

```
kubectl apply -f deployment_fargate.yaml
```

3. Überprüfen Sie, ob die Bereitstellung im EKS-Auto-Mode-Cluster ausgeführt wird:

```
kubectl get pod -o wide
```

```
NAME                              READY   STATUS    RESTARTS   AGE     IP
    NODE                  NOMINATED NODE   READINESS GATES
```

```
nginx-deployment-97967b68d-ffxxh   1/1     Running   0          3m31s
 192.168.43.240   i-0845aafcb51630ffb   <none>           <none>
nginx-deployment-97967b68d-mbcgj   1/1     Running   0          2m37s
 192.168.43.241   i-0845aafcb51630ffb   <none>           <none>
nginx-deployment-97967b68d-qpd8x   1/1     Running   0          2m35s
 192.168.43.242   i-0845aafcb51630ffb   <none>           <none>
```

4. Überprüfen Sie, ob kein Fargate-Knoten ausgeführt wird und keine Bereitstellung in den verwalteten Knoten in EKS Auto Mode ausgeführt wird:

```
kubectl get node -owide
```

```
NAME                  STATUS ROLES  AGE   VERSION              INTERNAL-IP     EXTERNAL-
IP OS-IMAGE                                          KERNEL-VERSION CONTAINER-RUNTIME
i-0845aafcb51630ffb Ready  <none> 3m30s v1.30.8-eks-3c20087 192.168.41.125
 3.81.118.95 Bottlerocket (EKS Auto) 2025.3.14 (aws-k8s-1.30) 6.1.129
 containerd://1.7.25+bottlerocket
```

Schritt 4: Workloads schrittweise migrieren

Wiederholen Sie Schritt 3 für jede Workload, die Sie migrieren möchten. Auf diese Weise können Sie Workloads einzeln oder in Gruppen verschieben, je nach Ihren Anforderungen und Ihrer Risikotoleranz.

Schritt 5: Das ursprüngliche Fargate-Profil entfernen

Sobald alle Workloads migriert wurden, können Sie das ursprüngliche `fargate`-Profil entfernen. Ersetze es *<fargate profile name>* durch den Namen deines Fargate-Profils:

```
aws eks delete-fargate-profile --cluster-name eks-fargate-demo-cluster --fargate-
profile-name <fargate profile name>
```

Schritt 6: CoreDNS herunterskalieren

Da EKS Auto Mode CoreDNS verwaltet, skalieren Sie die `coredns`-Bereitstellung auf 0 herunter:

```
kubectl scale deployment coredns -n kube-system —-replicas=0
```

Ausführung von Beispiel-Workloads in Clustern in EKS Auto Mode

Dieses Kapitel enthält Beispiele dafür, wie verschiedene Arten von Workloads in Amazon-EKS-Clustern bereitgestellt werden können, die im Automatikmodus ausgeführt werden. Die Beispiele veranschaulichen wichtige Workload-Muster, darunter Beispielanwendungen, Web-Anwendungen mit Lastenausgleich, zustandsbehaftete Workloads mit persistenter Speicherung und Workloads mit spezifischen Anforderungen an die Knoten-Platzierung. Jedes Beispiel enthält vollständige Manifeste und step-by-step Bereitstellungsanweisungen, die Sie als Vorlagen für Ihre eigenen Anwendungen verwenden können.

Bevor Sie mit den Beispielen fortfahren, stellen Sie sicher, dass ein EKS-Cluster im Auto-Modus läuft und dass Sie AWS CLI und kubectl installiert haben. Weitere Informationen finden Sie unter Einrichten. Die Beispiele setzen grundlegende Kenntnisse der Kubernetes-Konzepte und kubectl-Befehle voraus.

Sie können diese auf Anwendungsfällen basierenden Beispiele verwenden, um Workloads in Clustern in EKS Auto Mode auszuführen.

the section called “Erhöhte Workload bereitstellen”

Zeigt, wie Sie mithilfe von `kubectl`-Befehlen einen Beispiel-Workload in einem Cluster in EKS Auto Mode bereitstellen.

the section called “Load Balancer bereitstellen”

Zeigt, wie Sie eine containerisierte Version des Spiels 2048 in Amazon EKS bereitstellen.

the section called “Workload mit Statusverwaltung bereitstellen”

Zeigt, wie Sie eine Beispielanwendung mit Status in einem Cluster in EKS Auto Mode bereitstellen.

the section called “Beschleunigten Workload bereitstellen”

Zeigt, wie hardwarebeschleunigte Workloads auf Knoten bereitgestellt werden, die vom EKS Auto Mode verwaltet werden.

the section called “Kontrollbereitstellung”

Zeigt, wie Sie mithilfe einer Annotation steuern, ob eine Workload auf Knoten bereitgestellt wird, die von EKS Auto Mode verwaltet werden.

Bereitstellung einer erhöhten Beispiel-Workload in einem Cluster in Amazon EKS Auto Mode

In diesem Tutorial erfahren Sie, wie Sie eine Beispiel-Workload in einem EKS-Auto-Mode-Cluster bereitstellen und beobachten, wie dieser automatisch die erforderlichen Rechenressourcen bereitstellt. Sie werden `kubectl` Befehle verwenden, um das Verhalten des Clusters zu beobachten und aus erster Hand zu erfahren, wie der Auto Mode den Kubernetes-Betrieb vereinfacht. AWS Am Ende dieses Tutorials werden Sie verstehen, wie EKS Auto Mode auf Workload-Bereitstellungen reagiert, indem er die zugrunde liegenden Rechenressourcen automatisch verwaltet, ohne dass eine manuelle Konfiguration der Knotengruppe erforderlich ist.

Voraussetzungen

- Ein Cluster in Amazon EKS Auto Mode. Notieren Sie sich den Namen und die AWS Region des Clusters.
- Ein IAM-Prinzipal, beispielsweise ein Benutzer oder eine Rolle, mit ausreichenden Berechtigungen zur Verwaltung von Netzwerk-, Rechen- und EKS-Ressourcen.
  - Weitere Informationen finden Sie unter Erstellung von Rollen und Hinzufügen von Richtlinien im IAM-Benutzerhandbuch.
- `aws`-CLI installiert und mit einer IAM-Identität konfiguriert.
- `kubectl`-CLI installiert und mit dem Cluster verbunden.
  - Weitere Informationen finden Sie unter Einrichten.

Schritt 1: Vorhandene Rechenressourcen überprüfen (optional)

Verwenden Sie zunächst `kubectl`, um die Knoten-Pools in Ihrem Cluster aufzulisten.

```
kubectl get nodepools
```

Beispiel-Ausgabe:

```
general-purpose
```

In diesem Tutorial werden wir eine Workload bereitstellen, die für die Verwendung des `general-purpose`-Knoten-Pools konfiguriert ist. Dieser Knoten-Pool ist in EKS Auto Mode integriert und enthält sinnvolle Standardeinstellungen für allgemeine Workloads wie Microservices und Web-

Anwendungen. Sie können Ihren eigenen Knoten-Pool erstellen. Weitere Informationen finden Sie unter the section called “Knoten-Pool erstellen”.

Verwenden Sie anschließend `kubectl`, um die mit Ihrem Cluster verbundenen Knoten aufzulisten.

```
kubectl get nodes
```

Wenn Sie gerade einen EKS-Auto-Mode-Cluster erstellt haben, werden keine Knoten angezeigt.

In diesem Tutorial werden Sie eine Beispiel-Workload bereitstellen. Wenn keine Knoten vorhanden sind oder die Workload nicht auf die vorhandenen Knoten passt, wird der EKS-Auto-Mode-Cluster einen neuen Knoten bereitstellen

Schritt 2: Beispielanwendung im Cluster bereitstellen

Überprüfen Sie die folgende Kubernetes-Bereitstellung und speichern Sie sie als `inflate.yaml`

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: inflate
spec:
  replicas: 1
  selector:
    matchLabels:
      app: inflate
  template:
    metadata:
      labels:
        app: inflate
    spec:
      terminationGracePeriodSeconds: 0
      nodeSelector:
        eks.amazonaws.com/compute-type: auto
      securityContext:
        runAsUser: 1000
        runAsGroup: 3000
        fsGroup: 2000
      containers:
        - name: inflate
          image: public.ecr.aws/eks-distro/kubernetes/pause:3.7
          resources:
```

```
            requests:
              cpu: 1
          securityContext:
            allowPrivilegeEscalation: false
```

Beachten Sie, dass für den `eks.amazonaws.com/compute-type: auto`-Selektor die Bereitstellung des Workloads in einem Knoten von Amazon EKS Auto Mode erforderlich ist.

Wenden Sie die Bereitstellung auf Ihren Cluster an.

```
kubectl apply -f inflate.yaml
```

Schritt 3: Kubernetes-Ereignisse beobachten

Verwenden Sie den folgenden Befehl, um Kubernetes-Ereignisse zu überwachen, einschließlich der Erstellung eines neuen Knotens. Verwenden Sie `ctrl+c`, um die Überwachung von Ereignissen zu beenden.

```
kubectl get events -w --sort-by '.lastTimestamp'
```

Verwenden Sie `kubectl`, um die mit Ihrem Cluster verbundenen Knoten erneut aufzulisten. Notieren Sie sich den neu erstellten Knoten.

```
kubectl get nodes
```

Schritt 4: Knoten und Instanzen in der AWS Konsole anzeigen

Sie können die EKS-Automodus-Knoten in der EKS-Konsole und die zugehörigen EC2 Instanzen in der EC2 Konsole anzeigen.

EC2 Instances, die im automatischen Modus von EKS bereitgestellt werden, sind eingeschränkt. Sie können in EKS-Auto-Mode-Knoten keine beliebigen Befehle ausführen.

Schritt 5: Bereitstellung löschen

Verwenden Sie `kubectl`, um die Beispielbereitstellung zu löschen

```
kubectl delete -f inflate.yaml
```

Wenn keine weiteren Workloads in Ihrem Cluster bereitgestellt sind, bleibt der vom EKS Auto Mode erstellte Knoten leer.

In der Standardkonfiguration erkennt EKS Auto Mode Knoten, die seit 30 Sekunden leer sind, und beendet sie.

Verwenden Sie `kubectl` oder die EC2 Konsole, um zu bestätigen, dass die zugehörige Instanz gelöscht wurde.

Bereitstellung einer Beispiel-Workload für Load Balancer in EKS Auto Mode

Dieser Leitfaden führt Sie durch die Bereitstellung einer containerisierten Version des Spiels 2048 in Amazon EKS, einschließlich Load Balancing und Internetzugänglichkeit.

Voraussetzungen

- Ein EKS-Auto-Mode-Cluster
- `kubectl` für die Interaktion mit Ihrem Cluster konfiguriert
- Geeignete IAM-Berechtigungen für die Erstellung von ALB-Ressourcen

Schritt 1: Namespace erstellen

Erstellen Sie zunächst einen dedizierten Namespace für die 2048-Spiel-Anwendung.

Erstellen Sie eine Datei namens `01-namespace.yaml`:

```
apiVersion: v1
kind: Namespace
metadata:
  name: game-2048
```

Wenden Sie die Namespace-Konfiguration an:

```
kubectl apply -f 01-namespace.yaml
```

Schritt 2: Anwendung bereitstellen

Die Anwendung führt mehrere Replikate des 2048-Spiel-Containers aus.

Erstellen Sie eine Datei namens `02-deployment.yaml`:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  namespace: game-2048
  name: deployment-2048
spec:
  selector:
    matchLabels:
      app.kubernetes.io/name: app-2048
  replicas: 5
  template:
    metadata:
      labels:
        app.kubernetes.io/name: app-2048
    spec:
      containers:
        - image: public.ecr.aws/l6m2t8p7/docker-2048:latest
          imagePullPolicy: Always
          name: app-2048
          ports:
            - containerPort: 80
          resources:
            requests:
              cpu: "0.5"
```

> **Note**
>
> Sollten beim Laden des Images `public.ecr.aws/l6m2t8p7/docker-2048:latest` Fehler auftreten, überprüfen Sie, ob Ihre Knoten-IAM-Rolle über ausreichende Berechtigungen verfügt, um Images aus ECR abzurufen. Weitere Informationen finden Sie unter the section called “Knoten-IAM-Rolle”. Beachten Sie, dass es sich bei dem `docker-2048`-Image im Beispiel um ein `x86_64`-Image handelt, das in anderen Architekturen nicht ausgeführt werden kann.

Zentrale Komponenten:

- Stellt 5 Replikate der Anwendung bereit
- Verwendet ein öffentliches ECR-Image

- Erfordert 0,5 CPU-Kerne pro Pod
- Stellt Port 80 für HTTP-Datenverkehr bereit

Bereitstellung durchführen:

```
kubectl apply -f 02-deployment.yaml
```

Schritt 3: Service erstellen

Der Service macht die Bereitstellung für das Cluster-Netzwerk sichtbar.

Erstellen Sie eine Datei namens `03-service.yaml`:

```
apiVersion: v1
kind: Service
metadata:
  namespace: game-2048
  name: service-2048
spec:
  ports:
    - port: 80
      targetPort: 80
      protocol: TCP
  selector:
    app.kubernetes.io/name: app-2048
```

Zentrale Komponenten:

- Erstellt einen NodePort-Service
- Ordnet Port 80 dem Port 80 des Containers zu
- Verwendet den Label-Selektor, um Pods zu finden

Wenden Sie den Service an:

```
kubectl apply -f 03-service.yaml
```

Schritt 4: Load Balancing konfigurieren

Sie richten einen Ingress ein, um die Anwendung im Internet verfügbar zu machen.

Erstellen Sie zunächst die `IngressClass`. Erstellen Sie eine Datei namens `04-ingressclass.yaml`:

```
apiVersion: networking.k8s.io/v1
kind: IngressClass
metadata:
  labels:
    app.kubernetes.io/name: LoadBalancerController
  name: alb
spec:
  controller: eks.amazonaws.com/alb
```

> **Note**
>
> EKS Auto Mode erfordert Subnetz-Tags zur Identifizierung öffentlicher und privater Subnetze. Wenn Sie Ihren Cluster mit `eksctl` erstellt haben, verfügen Sie bereits über diese Tags. Weitere Informationen erhalten Sie unter the section called “Subnetze kennzeichnen”.

Erstellen Sie anschließend die Ingress-Ressource. Erstellen Sie eine Datei namens `05-ingress.yaml`:

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  namespace: game-2048
  name: ingress-2048
  annotations:
    alb.ingress.kubernetes.io/scheme: internet-facing
    alb.ingress.kubernetes.io/target-type: ip
spec:
  ingressClassName: alb
  rules:
    - http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: service-2048
                port:
```

```
                number: 80
```

Zentrale Komponenten:

- Erstellt ein internetbasiertes ALB
- Verwendet den IP-Zieltyp für die direkte Pod-Weiterleitung
- Leitet den gesamten Datenverkehr (/) an den Spiele-Service weiter

Wenden Sie die Ingress-Konfigurationen an:

```
kubectl apply -f 04-ingressclass.yaml
kubectl apply -f 05-ingress.yaml
```

Schritt 5: Bereitstellung überprüfen

1. Vergewissern Sie sich, dass alle Pods ausgeführt werden:

   ```
   kubectl get pods -n game-2048
   ```

2. Überprüfen Sie, ob der Service erstellt wurde:

   ```
   kubectl get svc -n game-2048
   ```

3. Rufen Sie den ALB-Endpunkt ab:

   ```
   kubectl get ingress -n game-2048
   ```

Das Feld ADDRESS in der Ingress-Ausgabe zeigt Ihren ALB-Endpunkt an. Warten Sie 2–3 Minuten, bis der ALB alle Ziele bereitgestellt und registriert hat.

Schritt 6: Auf das Spiel zugreifen

Öffnen Sie Ihren Web-Browser und navigieren Sie zur ALB-Endpunkt-URL aus dem vorherigen Schritt. Sie sollten nun die 2048-Spieloberfläche sehen.

Schritt 7: Bereinigen

So entfernen Sie alle in diesem Tutorial erstellten Ressourcen:

```
kubectl delete namespace game-2048
```

Dadurch werden alle Ressourcen im Namespace gelöscht, einschließlich der Bereitstellungs-, Service- und Ingress-Ressourcen.

Was hinter den Kulissen passiert

1. Die Bereitstellung erstellt fünf Pods, in denen das Spiel 2048 ausgeführt wird
2. Der Service bietet einen stabilen Netzwerkzugang zu diesen Pods
3. EKS Auto Mode:
   - Erstellt einen Application Load Balancer in AWS
   - Konfiguriert Zielgruppen für die Pods
   - Richtet Routing-Regeln ein, um den Datenverkehr zum Service zu leiten

Fehlersuche

Falls das Spiel nicht geladen wird:

- Stellen Sie sicher, dass alle Pods ausgeführt werden: `kubectl get pods -n game-2048`
- Überprüfen Sie den Ingress-Status: `kubectl describe ingress -n game-2048`
- Überprüfen Sie die ALB-Zustandsprüfungen: Zustand der Zielgruppe in der AWS-Konsole überprüfen

Bereitstellung einer Beispiel-Workload mit Statusverwaltung in EKS Auto Mode

Dieses Tutorial führt Sie durch die Bereitstellung einer Beispielanwendung mit Statusverwaltung in Ihrem Cluster von EKS Auto Mode. Die Anwendung schreibt Zeitstempel in ein persistentes Volume und demonstriert damit die automatische EBS-Volume-Bereitstellung und die Persistenzfunktionen von EKS Auto Mode.

Voraussetzungen

- Ein EKS-Auto-Mode-Cluster
- Die AWS CLI ist mit entsprechenden Berechtigungen konfiguriert

- `kubectl` installiert und konfiguriert
  - Weitere Informationen finden Sie unter Einrichten.

Schritt 1: Umgebung konfigurieren

1. Legen Sie Ihre Umgebungsvariablen fest:

```
export CLUSTER_NAME=my-auto-cluster
export AWS_REGION="us-west-2"
```

2. Aktualisieren Sie Ihre kubeconfig:

```
aws eks update-kubeconfig --name "${CLUSTER_NAME}"
```

Schritt 2: Speicherklasse erstellen

`StorageClass` definiert, wie der EKS Auto Mode EBS-Volumes bereitstellt.

EKS Auto Mode erstellt keine `StorageClass` für Sie. Sie müssen eine `StorageClass`-Referenzierung `ebs.csi.eks.amazonaws.com` erstellen, um die Speicherfunktion von EKS Auto Mode zu nutzen.

1. Erstellen Sie eine Datei namens `storage-class.yaml`:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: auto-ebs-sc
  annotations:
    storageclass.kubernetes.io/is-default-class: "true"
provisioner: ebs.csi.eks.amazonaws.com
volumeBindingMode: WaitForFirstConsumer
parameters:
  type: gp3
  encrypted: "true"
```

2. Wenden Sie `StorageClass` an:

```
kubectl apply -f storage-class.yaml
```

Zentrale Komponenten:

- `provisioner: ebs.csi.eks.amazonaws.com` – Verwendet EKS Auto Mode
- `volumeBindingMode: WaitForFirstConsumer` – Verzögert die Volume-Erstellung, bis ein Pod sie benötigt
- `type: gp3` – Gibt den EBS-Volume-Typ an
- `encrypted: "true"` – EBS verwendet den Standardschlüssel `aws/ebs` zum Verschlüsseln von Volumes, die mit dieser Klasse erstellt wurden. Dies ist zwar optional, wird aber empfohlen.
- `storageclass.kubernetes.io/is-default-class: "true"` – Kubernetes verwendet standardmäßig diese Speicherklasse, sofern Sie nicht für einen persistenten Volume-Anspruch eine andere Volume-Klasse angeben. Seien Sie beim Festlegen dieses Werts vorsichtig, wenn Sie von einem anderen Speicher-Controller migrieren. (optional)

Schritt 3: Persistenten Volume-Anspruch erstellen

Der PVC fordert Speicher von der `StorageClass` an.

1. Erstellen Sie eine Datei namens `pvc.yaml`:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: auto-ebs-claim
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: auto-ebs-sc
  resources:
    requests:
      storage: 8Gi
```

2. Wenden Sie den PVC an:

```
kubectl apply -f pvc.yaml
```

Zentrale Komponenten:

- `accessModes: ReadWriteOnce` – Das Volume kann jeweils von einem Knoten eingebunden werden
- `storage: 8Gi` – Fordert ein Volumen von 8 GiB an
- `storageClassName: auto-ebs-sc` – Verweist auf die von uns erstellte `StorageClass`

Schritt 4: Anwendung bereitstellen

Die Bereitstellung führt einen Container aus, der Zeitstempel in das persistente Volume schreibt.

1. Erstellen Sie eine Datei namens `deployment.yaml`:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: inflate-stateful
spec:
  replicas: 1
  selector:
    matchLabels:
      app: inflate-stateful
  template:
    metadata:
      labels:
        app: inflate-stateful
    spec:
      terminationGracePeriodSeconds: 0
      nodeSelector:
        eks.amazonaws.com/compute-type: auto
      containers:
        - name: bash
          image: public.ecr.aws/docker/library/bash:4.4
          command: ["/usr/local/bin/bash"]
          args: ["-c", "while true; do echo $(date -u) >> /data/out.txt; sleep 60;
 done"]
          resources:
            requests:
              cpu: "1"
          volumeMounts:
            - name: persistent-storage
              mountPath: /data
      volumes:
```

```
        - name: persistent-storage
          persistentVolumeClaim:
            claimName: auto-ebs-claim
```

2. Bereitstellung durchführen:

```
kubectl apply -f deployment.yaml
```

Zentrale Komponenten:

- Einfacher Bash-Container, der Zeitstempel in eine Datei schreibt
- Bindet den PVC bei `/data` ein
- Fordert 1 CPU-Kern an
- Verwendet Knotenselektor für EKS-verwaltete Knoten

Schritt 5: Einrichtung überprüfen

1. Überprüfen Sie, ob der Pod ausgeführt wird:

```
kubectl get pods -l app=inflate-stateful
```

2. Überprüfen Sie, ob der PVC eingebunden ist:

```
kubectl get pvc auto-ebs-claim
```

3. Überprüfen Sie das EBS-Volume:

```
# Get the PV name
PV_NAME=$(kubectl get pvc auto-ebs-claim -o jsonpath='{.spec.volumeName}')
# Describe the EBS volume
aws ec2 describe-volumes \
  --filters Name=tag:CSIVolumeName,Values=${PV_NAME}
```

4. Überprüfen Sie, ob Daten geschrieben werden:

```
kubectl exec "$(kubectl get pods -l app=inflate-stateful \
  -o=jsonpath='{.items[0].metadata.name}')" -- \
  cat /data/out.txt
```

Schritt 6: Bereinigen

Führen Sie den folgenden Befehl aus, um alle in diesem Tutorial erstellten Ressourcen zu entfernen:

```
# Delete all resources in one command
kubectl delete deployment/inflate-stateful pvc/auto-ebs-claim storageclass/auto-ebs-sc
```

Was hinter den Kulissen passiert

1. Der PVC fordert Speicher von der `StorageClass` an
2. Wenn der Pod geplant ist:
   a. EKS Auto Mode stellt ein EBS-Volume bereit
   b. Erstellt ein PersistentVolume
   c. Fügt das Volume zum Knoten hinzu
3. Der Pod bindet das Volume ein und beginnt mit dem Schreiben von Zeitstempeln

Snapshot-Controller

EKS Auto Mode ist mit dem Kubernetes CSI Snapshotter, auch bekannt als Snapshot-Controller, kompatibel. EKS Auto Mode umfasst jedoch keinen Snapshot-Controller. Sie sind für die Installation und Konfiguration des Snapshot-Controllers verantwortlich. Weitere Informationen finden Sie unter the section called “CSI-Snapshot-Controller”.

Überprüfen Sie die folgenden `VolumeSnapshotClass`, die sich auf die Speicherkapazität von EKS Auto Mode beziehen.

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: auto-ebs-vsclass
driver: ebs.csi.eks.amazonaws.com
deletionPolicy: Delete
```

Weitere Informationen zum Kubernetes CSI Snapshotter.

Einen beschleunigten Workload bereitstellen

Dieses Tutorial zeigt, wie Amazon EKS Auto Mode das Starten hardwarebeschleunigter Workloads vereinfacht. Amazon EKS Auto Mode optimiert den Betrieb über den Cluster hinaus. Dabei werden wichtige Infrastrukturkomponenten automatisiert und Funktionen für Rechenleistung, Netzwerke, Load Balancing, Speicher sowie Identitätszugriff und -verwaltung sofort bereitgestellt.

Amazon EKS Auto Mode umfasst die Treiber und Geräte-Plug-ins, die für bestimmte Instance-Typen wie NVIDIA- und AWS Neuron-Treiber erforderlich sind. Sie müssen diese Komponenten nicht installieren oder aktualisieren.

EKS Auto Mode verwaltet automatisch die Treiber für folgende Beschleuniger:

- AWS Trainium
- AWS Inferenz
- EC2 Beschleunigte Instances von NVIDIA GPUs auf Amazon

> **Note**
>
> EKS Auto Mode beinhaltet das NVIDIA-Geräte-Plugin für Kubernetes. Dieses Plugin wird automatisch ausgeführt und ist in Ihrem Cluster nicht als Daemon-Set sichtbar.

Zusätzliche Netzwerkunterstützung:

- Elastic Fabric Adapter (EFA)

Amazon EKS Auto Mode macht die aufwändige Verwaltung von Beschleunigertreibern und Geräte-Plugins überflüssig.

Sie können außerdem von Kosteneinsparungen profitieren, indem Sie den Cluster auf Null skalieren. Sie können EKS Auto Mode so konfigurieren, dass Instances beendet werden, wenn keine Workloads ausgeführt werden. Dies ist nützlich für Batch-basierte Inferenz-Workloads.

Nachfolgend finden Sie ein Beispiel für das Starten beschleunigter Workloads mit Amazon EKS Auto Mode.

Voraussetzungen

- Ein Kubernetes-Cluster mit konfiguriertem Amazon EKS Auto Mode.
- Eine `default`-EKS-Knotenklasse wird erstellt, wenn die verwalteten Knoten-Pools `general-purpose` oder `system` aktiviert sind.

Schritt 1: GPU-Workload bereitstellen

In diesem Beispiel erstellen Sie eine NodePool für NVIDIA-basierte Workloads, die 45 GB GPU-Speicher benötigen. Mit EKS Auto Mode definieren Sie Ihre Instance-Anforderungen mithilfe von Kubernetes-Planungsbeschränkungen.

Um den Amazon EKS Auto Mode `NodePool` und das Beispiel bereitzustellen`workload`, überprüfen Sie Folgendes NodePool und die Pod-Definition und speichern Sie unter `nodepool-gpu.yaml` und`pod.yaml`:

nodepool-gpu.yaml

```
apiVersion: karpenter.sh/v1
kind: NodePool
metadata:
  name: gpu
spec:
  disruption:
    budgets:
    - nodes: 10%
    consolidateAfter: 1h
    consolidationPolicy: WhenEmpty
  template:
    metadata: {}
    spec:
      nodeClassRef:
        group: eks.amazonaws.com
        kind: NodeClass
        name: default
      requirements:
        - key: "karpenter.sh/capacity-type"
          operator: In
          values: ["on-demand"]
        - key: "kubernetes.io/arch"
          operator: In
```

```
          values: ["amd64"]
        - key: "eks.amazonaws.com/instance-family"
          operator: In
          values:
          - g6e
          - g6
      taints:
        - key: nvidia.com/gpu
          effect: NoSchedule
      terminationGracePeriod: 24h0m0s
```

pod.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: nvidia-smi
spec:
  nodeSelector:
    eks.amazonaws.com/compute-type: auto
  restartPolicy: OnFailure
  containers:
  - name: nvidia-smi
    image: public.ecr.aws/amazonlinux/amazonlinux:2023-minimal
    args:
    - "nvidia-smi"
    resources:
      requests:
        memory: "30Gi"
        cpu: "3500m"
        nvidia.com/gpu: 1
      limits:
        memory: "30Gi"
        nvidia.com/gpu: 1
  tolerations:
  - key: nvidia.com/gpu
    effect: NoSchedule
    operator: Exists
```

Beachten Sie, dass für den `eks.amazonaws.com/compute-type: auto`-Selektor die Bereitstellung des Workloads in einem Knoten von Amazon EKS Auto Mode erforderlich ist. Das stellt NodePool auch einen Makel dar, der es erlaubt, nur Pods mit Toleranzen für Nvidia GPUs zu planen.

Wenden Sie die Arbeitslast NodePool und den Workload auf Ihren Cluster an.

```
kubectl apply -f nodepool-gpu.yaml
kubectl apply -f pod.yaml
```

Die Ausgabe sollte folgendermaßen aussehen:

```
nodepool.karpenter.sh/gpu configured created
pod/nvidia-smi created
```

Warten Sie einige Sekunden und überprüfen Sie die Knoten in Ihrem Cluster. Sie sollten nun einen neuen Knoten in Ihrem Cluster von Amazon EKS Auto Mode sehen:

```
> kubectl get nodes

NAME        TYPE           CAPACITY    ZONE         NODE                   READY   AGE
gpu-dnknr   g6e.2xlarge    on-demand   us-west-2b   i-02315c7d7643cdee6   True    76s
```

Schritt 2: Validieren

Sie können sehen, dass Amazon EKS Auto Mode ein `g6e.2xlarge` statt eines `g6.2xlarge` gestartet hat, da die Workload gemäß den folgenden Kubernetes-Planungsbeschränkungen eine Instance mit l40s GPU erforderte:

```
...
  nodeSelector:
    eks.amazonaws.com/instance-gpu-name: l40s
...
    requests:
        memory: "30Gi"
        cpu: "3500m"
        nvidia.com/gpu: 1
      limits:
        memory: "30Gi"
        nvidia.com/gpu: 1
```

Überprüfen Sie nun die Container-Protokolle, indem Sie den folgenden Befehl ausführen:

```
kubectl logs nvidia-smi
```

Beispielausgabe:

```
+-----------------------------------------------------------------------------------------
+
| NVIDIA-SMI 535.230.02             Driver Version: 535.230.02   CUDA Version: 12.2
 |
|-----------------------------------------+----------------------
+----------------------+
| GPU  Name                 Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC
 |
| Fan  Temp   Perf          Pwr:Usage/Cap |         Memory-Usage | GPU-Util  Compute M.
 |
|                                         |                      |               MIG M.
 |
|=========================================+======================
+======================|
|   0  NVIDIA L40S                    On  | 00000000:30:00.0 Off |                    0
 |
| N/A   27C    P8              23W / 350W |      0MiB / 46068MiB |      0%      Default
 |
|                                         |                      |                  N/A
 |
+-----------------------------------------+----------------------
+----------------------+

+-----------------------------------------------------------------------------------------
+
| Processes:
 |
|  GPU   GI   CI        PID   Type   Process name                            GPU Memory
 |
|        ID   ID                                                             Usage
 |
|
=========================================================================================|
|  No running processes found
 |
+-----------------------------------------------------------------------------------------
+
```

Sehen Sie sich nun die Container-Protokolle an, indem Sie den folgenden Befehl ausführen: Sie können sehen, dass der Container erkannt hat, dass er in einer Instance mit einer `NVIDIA` GPU

ausgeführt wird, und dass Sie keine Gerätetreiber installieren mussten, da dies von Amazon EKS Auto Mode verwaltet wird.

Schritt 3: Bereinigen

Um alle erstellten Objekte zu entfernen, verwenden Sie, `kubectl` um die Beispielbereitstellung zu löschen, NodePool sodass der Knoten beendet wird:

```
kubectl delete -f nodepool-gpu.yaml
kubectl delete -f pod.yaml
```

NodePools Beispielreferenz

Erstellen Sie eine NVIDIA NodePool

Im Folgenden wird Folgendes NodePool definiert:

- Nur Instances der `g6e`- und `g6`-Familie starten
- Knoten konsolidieren, wenn sie 1 Stunde lang leer sind
  - Der Wert von 1 Stunde für `consolodateAfter` unterstützt spitzenlastige Workloads und reduziert die Fluktuation der Knoten. Sie können `consolidateAfter` basierend auf Ihren Workload-Anforderungen optimieren.

Beispiel NodePool mit GPU-Instanzfamilie und Konsolidierung

```
apiVersion: karpenter.sh/v1
kind: NodePool
metadata:
  name: gpu
spec:
  disruption:
    budgets:
    - nodes: 10%
    consolidateAfter: 1h
    consolidationPolicy: WhenEmpty
  template:
    metadata: {}
    spec:
      nodeClassRef:
        group: eks.amazonaws.com
```

```
        kind: NodeClass
        name: default
      requirements:
        - key: "karpenter.sh/capacity-type"
          operator: In
          values: ["on-demand"]
        - key: "kubernetes.io/arch"
          operator: In
          values: ["amd64"]
        - key: "eks.amazonaws.com/instance-family"
          operator: In
          values:
          - g6e
          - g6
      terminationGracePeriod: 24h0m0s
```

Anstatt `eks.amazonaws.com/instance-gpu-name` festzulegen, können Sie `eks.amazonaws.com/instance-family` verwenden, um die Instance-Familie anzugeben. Weitere bekannte Labels, welche die Überprüfung der Zeitplanung beeinflussen, finden Sie unter the section called “Unterstützte Labels für EKS Auto Mode”.

Wenn Sie spezielle Speicheranforderungen haben, können Sie den kurzlebigen Speicher der Knoten anpassen `size` und `throughput` Ihren eigenen Speicher erstellen`iops`, auf den Sie NodeClassverweisen können. NodePool Erfahren Sie mehr über die konfigurierbaren NodeClass Optionen.

Beispiel für eine Speicherkonfiguration für NodeClass

```
apiVersion: eks.amazonaws.com/v1
kind: NodeClass
metadata:
  name: gpu
spec:
  ephemeralStorage:
    iops: 3000
    size: 80Gi
    throughput: 125
```

Definieren Sie ein AWS Trainium und Inferentia AWS NodePool

Im Folgenden NodePool ist `eks.amazonaws.com/instance-category` festgelegt, dass nur Instanzen der Inferentia- und Trainium-Familie gestartet werden sollen:

```
        - key: "eks.amazonaws.com/instance-category"
          operator: In
          values:
            - inf
            - trn
```

EKS-Auto-Mode-Einstellungen konfigurieren

In diesem Kapitel wird beschrieben, wie Sie bestimmte Aspekte Ihrer Amazon Elastic Kubernetes Service (EKS)-Auto-Mode-Cluster konfigurieren. Während EKS Auto Mode die meisten Infrastrukturkomponenten automatisch verwaltet, können Sie bestimmte Features an Ihre Workload-Anforderungen anpassen.

Mithilfe der in diesem Thema beschriebenen Konfigurationsoptionen können Sie Netzwerkeinstellungen, Rechenressourcen und Lastausgleichsverhalten ändern und gleichzeitig die Vorteile der automatisierten Infrastrukturverwaltung beibehalten. Bevor Sie Konfigurationsänderungen vornehmen, überprüfen Sie die verfügbaren Optionen in den folgenden Abschnitten, um zu bestimmen, welcher Ansatz Ihren Anforderungen am besten entspricht.

Welche Features möchten Sie konfigurieren?	Konfigurationsoption
Knotenvernetzung und -speicherung • Knotenplatzierung über öffentliche und private Subnetze hinweg konfigurieren • Benutzerdefinierte Sicherheitsgruppen für die Knoten-Zugriffskontrolle definieren • Richtlinien für die Netzwerk-Adressübersetzung (SNAT) anpassen • Kubernetes-Netzwerkrichtlinien, detaillierte Protokollierung und Überwachung von Netzwerkrichtlinien aktivieren • Parameter für flüchtigen Speicher (Größe, IOPS, Durchsatz) festlegen • Verschlüsselten flüchtigen Speicher mit benutzerdefinierten KMS-Schlüsseln konfigurieren	the section called “Knotenklasse erstellen”

Welche Features möchten Sie konfigurieren?	Konfigurationsoption
• Pod-Datenverkehr in separaten Subnetzen von den Knoten isolieren	
Knoten-Rechenressourcen • Wählen Sie bestimmte EC2 Instance-Typen und Familien aus • Definieren Sie CPU-Architekturen (x86_64,) ARM64 • Kapazitätstypen konfigurieren (On-Demand, Spot) • Availability Zones angeben • Knoten-Taints und -Labels konfigurieren • Ressourcenlimits für die CPU- und Speichern utzung festlegen	the section called “Knoten-Pool erstellen”
Knotenpools mit statischer Kapazität • Behalten Sie unabhängig vom Pod-Bedarf eine feste Anzahl von Knoten bei • Konfigurieren Sie vorhersehbare Kapazitäten für Reserved Instances • Legen Sie eine bestimmte Anzahl von Replikaten fest, um eine konsistente Infrastru kturnutzung zu gewährleisten • Skalieren Sie statische Knotenpools manuell • Überwachen Sie den Status des Knotenpools mit statischer Kapazität	the section called “Knotenpools mit statischer Kapazität”

Welche Features möchten Sie konfigurieren?	Konfigurationsoption
Einstellungen für Application Load Balancer • Interne oder internetbasierte Load Balancer bereitstellen • Konfigurieren des zonenübergreifenden Load Balancing • Zeitüberschreitungen für Leerlauf festlegen • Aktivieren Sie HTTP/2 und unterstützen Sie WebSocket • Parameter für Zustandsprüfung konfigurieren • TLS-Zertifikateinstellungen festlegen • Zielgruppenattribute definieren • Stellen Sie den IP-Adresstyp ein (IPv4, Dual-Stack)	the section called “Ingress-Klasse erstellen”
Einstellungen für Network Load Balancer • Direktes Pod-IP-Routing konfigurieren • Aktivieren von zonenübergreifendem Load Balancing • Zeitlimit für inaktive Verbindung festlegen • Parameter für Zustandsprüfung konfigurieren • Subnetz-Platzierung festlegen • Stellen Sie den IP-Adresstyp ein (IPv4, Dual-Stack) • Beibehaltung der Quell-IP des Clients konfigurieren • Zielgruppenattribute definieren	the section called “Service erstellen”

Welche Features möchten Sie konfigurieren?	Konfigurationsoption
Einstellungen für Speicherklasse • EBS-Volume-Typen (gp3, io1, io2 usw.) definieren • Volume-Verschlüsselung und KMS-Schlüsselverwendung konfigurieren • IOPS- und Durchsatz-Parameter festlegen • Als Standard-Speicherklasse festlegen • Benutzerdefinierte Tags für bereitgestellte Volumes definieren	the section called “Erstellen StorageClass”
ODCR-Nutzung steuern • Konfigurieren Sie die Workload-Bereitstellung für EC2 Kapazitätsreservierungen auf Abruf • Wählen Sie für die gezielte Kapazitätsnutzung explizit eine bestimmte ODCRs ID aus • Verwenden Sie die Tag-basierte Auswahl für folgende Zielgruppen ODCRs • ODCRs Filtern Sie für kontoübergreifende Szenarien nach dem eigenen AWS Konto • Steuern Sie, ob Workloads automatisch „Öffnen“ verwenden ODCRs	the section called “Steuerung von Kapazitätsreservierungen”
Erweiterte Sicherheit für Knoten • Aktivieren Sie nach FIPS 140-2 validierte kryptografische Module • Konfigurieren Sie den Kernel-Lockdown-Sicherheitsmodus	the section called “Erweiterte Sicherheit”

Knotenklasse für Amazon EKS erstellen

Amazon-EKS-Knotenklassen sind Vorlagen, die eine differenzierte Steuerung der Konfiguration Ihrer im EKS Auto Mode verwalteten Knoten ermöglichen. Eine Knotenklasse definiert Einstellungen auf Infrastrukturebene, die für Knotengruppen in Ihrem EKS-Cluster gelten, einschließlich Netzwerkkonfiguration, Speichereinstellungen und Ressourcen-Kennzeichnung. In diesem Thema wird erläutert, wie Sie eine Knotenklasse erstellen und konfigurieren, um Ihre spezifischen Betriebsanforderungen zu erfüllen.

Wenn Sie anpassen müssen, wie EKS Auto Mode EC2 Instanzen über die Standardeinstellungen hinaus bereitstellt und konfiguriert, bietet Ihnen die Erstellung einer Node-Klasse eine präzise Kontrolle über kritische Infrastrukturparameter. Beispielsweise können Sie zur Erhöhung der Sicherheit die Platzierung in einem privaten Subnetz festlegen, für leistungssensitive Workloads flüchtigen Instance-Speicher konfigurieren oder zur Kostenverteilung benutzerdefinierte Tags zuweisen.

Erstellung einer Knotenklasse

Um eine `NodeClass` zu erstellen, gehen Sie wie folgt vor:

1. YAML-Datei (zum Beispiel `nodeclass.yaml`) mit Ihrer Knotenklassen-Konfiguration erstellen
2. Konfiguration mithilfe von `kubectl` auf Ihren Cluster anwenden
3. Verweisen Sie in Ihrer Knoten-Pool-Konfiguration auf die Knotenklasse. Weitere Informationen finden Sie unter the section called “Knoten-Pool erstellen”.

Sie müssen `kubectl` installiert und konfiguriert haben. Weitere Informationen finden Sie unter Einrichten.

Beispiel für eine grundlegende Knotenklasse

Hier ist ein Beispiel für eine Knotenklasse:

```
apiVersion: eks.amazonaws.com/v1
kind: NodeClass
metadata:
  name: private-compute
spec:
  subnetSelectorTerms:
    - tags:
        Name: "private-subnet"
```

```
        kubernetes.io/role/internal-elb: "1"
  securityGroupSelectorTerms:
    - tags:
        Name: "eks-cluster-sg"
  ephemeralStorage:
    size: "160Gi"
```

Dadurch wird die Menge an flüchtigem Speicher auf dem Knoten NodeClass erhöht.

Wenden Sie diese Konfiguration wie folgt an

```
kubectl apply -f nodeclass.yaml
```

Verweisen Sie anschließend in Ihrer Knoten-Pool-Konfiguration auf die Knotenklasse. Weitere Informationen finden Sie unter the section called “Knoten-Pool erstellen”.

Knotenklassen-Zugriffseintrag erstellen

Wenn Sie eine benutzerdefinierte Knotenklasse erstellen, müssen Sie einen EKS-Zugriffseintrag erstellen, damit die Knoten dem Cluster beitreten können. EKS erstellt automatisch Zugriffseinträge, wenn Sie die integrierte Knotenklasse und Knoten-Pools verwenden.

Informationen zur Funktionsweise von Zugriffseinträgen finden Sie unter the section called “Zugriffseinträge”

Beim Erstellen von Zugriffseinträgen für EKS-Auto-Mode-Knotenklassen müssen Sie den `EC2`-Zugriffseintragstyp verwenden.

Zugriffseintrag mit der CLI erstellen

Gehen Sie wie folgt vor, um einen Zugriffseintrag für EC2 Knoten zu erstellen und die EKS Auto Node Policy zuzuordnen:

Aktualisieren Sie die folgenden CLI-Befehle mit Ihrem Cluster-Namen und der Knotenrollen-ARN. Die Knotenrollen-ARN ist in der Knotenklasse YAML angegeben.

```
# Create the access entry for EC2 nodes
aws eks create-access-entry \
  --cluster-name <cluster-name> \
  --principal-arn <node-role-arn> \
  --type EC2

# Associate the auto node policy
```

```
aws eks associate-access-policy \
  --cluster-name <cluster-name> \
  --principal-arn <node-role-arn> \
  --policy-arn arn:aws: eks::aws:cluster-access-policy/AmazonEKSAutoNodePolicy \
  --access-scope type=cluster
```

Erstellen Sie einen Zugriffseintrag mit CloudFormation

Um einen Zugriffseintrag für EC2 Knoten zu erstellen und die EKS Auto Node Policy zuzuordnen:

Aktualisieren Sie Folgendes CloudFormation mit Ihrem Clusternamen und Ihrem Knotenrollen-ARN. Die Knotenrollen-ARN ist in der Knotenklasse YAML angegeben.

```
EKSAutoNodeRoleAccessEntry:
  Type: AWS::EKS::AccessEntry
  Properties:
    ClusterName: <cluster-name>
    PrincipalArn: <node-role-arn>
    Type: "EC2"
    AccessPolicies:
      - AccessScope:
          Type: cluster
        PolicyArn: arn:aws: eks::aws:cluster-access-policy/AmazonEKSAutoNodePolicy
  DependsOn: [ <cluster-name> ] # previously defined in CloudFormation
```

Informationen zur Bereitstellung von CloudFormation Stacks finden Sie unter Erste Schritte mit CloudFormation

Knotenklassen-Spezifikation

```
apiVersion: eks.amazonaws.com/v1
kind: NodeClass
metadata:
  name: my-node-class
spec:
  # Required fields

  # role and instanceProfile are mutually exclusive fields.
  role: MyNodeRole  # IAM role for EC2 instances
  # NOTE: instance profile names must start with the 'eks' prefix. this is a
  # temporary limitation to be lifted soon.
  # instanceProfile: eks-MyNodeInstanceProfile  # IAM instance-profile for EC2
 instances
```

```
  subnetSelectorTerms:
    - tags:
        Name: "private-subnet"
        kubernetes.io/role/internal-elb: "1"
    # Alternative using direct subnet ID
    # - id: "subnet-0123456789abcdef0"

  securityGroupSelectorTerms:
    - tags:
        Name: "eks-cluster-sg"
    # Alternative approaches:
    # - id: "sg-0123456789abcdef0"
    # - name: "eks-cluster-security-group"

  # Optional: Pod subnet selector for advanced networking
  podSubnetSelectorTerms:
    - tags:
        Name: "pod-subnet"
        kubernetes.io/role/pod: "1"
    # Alternative using direct subnet ID
    # - id: "subnet-0987654321fedcba0"
# must include Pod security group selector also
 podSecurityGroupSelectorTerms:
  - tags:
      Name: "eks-pod-sg"
    # Alternative using direct security group ID
    # - id: "sg-0123456789abcdef0"

  # Optional: Selects on-demand capacity reservations and capacity blocks
  # for EKS Auto Mode to prioritize.
  capacityReservationSelectorTerms:
    - id: cr-56fac701cc1951b03
    # Alternative Approaches
    - tags:
        Name: "targeted-odcr"
      # Optional owning account ID filter
      owner: "012345678901"

  # Optional fields
  snatPolicy: Random  # or Disabled

  networkPolicy: DefaultAllow  # or DefaultDeny
  networkPolicyEventLogs: Disabled  # or Enabled
```

```
  ephemeralStorage:
    size: "80Gi"    # Range: 1-59000Gi or 1-64000G or 1-58Ti or 1-64T
    iops: 3000      # Range: 3000-16000
    throughput: 125 # Range: 125-1000
    # Optional KMS key for encryption
    kmsKeyID: "arn:aws: kms:region:account:key/key-id"
    # Accepted formats:
    # KMS Key ID
    # KMS Key ARN
    # Key Alias Name
    # Key Alias ARN

  advancedNetworking:
    # Optional: Controls whether public IP addresses are assigned to instances that are
 launched with the nodeclass.
    # If not set, defaults to the MapPublicIpOnLaunch setting on the subnet.
    associatePublicIPAddress: false

    # Optional: Forward proxy, commonly requires certificateBundles as well
    # for EC2, see https://repost.aws/knowledge-center/eks-http-proxy-containerd-
automation
    httpsProxy: http://192.0.2.4:3128 #commonly port 3128 (Squid) or 8080 (NGINX) #Max
 255 characters
    #httpsProxy: http://[2001:db8::4]:3128 # IPv6 address with port, use []
    noProxy: #Max 50 entries
        - localhost #Max 255 characters each
        - 127.0.0.1
        #- ::1 # IPv6 localhost
        #- 0:0:0:0:0:0:0:1 # IPv6 localhost
        - 169.254.169.254 # EC2 Instance Metadata Service
        #- [fd00:ec2::254] # IPv6 EC2 Instance Metadata Service
        # Domains to exclude, put all VPC endpoints here
        - .internal
        - .eks.amazonaws.com
    # ipv4PrefixSize is default to Auto which is prefix and fallback to secondary IP.
 "32" is the secondary IP mode.
    ipv4PrefixSize: Auto # or "32"

  advancedSecurity:
    # Optional, US regions only: Specifying `fips: true` will cause nodes in the
 nodeclass to run FIPS compatible AMIs.
    fips: false
```

```
  # Optional: Custom certificate bundles.
  certificateBundles:
    - name: "custom-cert"
      data: "base64-encoded-cert-data"

  # Optional: Additional EC2 tags (with restrictions)
  tags:
    Environment: "production"
    Team: "platform"
    # Note: Cannot use restricted tags like:
    # - kubernetes.io/cluster/*
    # - karpenter.sh/provisioner-name
    # - karpenter.sh/nodepool
    # - karpenter.sh/nodeclaim
    # - karpenter.sh/managed-by
    # - eks.amazonaws.com/nodeclass
```

Überlegungen

- Wenn Sie überprüfen möchten, über wie viel lokalen Speicherplatz eine Instance verfügt, können Sie den Knoten beschreiben, um die flüchtige Speicher-Ressource anzuzeigen.
- Volumenverschlüsselung — EKS verwendet den konfigurierten benutzerdefinierten KMS-Schlüssel, um das schreibgeschützte Root-Volume der Instanz und das Datenvolumen zu verschlüsseln. read/write
- IAM-Rolle des Knotens ersetzen – Wenn Sie die einer `NodeClass` zugeordneten IAM-Rolle ändern, müssen Sie einen neuen Zugriffseintrag erstellen. EKS erstellt während der Cluster-Erstellung automatisch einen Zugriffseintrag für die IAM-Rolle des Knotens. Die IAM-Rolle des Knotens erfordert die `AmazonEKSAutoNodePolicy`-EKS-Zugriffsrichtlinie. Weitere Informationen finden Sie unter the section called “Zugriffseinträge”.
- Maximale Pod-Dichte – EKS begrenzt die maximale Anzahl von Pods auf einem Knoten auf 110. Diese Begrenzung wird nach der vorhandenen Berechnung der maximalen Pods angewendet. Weitere Informationen finden Sie unter the section called “Amazon EC2-Instance-Typen”.
- Tags — Wenn Sie Tags von Kubernetes an weitergeben möchten EC2, müssen Sie zusätzliche IAM-Berechtigungen konfigurieren. Weitere Informationen finden Sie unter the section called “Identität und Zugriff”.
- Standard-Knotenklasse – Benennen Sie Ihre benutzerdefinierte Knotenklasse `default` nicht. Dies liegt daran, dass EKS Auto Mode eine `NodeClass` mit dem Namen `default` enthält, die automatisch bereitgestellt wird, wenn Sie mindestens ein integriertes `NodePool` aktivieren.

Informationen zur Aktivierung des integrierten `NodePools` finden Sie unter the section called “Integrierte Knoten-Pool überprüfen”.

- **`subnetSelectorTerms`**-Verhalten bei mehreren Subnetzen – Wenn mehrere Subnetze vorhanden sind, die den `subnetSelectorTerms`-Bedingungen entsprechen oder die Sie anhand der ID angeben, erstellt der EKS Auto Mode Knoten, die über die Subnetze verteilt sind.
  - Wenn sich die Subnetze in unterschiedlichen Availability Zones (AZs) befinden, können Sie Kubernetes-Funktionen wie Spread-Beschränkungen für die Pod-Topologie und Topology Aware Routing verwenden, um Pods bzw. Traffic über die Zonen zu verteilen.
  - Wenn es in derselben AZ mehrere Subnetze gibt, die mit `subnetSelectorTerms` übereinstimmen, erstellt EKS Auto Mode Pods auf jedem Knoten, die über die Subnetze in dieser AZ verteilt sind. EKS Auto Mode erstellt sekundäre Netzwerkschnittstellen auf jedem Knoten in den anderen Subnetzen derselben AZ. Die Auswahl erfolgt anhand der Anzahl der verfügbaren IP-Adressen in jedem Subnetz, um die Subnetze effizienter zu nutzen. Sie können jedoch nicht festlegen, welches Subnetz EKS Auto Mode für jeden Pod verwendet. Wenn Sie möchten, dass Pods in bestimmten Subnetzen ausgeführt werden, verwenden Sie stattdessen the section called “Subnetz-Auswahl für Pods”.

Subnetz-Auswahl für Pods

Die `podSubnetSelectorTerms`- und `podSecurityGroupSelectorTerms`-Felder und ermöglichen erweiterte Netzwerkkonfigurationen, indem sie es Pods ermöglichen, in anderen Subnetzen als ihren Knoten ausgeführt zu werden. Diese Trennung bietet eine verbesserte Kontrolle über das Routing des Netzwerkverkehrs und die Sicherheitsrichtlinien. Beachten Sie, dass `podSecurityGroupSelectorTerms` zusammen mit `podSubnetSelectorTerms` erforderlich sind.

Anwendungsfälle

Verwenden Sie `podSubnetSelectorTerms`, wenn Sie Folgendes benötigen:

- Trennen Sie den Infrastrukturverkehr (node-to-node Kommunikation) vom Anwendungsverkehr (Kommunikation) Pod-to-Pod
- Wenden Sie auf Knoten-Subnetze andere Netzwerkkonfigurationen an als auf Pod-Subnetze.
- Implementieren Sie unterschiedliche Sicherheitsrichtlinien oder Routing-Regeln für Knoten und Pods.

- Konfigurieren Sie Reverse-Proxys oder Netzwerkfilter speziell für den Knoten-Datenverkehr, ohne den Pod-Datenverkehr zu beeinträchtigen. Verwenden Sie `advancedNetworking` und `certificateBundles`, um Ihren Reverse-Proxy und alle selbstsignierten oder privaten Zertifikate für den Proxy zu definieren.

Beispielkonfiguration

```
apiVersion: eks.amazonaws.com/v1
kind: NodeClass
metadata:
  name: advanced-networking
spec:
  role: MyNodeRole

  # Subnets for EC2 instances (nodes)
  subnetSelectorTerms:
    - tags:
        Name: "node-subnet"
        kubernetes.io/role/internal-elb: "1"

  securityGroupSelectorTerms:
    - tags:
        Name: "eks-cluster-sg"

  # Separate subnets for Pods
  podSubnetSelectorTerms:
    - tags:
        Name: "pod-subnet"
        kubernetes.io/role/pod: "1"

  podSecurityGroupSelectorTerms:
  - tags:
      Name: "eks-pod-sg"
```

Überlegungen zu Subnetz-Selektoren für Pods

- Reduzierte Pod-Dichte: Bei Verwendung von `podSubnetSelectorTerms` können weniger Pods auf jedem Knoten ausgeführt werden, da sich die primäre Netzwerkschnittstelle des Knotens im Knoten-Subnetz befindet und nicht für Pods im Pod-Subnetz verwendet werden kann
- Einschränkungen des Subnetz-Selektors: Die Standardkonfigurationen `subnetSelectorTerms` und `securityGroupSelectorTerms` gelten nicht für die Auswahl des Pod-Subnetzes.

- Netzwerk-Planung: Stellen Sie sicher, dass in den Knoten- und Pod-Subnetzen ausreichend IP-Adressraum vorhanden ist, um Ihre Workload-Anforderungen zu erfüllen.
- Routing-Konfiguration: Stellen Sie sicher, dass die Routing-Tabelle und die Netzwerkzugriffssteuerungsliste (ACL) der Pod-Subnetze ordnungsgemäß für die Kommunikation zwischen Knoten- und Pod-Subnetzen konfiguriert sind.
- Availability Zones: Stellen Sie sicher, dass Sie Pod-Subnetze für mehrere AZs erstellt haben. Wenn Sie ein bestimmtes Pod-Subnetz verwenden, muss es sich in derselben AZ wie die Knoten-Subnetz-AZ befinden

Sekundärer IP-Modus für Pods

Die `ipv4PrefixSize` Felder ermöglichen erweiterte Netzwerkkonfigurationen, da Knoten nur sekundäre IP-Adressen zugewiesen werden können. Diese Funktion weist Knoten kein Präfix (/28) zu und behält nur eine sekundäre IP als Minimalwert bei. IPTarget

Anwendungsfälle

Verwenden Sie `ipv4PrefixSize`, wenn Sie Folgendes benötigen:

- Reduzierte IP-Auslastung: In jedem Knoten wird nur eine IP-Adresse aufgewärmt.
- Geringere Abwanderungsrate der Pods: Die Geschwindigkeit der Pod-Erstellung ist kein großes Problem.
- Keine Präfixfragmentierung: Die durch das Präfix verursachte Fragmentierung ist ein großes Problem oder verhindert die Verwendung des automatischen Modus.

Beispielkonfiguration

```
apiVersion: eks.amazonaws.com/v1
kind: NodeClass
metadata:
  name: advanced-networking
spec:
  role: MyNodeRole

  advancedNetworking:
    ipv4PrefixSize: "32"
```

Überlegungen zum sekundären IP-Modus

- Geringere Geschwindigkeit bei der Pod-Erstellung: Da nur eine sekundäre IP aufgewärmt ist, benötigt der IPAM-Dienst mehr Zeit, um mehr Pods bereitzustellen IPs .

Einen Knotenpool für EKS Auto Mode erstellen

Amazon-EKS-Knoten-Pools bieten eine flexible Möglichkeit zur Verwaltung von Rechenressourcen in Ihrem Kubernetes-Cluster. Dieses Thema veranschaulicht, wie Sie mit Karpenter, einem Tool zur Bereitstellung von Knoten, das die Cluster-Skalierung und Ressourcenauslastung optimiert, Knoten-Pools erstellen und konfigurieren können. Mit der NodePool-Ressource von Karpenter können Sie spezifische Anforderungen für Ihre Rechenressourcen definieren, darunter Instance-Typen, Availability Zones, Architekturen und Kapazitätstypen.

Die integrierten `system`- und `general-purpose`- Knoten-Pools können nicht geändert werden. Sie können sie lediglich aktivieren oder deaktivieren. Weitere Informationen finden Sie unter the section called “Integrierte Knoten-Pool überprüfen”.

Die NodePool-Spezifikation ermöglicht eine detaillierte Steuerung der Rechenressourcen Ihres EKS-Clusters durch verschiedene unterstützte Labels und Anforderungen. Dazu gehören Optionen zur Festlegung von EC2-Instance-Kategorien, CPU-Konfigurationen, Availability Zones, Architekturen (ARM64/AMD64) und Kapazitätstypen (Spot oder On-Demand). Sie können auch Ressourcenlimits für die CPU- und Speichernutzung festlegen, um sicherzustellen, dass Ihr Cluster innerhalb der erforderlichen Betriebsgrenzen bleibt.

EKS Auto Mode nutzt bekannte Kubernetes-Labels, um konsistente und standardisierte Methoden zur Identifizierung von Knotenmerkmalen bereitzustellen. Diese Labels, wie beispielsweise `topology.kubernetes.io/zone` für Availability Zones und `kubernetes.io/arch` für die CPU-Architektur, entsprechen den gängigen Kubernetes-Konventionen. Darüber hinaus erweitern EKS-spezifische Labels (mit dem Präfix `eks.amazonaws.com/`) diese Funktionalität um AWS-spezifische Attribute wie Instance-Typen, CPU-Hersteller, GPU-Fähigkeiten und Netzwerk-Spezifikationen. Dieses standardisierte Kennzeichnungssystem ermöglicht eine nahtlose Integration in vorhandene Kubernetes-Tools und bietet gleichzeitig eine umfassende AWS-Infrastrukturintegration.

Einen NodePool erstellen

Führen Sie die folgenden Schritte aus, um einen NodePool für Ihren Amazon-EKS-Cluster zu erstellen

1. Erstellen Sie eine YAML-Datei mit dem Namen `nodepool.yaml` und der erforderlichen NodePool-Konfiguration. Sie können die folgende Beispielkonfiguration verwenden.
2. Wenden Sie den NodePool auf Ihren Cluster an:

```
kubectl apply -f nodepool.yaml
```

3. Überprüfen Sie, ob der NodePool erfolgreich erstellt wurde:

```
kubectl get nodepools
```

4. (Optional) Überwachen Sie den NodePool-Status:

```
kubectl describe nodepool default
```

Stellen Sie sicher, dass Ihr NodePool auf eine gültige NodeClass verweist, die in Ihrem Cluster vorhanden ist. Die NodeClass definiert AWS-spezifische Konfigurationen für Ihre Rechenressourcen. Weitere Informationen finden Sie unter the section called “Knotenklasse erstellen”.

Beispiel-NodePool

```
apiVersion: karpenter.sh/v1
kind: NodePool
metadata:
  name: my-node-pool
spec:
  template:
    metadata:
      labels:
        billing-team: my-team
    spec:
      nodeClassRef:
        group: eks.amazonaws.com
        kind: NodeClass
        name: default

      requirements:
        - key: "eks.amazonaws.com/instance-category"
          operator: In
          values: ["c", "m", "r"]
        - key: "eks.amazonaws.com/instance-cpu"
```

```
          operator: In
          values: ["4", "8", "16", "32"]
        - key: "topology.kubernetes.io/zone"
          operator: In
          values: ["us-west-2a", "us-west-2b"]
        - key: "kubernetes.io/arch"
          operator: In
          values: ["arm64", "amd64"]

  limits:
    cpu: "1000"
    memory: 1000Gi
```

Unterstützte Labels für EKS Auto Mode

EKS Auto Mode unterstützt die folgenden bekannten Labels.

> **Note**
>
> EKS Auto Mode verwendet andere Bezeichnungen als Karpenter. Bezeichnungen, die sich auf von EC2 verwaltete Instances beziehen, beginnen mit `eks.amazonaws.com`.

Label (Bezeichnung)	Beispiel	Beschreibung
topology.kubernetes.io/zone	us-east-2a	Region AWS
node.kubernetes.io/instance-type	g4dn.8xgroß	AWS-Instance-Typ
kubernetes.io/arch	amd64	Architekturen werden durch GOARCH-Werte auf der Instance definiert.
karpenter.sh/capacity-type	spot	Zu den Kapazitätstypen gehören `spot`, `on-demand`
eks.amazonaws.com/instance-hypervisor	nitro	Instance-Typen, die einen bestimmten Hypervisor verwenden

Label (Bezeichnung)	Beispiel	Beschreibung
eks.amazonaws.com/compute-type	auto	Identifiziert von EKS Auto Mode verwaltete Knoten
eks.amazonaws.com/instance-encryption-in-transit-supported	true	Instance-Typen, welche die Verschlüsselung während der Übertragung unterstützen (oder nicht)
eks.amazonaws.com/instance-category	g	Instance-Typen derselben Kategorie, in der Regel die Zeichenfolge vor der Generierungsnummer
eks.amazonaws.com/instance-generation	4	Instace-Typ-Generierungsnummer innerhalb einer Instance-Kategorie
eks.amazonaws.com/instance-family	g4dn	Instance-Typen mit ähnlichen Eigenschaften, aber unterschiedlichen Ressourcenmengen
eks.amazonaws.com/instance-size	8xlarge	Instance-Typen mit ähnlichen Ressourcenmengen, jedoch unterschiedlichen Eigenschaften
eks.amazonaws.com/instance-cpu	32	Anzahl der CPUs auf der Instance
eks.amazonaws.com/instance-cpu-manufacturer	`aws`	Name des CPU-Herstellers
eks.amazonaws.com/instance-memory	13 1072	Anzahl der Mebibyte Speicher auf der Instance
eks.amazonaws.com/instance-ebs-bandwidth	9 500	Anzahl der verfügbaren maximalen Megabits auf der Instance
eks.amazonaws.com/instance-network-bandwidth	13 1072	Anzahl der verfügbaren Basis-Megabits auf der Instance
eks.amazonaws.com/instance-gpu-name	t4	Name der GPU auf der Instance, sofern verfügbar

Label (Bezeichnung)	Beispiel	Beschreibung
eks.amazonaws.com/instance-gpu-manufacturer	nvidia	Name des GPU-Herstellers
eks.amazonaws.com/instance-gpu-count	1	Anzahl der GPUs auf der Instance
eks.amazonaws.com/instance-gpu-memory	16 384	Anzahl der Mebibyte Speicher auf der GPU
eks.amazonaws.com/instance-local-nvme	900	Anzahl der Gibibyte lokalen NVMe-Speichers auf der Instance

> **Note**
>
> EKS Auto Mode unterstützt nur bestimmte Instances und hat Mindestgrößenanforderungen. Weitere Informationen finden Sie unter the section called “Unterstützte Instance-Referenz für EKS Auto Mode”.

Von EKS Auto Mode nicht unterstützte Labels

EKS Auto Mode unterstützt die folgenden Labels nicht.

- EKS Auto Mode unterstützt Linux nicht
  - `node.kubernetes.io/windows-build`
  - `kubernetes.io/os`

Integrierte Knoten-Pools deaktivieren

Wenn Sie benutzerdefinierte Knoten-Pools erstellen, können Sie die integrierten Knoten-Pools deaktivieren. Weitere Informationen finden Sie unter the section called “Integrierte Knoten-Pool überprüfen”.

Cluster ohne integrierte Knoten-Pools

Sie können einen Cluster ohne integrierte Knoten-Pools erstellen. Dies ist nützlich, wenn Ihr Unternehmen benutzerdefinierte Knoten-Pools erstellt hat.

> **Note**
>
> Wenn Sie einen Cluster ohne integrierte Knoten-Pools erstellen, wird die `default` NodeClass nicht automatisch bereitgestellt. Sie müssen eine benutzerdefinierte NodeClass erstellen. Weitere Informationen finden Sie unter the section called “Knotenklasse erstellen”.

Übersicht:

1. Erstellen Sie einen EKS-Cluster, bei dem die `nodePools`- und `nodeRoleArn`-Werte leer sind.
   - Beispiel-eksctl `autoModeConfig`:

     ```
     autoModeConfig:
       enabled: true
       nodePools: []
       # Do not set a nodeRoleARN
     ```

     Weitere Informationen finden Sie unter the section called “eksctl CLI”.
2. Benutzerdefinierte Knotenklasse mit einer Knotenrollen-ARN erstellen
   - Weitere Informationen finden Sie unter the section called “Knotenklasse erstellen”.
3. Zugriffseintrag für die benutzerdefinierte Knotenklasse erstellen
   - Weitere Informationen finden Sie unter the section called “Knotenklassen-Zugriffseintrag erstellen”.
4. Erstellen Sie einen benutzerdefinierten Knoten-Pool, wie oben beschrieben.

Unterbrechung

Sie können EKS Auto Mode so konfigurieren, dass Knoten über Ihren NodePool auf verschiedene Arten unterbrochen werden. Sie können `spec.disruption.consolidationPolicy`, `spec.disruption.consolidateAfter` oder `spec.template.spec.expireAfter` verwenden. Sie können die Unterbrechung von EKS Auto Mode auch über den

`spec.disruption.budgets` des NodePools begrenzen. Außerdem können Sie die Zeitfenster und die Anzahl der gleichzeitig unterbrochenen Knoten steuern. Anweisungen zur Konfiguration dieses Verhaltens finden Sie unter Unterbrechung in der Karpenter-Dokumentation.

Sie können Unterbrechungen für Knoten-Pools wie folgt konfigurieren:

- Identifizieren Sie, wann Instances nicht ausreichend ausgelastet sind, und konsolidieren Sie Workloads.
- Erstellen Sie ein Budget für Knoten-Pool-Unterbrechung, um die Beendigung von Knoten aufgrund von Abweichungen, Leerständen und Konsolidierungen zu begrenzen.

Standardmäßig macht EKS Auto Mode Folgendes:

- Konsolidiert nicht ausgelastete Instances.
- Beendet Instances nach 336 Stunden.
- Legt ein einzelnes Budget für Unterbrechung von 10 % der Knoten fest.
- Ermöglicht den Austausch von Knoten aufgrund von Abweichungen, wenn eine neue AMI im Automatikmodus veröffentlicht wird. Das kommt etwa einmal pro Woche vor.

Kündigungsfrist

Wenn ein `terminationGracePeriod` in einem EKS Auto NodePool nicht explizit definiert ist, wendet das System automatisch eine standardmäßige 24-stündige Kündigungsfrist auf den zugehörigen NodeClaim an. Während EKS-Auto-Kunden in ihren benutzerdefinierten NodePool-Konfigurationen keinen Standardwert von `terminationGracePeriod` sehen, beobachten sie diesen Standardwert im NodeClaim. Die Funktionalität bleibt konsistent, unabhängig davon, ob die Karenzzeit explizit im NodePool festgelegt oder im NodeClaim standardmäßig festgelegt ist. Dies gewährleistet ein vorhersehbares Knoten-Beendigungsverhalten im gesamten Cluster.

Knotenpools mit statischer Kapazität im automatischen EKS-Modus

Amazon EKS Auto Mode unterstützt Knotenpools mit statischer Kapazität, die unabhängig vom Pod-Bedarf eine feste Anzahl von Knoten aufrechterhalten. Knotenpools mit statischer Kapazität sind nützlich für Workloads, die vorhersehbare Kapazität, reservierte Instances oder spezifische Compliance-Anforderungen erfordern, bei denen Sie eine konsistente Infrastruktur aufrechterhalten müssen.

Im Gegensatz zu dynamischen Knotenpools, die auf der Grundlage der Pod-Scheduling-Anforderungen skaliert werden, behalten Knotenpools mit statischer Kapazität die Anzahl der von Ihnen konfigurierten Knoten bei.

Einfaches Beispiel

Hier ist ein einfacher Knotenpool mit statischer Kapazität, der 5 Knoten verwaltet:

```
apiVersion: karpenter.sh/v1
kind: NodePool
metadata:
  name: my-static-nodepool
spec:
  replicas: 5  # Maintain exactly 5 nodes

  template:
    spec:
      nodeClassRef:
        group: eks.amazonaws.com
        kind: NodeClass
        name: default

      requirements:
        - key: "eks.amazonaws.com/instance-category"
          operator: In
          values: ["m", "c"]
        - key: "topology.kubernetes.io/zone"
          operator: In
          values: ["us-west-2a", "us-west-2b"]

  limits:
    nodes: 8
```

Konfigurieren Sie einen Knotenpool mit statischer Kapazität

Um einen Knotenpool mit statischer Kapazität zu erstellen, legen Sie das `replicas` Feld in Ihrer Spezifikation fest. NodePool Das `replicas` Feld definiert die genaue Anzahl von Knoten, die der Knotenpool verwalten wird.

Einschränkungen des Knotenpools mit statischer Kapazität

Knotenpools mit statischer Kapazität weisen mehrere wichtige Einschränkungen und Verhaltensweisen auf:

Einschränkungen bei der Konfiguration:

- Der Modus kann nicht gewechselt werden: Sobald Sie einen Knotenpool eingerichtet `replicas` haben, können Sie ihn nicht mehr entfernen. Der Knotenpool kann nicht zwischen statischem und dynamischem Modus wechseln.
- Begrenzte Ressourcenlimits: Nur das `limits.nodes` Feld wird im Abschnitt „Limits“ unterstützt. CPU- und Speichergrenzwerte gelten nicht.
- Kein Gewichtungsfeld: Das `weight` Feld kann nicht für Knotenpools mit statischer Kapazität festgelegt werden, da die Knotenauswahl nicht auf der Priorität basiert.

Betriebsverhalten:

- Keine Konsolidierung: Knoten in Pools mit statischer Kapazität werden für eine Konsolidierung auf der Grundlage der Auslastung nicht berücksichtigt.
- Skalierung von Abläufen: Skalierungsoperationen umgehen Budgets für Knotenunterbrechungen, werden aber trotzdem eingehalten. PodDisruptionBudgets
- Node-Austausch: Nodes werden aufgrund von Drift (z. B. AMI-Updates) und Ablaufzeiten immer noch auf Grundlage Ihrer Konfiguration ersetzt.

Skalieren Sie einen Knotenpool mit statischer Kapazität

Sie können die Anzahl der Replikate in einem Knotenpool mit statischer Kapazität mit dem folgenden Befehl ändern: `kubectl scale`

```
# Scale down to 5 nodes
kubectl scale nodepool static-nodepool --replicas=5
```

Beim Herunterskalieren beendet der automatische Modus von EKS die Knoten ordnungsgemäß. Dabei werden laufende Pods berücksichtigt PodDisruptionBudgets und können auf die verbleibenden Knoten verschoben werden.

Überwachen Sie Knotenpools mit statischer Kapazität

Verwenden Sie die folgenden Befehle, um Ihre Knotenpools mit statischer Kapazität zu überwachen:

```
# View node pool status
kubectl get nodepool static-nodepool

# Get detailed information including current node count
kubectl describe nodepool static-nodepool

# Check the current number of nodes
kubectl get nodepool static-nodepool -o jsonpath='{.status.nodes}'
```

Das `status.nodes` Feld zeigt die aktuelle Anzahl der vom Knotenpool verwalteten Knoten, die unter normalen Bedingungen Ihrer gewünschten `replicas` Anzahl entsprechen sollte.

Beispielkonfigurationen

Grundlegender Knotenpool mit statischer Kapazität

```
apiVersion: karpenter.sh/v1
kind: NodePool
metadata:
  name: basic-static
spec:
  replicas: 5

  template:
    spec:
      nodeClassRef:
        group: eks.amazonaws.com
        kind: NodeClass
        name: default

      requirements:
        - key: "eks.amazonaws.com/instance-category"
          operator: In
          values: ["m"]
        - key: "topology.kubernetes.io/zone"
          operator: In
          values: ["us-west-2a"]
```

```
  limits:
    nodes: 8  # Allow scaling up to 8 during operations
```

Statische Kapazität mit bestimmten Instanztypen

```
apiVersion: karpenter.sh/v1
kind: NodePool
metadata:
  name: reserved-instances
spec:
  replicas: 20

  template:
    metadata:
      labels:
        instance-type: reserved
        cost-center: production
    spec:
      nodeClassRef:
        group: eks.amazonaws.com
        kind: NodeClass
        name: default

      requirements:
        - key: "node.kubernetes.io/instance-type"
          operator: In
          values: ["m5.2xlarge"]  # Specific instance type
        - key: "karpenter.sh/capacity-type"
          operator: In
          values: ["on-demand"]
        - key: "topology.kubernetes.io/zone"
          operator: In
          values: ["us-west-2a", "us-west-2b", "us-west-2c"]

  limits:
    nodes: 25

  disruption:
    # Conservative disruption for production workloads
    budgets:
      - nodes: 10%
```

Knotenpool mit statischer Kapazität in mehreren Zonen

```
apiVersion: karpenter.sh/v1
kind: NodePool
metadata:
  name: multi-zone-static
spec:
  replicas: 12  # Will be distributed across specified zones

  template:
    metadata:
      labels:
        availability: high
    spec:
      nodeClassRef:
        group: eks.amazonaws.com
        kind: NodeClass
        name: default

      requirements:
        - key: "eks.amazonaws.com/instance-category"
          operator: In
          values: ["c", "m"]
        - key: "eks.amazonaws.com/instance-cpu"
          operator: In
          values: ["8", "16"]
        - key: "topology.kubernetes.io/zone"
          operator: In
          values: ["us-west-2a", "us-west-2b", "us-west-2c"]
        - key: "karpenter.sh/capacity-type"
          operator: In
          values: ["on-demand"]

  limits:
    nodes: 15

  disruption:
    budgets:
      - nodes: 25%
```

Best Practices

Kapazitätsplanung:

- Legen Sie einen `limits.nodes` höheren Wert als fest`replicas`, um eine temporäre Skalierung bei Vorgängen zum Austausch von Knoten zu ermöglichen.
- Berücksichtigen Sie bei der Festlegung von Grenzwerten die maximale Kapazität, die bei Node-Drift- oder AMI-Updates benötigt wird.

Auswahl der Instanz:

- Verwenden Sie bestimmte Instance-Typen, wenn Sie Reserved Instances haben oder bestimmte Hardwareanforderungen haben.
- Vermeiden Sie zu restriktive Anforderungen, die die Instance-Verfügbarkeit während der Skalierung einschränken könnten.

Störungsmanagement:

- Richten Sie angemessene Budgets für Unterbrechungen ein, um ein Gleichgewicht zwischen Verfügbarkeit und Wartungsbetrieb herzustellen.
- Berücksichtigen Sie bei der Festlegung von Budgetanteilen die Toleranz Ihrer Anwendung in Bezug auf den Austausch von Knoten.

Überwachung:

- Überwachen Sie das `status.nodes` Feld regelmäßig, um sicherzustellen, dass Ihre gewünschte Kapazität erhalten bleibt.
- Richten Sie Warnmeldungen ein, wenn die tatsächliche Knotenzahl von den gewünschten Replikaten abweicht.

Zonenverteilung:

- Für eine hohe Verfügbarkeit sollten Sie die statische Kapazität auf mehrere Availability Zones verteilen.
- Wenn Sie einen Knotenpool mit statischer Kapazität erstellen, der sich über mehrere Availability Zones erstreckt, verteilt EKS Auto Mode die Knoten auf die angegebenen Zonen, aber es kann nicht garantiert werden, dass die Verteilung gleichmäßig ist.

- Für eine vorhersehbare und gleichmäßige Verteilung über die Availability Zones sollten Sie separate Knotenpools mit statischer Kapazität erstellen, von denen jeder anhand der Anforderung an eine bestimmte Availability Zone gebunden ist. `topology.kubernetes.io/zone`
- Wenn Sie 12 Knoten benötigen, die gleichmäßig auf drei Zonen verteilt sind, erstellen Sie drei Knotenpools mit jeweils 4 Replikaten anstelle eines Knotenpools mit 12 Replikaten in drei Zonen.

Fehlerbehebung

Knoten, die die gewünschten Replikate nicht erreichen:

- Prüfen Sie, ob der `limits.nodes` Wert ausreichend ist
- Stellen Sie sicher, dass Ihre Anforderungen die Instanzauswahl nicht zu stark einschränken
- Prüfen Sie die AWS Servicekontingente für die Instanztypen und Regionen, die Sie verwenden

Der Austausch des Knotens dauert zu lange:

- Passen Sie die Budgets für Unterbrechungen an, um mehr gleichzeitige Ersatzlieferungen zu ermöglichen
- Prüfen Sie, ob PodDisruptionBudgets die Kündigung eines Knotens verhindert wird

Unerwarteter Knotenabbruch:

- Überprüfen Sie `expireAfter` die `terminationGracePeriod` Einstellungen
- Suchen Sie nach manuellen Knotenabbrüchen oder AWS Wartungsereignissen

Erstellen Sie einen IngressClass , um einen Application Load Balancer zu konfigurieren

EKS Auto Mode automatisiert Routineaufgaben für den Lastausgleich, einschließlich der Bereitstellung von Cluster-Anwendungen im Internet

AWS schlägt vor, Application Load Balancers (ALB) für den HTTP- und HTTPS-Verkehr zu verwenden. Application Load Balancers können Anfragen basierend auf dem Inhalt der Anfrage weiterleiten. Weitere Informationen über Application Load Balancer finden Sie unter Was ist elastischer Lastenausgleich?.

Der automatische Modus von EKS erstellt und konfiguriert Application Load Balancers (). ALBs Beispielsweise erstellt EKS Auto Mode einen Load Balancer, wenn Sie ein `Ingress`-Kubernetes-Objekt erstellen, und konfiguriert ihn so, dass der Datenverkehr an Ihre Cluster-Workload weitergeleitet wird.

Übersicht

1. Erstellen Sie eine Workload, die Sie im Internet verfügbar machen möchten.
2. Erstellen Sie eine `IngressClassParams` Ressource und geben Sie AWS bestimmte Konfigurationswerte an, z. B. das zu verwendende Zertifikat SSL/TLS und VPC-Subnetze.
3. Erstellen Sie eine `IngressClass`-Ressource und legen Sie fest, dass EKS Auto Mode als Controller für die Ressource dienen soll.
4. Erstellen Sie eine `Ingress`-Ressource, die einen HTTP-Pfad und einen Port mit einer Cluster-Workload verknüpft.

EKS Auto Mode erstellt einen Application Load Balancer, der auf die in der `Ingress`-Ressource angegebene Workload verweist und die in der `IngressClassParams`-Ressource angegebene Load-Balancer-Konfiguration verwendet.

Voraussetzungen

- EKS Auto Mode in einem Amazon-EKS-Cluster aktiviert
- Kubectl für die Verbindung mit Ihrem Cluster konfiguriert
  - Sie können `kubectl apply -f <filename>` verwenden um die unten aufgeführten YAML-Beispiel-Konfigurationsdateien in Ihren Cluster anwenden.

> **Note**
>
> EKS Auto Mode erfordert Subnetz-Tags zur Identifizierung öffentlicher und privater Subnetze. Wenn Sie Ihren Cluster mit `eksctl` erstellt haben, verfügen Sie bereits über diese Tags. Weitere Informationen erhalten Sie unter the section called “Subnetze kennzeichnen”.

Schritt 1: Workload erstellen

Erstellen Sie zunächst eine Workload, die Sie im Internet verfügbar machen möchten. Dies kann jede Kubernetes-Ressource sein, die HTTP-Datenverkehr bereitstellt, z. B. eine Bereitstellung oder ein Service.

In diesem Beispiel wird ein einfacher HTTP-Service mit dem Namen `service-2048` verwendet, der auf Port `80` empfangsbereit ist. Erstellen Sie diesen Service und dessen Bereitstellung, indem Sie das folgende Manifest `2048-deployment-service.yaml` anwenden:

```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: deployment-2048
spec:
  selector:
    matchLabels:
      app.kubernetes.io/name: app-2048
  replicas: 2
  template:
    metadata:
      labels:
        app.kubernetes.io/name: app-2048
    spec:
      containers:
        - image: public.ecr.aws/l6m2t8p7/docker-2048:latest
          imagePullPolicy: Always
          name: app-2048
          ports:
            - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: service-2048
spec:
  ports:
    - port: 80
      targetPort: 80
      protocol: TCP
  type: NodePort
  selector:
```

```
    app.kubernetes.io/name: app-2048
```

Wenden Sie die Konfiguration auf Ihren Cluster an:

```
kubectl apply -f 2048-deployment-service.yaml
```

Die oben aufgeführten Ressourcen werden im Standard-Namespace erstellt. Sie können dies durch Ausführung des folgenden Befehls überprüfen:

```
kubectl get all -n default
```

Schritt 2: Erstellen IngressClassParams

Erstellen Sie ein `IngressClassParams` Objekt, um AWS spezifische Konfigurationsoptionen für den Application Load Balancer anzugeben. In diesem Beispiel erstellen wir eine `IngressClassParams`-Ressource mit dem Namen `alb` (die Sie im nächsten Schritt verwenden werden), die das Load-Balancer-Schema als `internet-facing` in einer Datei mit dem Namen `alb-ingressclassparams.yaml` festlegt.

```
apiVersion: eks.amazonaws.com/v1
kind: IngressClassParams
metadata:
  name: alb
spec:
  scheme: internet-facing
```

Wenden Sie die Konfiguration auf Ihren Cluster an:

```
kubectl apply -f alb-ingressclassparams.yaml
```

Schritt 3: Erstellen IngressClass

Erstellen Sie eine`IngressClass`, die auf die AWS spezifischen Konfigurationswerte verweist, die in der `IngressClassParams` Ressource in einer Datei mit dem Namen festgelegt sind`alb-ingressclass.yaml`. Notieren Sie den Namen des `IngressClass`. In diesem Beispiel haben sowohl `IngressClass` als auch `IngressClassParams` den Namen `alb`.

Verwenden Sie die `is-default-class`-Annotation, um zu steuern, ob `Ingress`-Ressourcen diese Klasse standardmäßig verwenden sollen.

```
apiVersion: networking.k8s.io/v1
kind: IngressClass
metadata:
  name: alb
  annotations:
    # Use this annotation to set an IngressClass as Default
    # If an Ingress doesn't specify a class, it will use the Default
    ingressclass.kubernetes.io/is-default-class: "true"
spec:
  # Configures the IngressClass to use EKS Auto Mode
  controller: eks.amazonaws.com/alb
  parameters:
    apiGroup: eks.amazonaws.com
    kind: IngressClassParams
    # Use the name of the IngressClassParams set in the previous step
    name: alb
```

Weitere Informationen zu den Konfigurationsoptionen finden Sie unter the section called “IngressClassParams Referenz”.

Wenden Sie die Konfiguration auf Ihren Cluster an:

```
kubectl apply -f alb-ingressclass.yaml
```

Schritt 4: Ingress erstellen

Erstellen Sie eine `Ingress`-Ressource in einer Datei mit dem Namen `alb-ingress.yaml`. Der Zweck dieser Ressource besteht darin, Pfade und Ports dem Application Load Balancer mit Workloads in Ihrem Cluster zuzuordnen. Für dieses Beispiel erstellen wir eine `Ingress`-Ressource mit dem Namen `2048-ingress`, die den Datenverkehr an einen Service mit dem Namen `service-2048` auf Port 80 weiterleitet.

Weitere Informationen zum Konfigurieren dieser Ressource finden Sie unter Ingress in der Kubernetes-Dokumentation.

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: 2048-ingress
spec:
  # this matches the name of IngressClass.
```

```
  # this can be omitted if you have a default ingressClass in cluster: the one with
 ingressclass.kubernetes.io/is-default-class: "true"  annotation
  ingressClassName: alb
  rules:
    - http:
        paths:
          - path: /*
            pathType: ImplementationSpecific
            backend:
              service:
                name: service-2048
                port:
                  number: 80
```

Wenden Sie die Konfiguration auf Ihren Cluster an:

```
kubectl apply -f alb-ingress.yaml
```

Schritt 5: Status überprüfen

Verwenden Sie `kubectl`, um den Status von `Ingress` zu ermitteln. Es kann einige Minuten dauern, bis der Load Balancer verfügbar ist.

Verwenden Sie den Namen der `Ingress`-Ressource, die Sie im vorherigen Schritt festgelegt haben. Beispiel:

```
kubectl get ingress 2048-ingress
```

Sobald die Ressource bereit ist, rufen Sie den Domain-Namen des Load Balancers ab.

```
kubectl get ingress 2048-ingress -o
 jsonpath='{.status.loadBalancer.ingress[0].hostname}'
```

Um den Service in einem Webbrowser anzuzeigen, überprüfen Sie den Port und den Pfad, die in der `Ingress`-Rettungsdatei angegeben sind.

Schritt 6: Bereinigen

Zur Bereinigung des Load Balancer verwenden Sie den folgenden Befehl:

```
kubectl delete ingress 2048-ingress
```

```
kubectl delete ingressclass alb
kubectl delete ingressclassparams alb
```

Der automatische Modus von EKS löscht automatisch den zugehörigen Load Balancer in Ihrem AWS Konto.

IngressClassParams Referenz

Die folgende Tabelle dient als Kurzübersicht über häufig verwendete Konfigurationsoptionen.

Feld	Description	Beispielwert
`scheme`	Legt fest, ob der ALB intern oder mit dem Internet verbunden ist	`internet-facing`
`namespaceSelector`	Schränkt ein, welche Namespaces dies verwenden können IngressClass	`environment: prod`
`group.name`	Gruppiert mehrere Ingresses , um einen einzigen ALB gemeinsam zu nutzen	`retail-apps`
`ipAddressType`	Legt den IP-Adresstyp für den ALB fest	`dualstack`
`subnets.ids`	Liste der Subnetze für die ALB-Bereitstellung IDs	`subnet-xxxx, subnet-yyyy`
`subnets.tags`	Tag-Filter zur Auswahl von Subnetzen für ALB	`Environment: prod`
`certificateARNs`	ARNs der zu verwendenden SSL-Zertifikate	`arn:aws: acm:region:account:certificate/id`
`tags`	Benutzerdefinierte Tags für AWS Ressourcen	`Environment: prod, Team: platform`

Feld	Description	Beispielwert
`loadBalancerAttributes`	Load-Balancer-spezifische Attribute	`idle_timeout.timeout_seconds: 60`

Überlegungen

- Sie können Anmerkungen nicht für eine verwenden, IngressClass um Load Balancer mit dem automatischen EKS-Modus zu konfigurieren.
- Im EKS-Automodus können Sie keine Einstellungen ListenerAttributevornehmen.
- Sie müssen die Cluster-IAM-Rolle aktualisieren, um die Tag-Weitergabe von Kubernetes an AWS Load Balancer-Ressourcen zu ermöglichen. Weitere Informationen finden Sie unter the section called “Benutzerdefinierte AWS Tags für EKS Auto-Ressourcen”.
- Informationen zum Zuordnen von Ressourcen zum automatischen Modus von EKS oder zum selbstverwalteten Load AWS Balancer Controller finden Sie unter. the section called “Migrationsreferenz”
- Informationen zum Beheben von Problemen mit Load Balancern finden Sie unter the section called “Fehlerbehebung”.
- Weitere Überlegungen zur Verwendung der Load-Balancing-Funktion von EKS Auto Mode finden Sie unter the section called “Lastausgleich”.

Die folgenden Tabellen bieten einen detaillierten Vergleich der Änderungen IngressClassParams, der Ingress-Anmerkungen und der TargetGroupBinding Konfigurationen für den EKS-Automatikmodus. Diese Tabellen zeigen die wichtigsten Unterschiede zwischen der Lastenausgleichsfunktion von EKS Auto Mode und dem Open-Source Load Balancer Controller auf, darunter Änderungen der API-Version, veraltete Features und aktualisierte Parameternamen.

IngressClassParams

Vorherige	Neu	Description
`elbv2.k8s.aws/v1beta1`	`eks.amazonaws.com/v1`	Änderung der API-Version

Vorherige	Neu	Description
`spec.certificateArn`	`spec.certificateARNs`	Support für mehrere Zertifikate ARNs
`spec.subnets.tags`	`spec.subnets.matchTags`	Geändertes Schema für die Subnetz-Zuordnung
`spec.listeners.listenerAttributes`	Nicht unterstützt	Wird von EKS Auto Mode noch nicht unterstützt

Ingress-Annotationen

Vorherige	Neu	Description
`kubernetes.io/ingress.class`	Nicht unterstützt	`spec.ingressClassName` in Eingangsobjekten verwenden
`alb.ingress.kubernetes.io/group.name`	Nicht unterstützt	Geben Sie IngressClass nur Gruppen an
`alb.ingress.kubernetes.io/waf-acl-id`	Nicht unterstützt	Stattdessen WAF v2 verwenden
`alb.ingress.kubernetes.io/web-acl-id`	Nicht unterstützt	Stattdessen WAF v2 verwenden
`alb.ingress.kubernetes.io/shield-advanced-protection`	Nicht unterstützt	Shield-Integration deaktiviert
`alb.ingress.kubernetes.io/auth-type: oidc`	Nicht unterstützt	OIDC-Authentifizierungstyp wird derzeit nicht unterstützt

TargetGroupBinding

Vorherige	Neu	Description
`elbv2.k8s.aws/v1beta1`	`eks.amazonaws.com/v1`	Änderung der API-Version
`spec.targetType` optional	`spec.targetType` erforderlich	Explizite Angabe des Zieltyps
`spec.networking.ingress.from`	Nicht unterstützt	NLB ohne Sicherheitsgruppen wird nicht mehr unterstützt

Um die benutzerdefinierte TargetGroupBinding Funktion verwenden zu können, müssen Sie die Zielgruppe mit dem eks: eks-cluster-name -Tag und dem Clusternamen taggen, um dem Controller die erforderlichen IAM-Berechtigungen zu gewähren. Beachten Sie, dass der Controller die Zielgruppe löscht, wenn die TargetGroupBinding Ressource oder der Cluster gelöscht wird.

Verwendung von Service-Annotationen zur Konfiguration von Network Load Balancers

Erfahren Sie, wie Sie Network Load Balancers (NLB) in Amazon EKS mithilfe von Kubernetes-Service-Annotationen konfigurieren. In diesem Thema werden die Anmerkungen erläutert, die vom automatischen Modus von EKS für die Anpassung des NLB-Verhaltens unterstützt werden, einschließlich Internetzugriff, Integritätsprüfungen, SSL/TLS Kündigung und IP-Targeting-Modi.

Wenn Sie einen Kubernetes-Dienst vom Typ `LoadBalancer` im EKS-Automodus erstellen, stellt EKS automatisch einen AWS Network Load Balancer bereit und konfiguriert ihn auf der Grundlage der von Ihnen angegebenen Anmerkungen. Dieser deklarative Ansatz ermöglicht es Ihnen, Load-Balancer-Konfigurationen direkt über Ihre Kubernetes-Manifeste zu verwalten und so die Infrastruktur als Code zu verwalten.

Der automatische Modus von EKS übernimmt standardmäßig die Bereitstellung von Network Load Balancer für alle Dienste dieses Typs LoadBalancer — es ist keine zusätzliche Controller-Installation oder -Konfiguration erforderlich. Die `loadBalancerClass: eks.amazonaws.com/nlb`-Spezifikation wird automatisch als Cluster-Standard festgelegt, wodurch der Bereitstellungsprozess optimiert und gleichzeitig die Kompatibilität mit vorhandenen Kubernetes-Workloads gewährleistet wird.

> **Note**
>
> EKS Auto Mode erfordert Subnetz-Tags zur Identifizierung öffentlicher und privater Subnetze. Wenn Sie Ihren Cluster mit `eksctl` erstellt haben, verfügen Sie bereits über diese Tags. Weitere Informationen erhalten Sie unter the section called “Subnetze kennzeichnen”.

Beispiel-Service

Weitere Informationen zur Kubernetes-`Service`-Ressource finden Sie in der Kubernetes-Dokumentation.

Überprüfen Sie die unten stehende `Service`-Beispiel-Ressource:

```
apiVersion: v1
kind: Service
metadata:
  name: echoserver
  annotations:
    # Specify the load balancer scheme as internet-facing to create a public-facing
 Network Load Balancer (NLB)
    service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing
spec:
  selector:
    app: echoserver
  ports:
    - port: 80
      targetPort: 8080
      protocol: TCP
  type: LoadBalancer
  # Specify the new load balancer class for NLB as part of EKS Auto Mode feature
  # For clusters with Auto Mode enabled, this field can be omitted as it's the default
  loadBalancerClass: eks.amazonaws.com/nlb
```

Häufig verwendete Annotationen

Die folgende Tabelle listet häufig verwendete Annotationen auf, die von EKS Auto Mode unterstützt werden. Beachten Sie, dass EKS Auto Mode möglicherweise nicht alle Annotationen unterstützt.

> **Tip**
>
> Alle folgenden Annotationen müssen mit `service.beta.kubernetes.io/` vorangestellt werden

Feld	Description	Beispiel
`aws-load-balancer-type`	Gibt den Typ des Load Balancers an. Verwenden Sie `external` für neue Bereitstellungen.	`external`
`aws-load-balancer-nlb-target-type`	Gibt an, ob der Datenverkehr an Knoteninstanzen oder direkt an den Pod weitergeleitet werden soll. IPs Verwenden Sie `instance` für Standard-Bereitstellungen oder `ip` für die direkte Pod-Weiterleitung.	`instance`
`aws-load-balancer-scheme`	Steuert, ob der Load Balancer intern oder mit dem Internet verbunden ist.	`internet-facing`
`aws-load-balancer-healthcheck-protocol`	Protokoll für die Zustandsprüfung der Zielgruppe. Gängige Optionen sind `TCP` (Standard) oder `HTTP`.	`HTTP`
`aws-load-balancer-healthcheck-path`	Der HTTP-Pfad für Integritätsprüfungen bei Verwendung HTTP/HTTPS des Protokolls.	`/healthz`
`aws-load-balancer-healthcheck-port`	Der für Zustandsprüfungen verwendete Port. Dies kann	`traffic-port`

Feld	Description	Beispiel
	eine bestimmte Portnummer oder `traffic-port` sein.	
`aws-load-balancer-subnets`	Gibt an, in welchen Subnetzen der Load Balancer erstellt werden soll. Kann Subnetz IDs oder Namen verwenden.	`subnet-xxxx, subnet-yyyy`
`aws-load-balancer-ssl-cert`	ARN des SSL-Zertifikats von AWS Certificate Manager für HTTPS/TLS.	`arn:aws: acm:region:account:certificate/cert-id`
`aws-load-balancer-ssl-ports`	Legt fest, welche Ports SSL/TLS verwenden sollen.	`443, 8443`
`load-balancer-source-ranges`	Für den Zugriff auf den Load Balancer zulässige CIDR-Bereiche.	`10.0.0.0/24, 192.168.1.0/24`
`aws-load-balancer-additional-resource-tags`	Zusätzliche AWS Tags, die auf den Load Balancer und verwandte Ressourcen angewendet werden sollen.	`Environment=prod,Team=platform`
`aws-load-balancer-ip-address-type`	Gibt an, ob der Load Balancer Dual-Stack (IPv4 +) verwendet IPv4 . IPv6	`ipv4` oder `dualstack`

Überlegungen

- Sie müssen die Cluster-IAM-Rolle aktualisieren, um die Tag-Weitergabe von Kubernetes an AWS Load Balancer-Ressourcen zu ermöglichen. Weitere Informationen finden Sie unter the section called “Benutzerdefinierte AWS Tags für EKS Auto-Ressourcen”.
- Informationen zum Zuordnen von Ressourcen zum automatischen Modus von EKS oder zum selbstverwalteten Load AWS Balancer Controller finden Sie unter. the section called “Migrationsreferenz”

- Informationen zum Beheben von Problemen mit Load Balancern finden Sie unter the section called “Fehlerbehebung”.
- Weitere Überlegungen zur Verwendung der Load-Balancing-Funktion von EKS Auto Mode finden Sie unter the section called “Lastausgleich”.

Bei der Migration zu EKS Auto Mode für Load Balancing sind mehrere Änderungen an Service-Annotationen und Ressourcen-Konfigurationen erforderlich. Die folgenden Tabellen zeigen die wichtigsten Unterschiede zwischen der bisherigen und der neuen Implementierung, einschließlich nicht unterstützter Optionen und empfohlener Alternativen.

Service-Annotationen

Vorherige	Neu	Description
`service.beta.kubernetes.io/load-balancer-source-ranges`	Nicht unterstützt	`spec.loadBalancerSourceRanges` für Service verwenden
`service.beta.kubernetes.io/aws-load-balancer-type`	Nicht unterstützt	`spec.loadBalancerClass` für Service verwenden
`service.beta.kubernetes.io/aws-load-balancer-internal`	Nicht unterstützt	Verwenden von `service.beta.kubernetes.io/aws-load-balancer-scheme`
`service.beta.kubernetes.io/aws-load-balancer-proxy-protocol`	Nicht unterstützt	Stattdessen `service.beta.kubernetes.io/aws-load-balancer-target-group-attributes` verwenden
Verschiedene Attribute des Load Balancers	Nicht unterstützt	Verwenden von `service.beta.kubernetes.io/aws-load-balancer-attributes`

Vorherige	Neu	Description
`service.beta.kubernetes.io/aws-load-balancer-access-log-enabled`	Nicht unterstützt	Stattdessen `service.beta.kubernetes.io/aws-load-balancer-attributes` verwenden
`service.beta.kubernetes.io/aws-load-balancer-access-log-s3-bucket-name`	Nicht unterstützt	Stattdessen `service.beta.kubernetes.io/aws-load-balancer-attributes` verwenden
`service.beta.kubernetes.io/aws-load-balancer-access-log-s3-bucket-prefix`	Nicht unterstützt	Stattdessen `service.beta.kubernetes.io/aws-load-balancer-attributes` verwenden
`service.beta.kubernetes.io/aws-load-balancer-cross-zone-load-balancing-enabled`	Nicht unterstützt	Stattdessen `service.beta.kubernetes.io/aws-load-balancer-attributes` verwenden

Um von veralteten Annotation zu Load-Balancer-Attributen zu migrieren, konsolidieren Sie diese Einstellungen in der `service.beta.kubernetes.io/aws-load-balancer-attributes`-Annotation. Diese Annotation akzeptiert eine durch Kommas getrennte Liste von Schlüssel-Wert-Paaren für verschiedene Load-Balancer-Attribute. Verwenden Sie beispielsweise das folgende Format, um die Zugriffsprotokollierung und den zonenübergreifenden Lastenausgleich zu spezifizieren:

```
service.beta.kubernetes.io/aws-load-balancer-attributes:
 access_logs.s3.enabled=true,access_logs.s3.bucket=my-bucket,access_logs.s3.prefix=my-prefix,load_balancing.cross_zone.enabled=true
```

Dieses konsolidierte Format bietet eine konsistentere und flexiblere Möglichkeit zur Konfiguration von Load Balancer-Attributen und reduziert gleichzeitig die Anzahl der erforderlichen einzelnen

Annotationen. Überprüfen Sie Ihre vorhandenen Service-Konfigurationen und aktualisieren Sie sie, um dieses konsolidierte Format zu verwenden.

TargetGroupBinding

Vorherige	Neu	Description
`elbv2.k8s.aws/v1beta1`	`eks.amazonaws.com/v1`	Änderung der API-Version
`spec.targetType` optional	`spec.targetType` erforderlich	Explizite Angabe des Zieltyps
`spec.networking.ingress.from`	Nicht unterstützt	NLB ohne Sicherheitsgruppen wird nicht mehr unterstützt

Hinweis: Um die benutzerdefinierte TargetGroupBinding Funktion verwenden zu können, müssen Sie die Zielgruppe mit dem Tag mit `eks:eks-cluster-name` dem Clusternamen kennzeichnen, um dem Controller die erforderlichen IAM-Berechtigungen zu gewähren. Beachten Sie, dass der Controller die Zielgruppe löscht, wenn die TargetGroupBinding Ressource oder der Cluster gelöscht wird.

Erstellen einer Speicherklasse

Eine `StorageClass` im Amazon EKS Auto Mode definiert, wie Amazon-EBS-Volumes automatisch bereitgestellt werden, wenn Anwendungen persistenten Speicher anfordern. Auf dieser Seite wird erläutert, wie Sie eine `StorageClass` erstellen und konfigurieren, das mit dem Amazon EKS Auto Mode zusammenarbeitet, um EBS-Volumes bereitzustellen.

Durch die Konfiguration einer `StorageClass` können Sie Standardeinstellungen für Ihre EBS-Volumes festlegen, darunter Volume-Typ, Verschlüsselung, IOPS und andere Speicherparameter. Sie können den auch so konfigurieren`StorageClass`, dass AWS KMS-Schlüssel für die Verschlüsselungsverwaltung verwendet werden.

EKS Auto Mode erstellt keine `StorageClass` für Sie. Sie müssen eine `StorageClass`-Referenzierung `ebs.csi.eks.amazonaws.com` erstellen, um die Speicherfunktion von EKS Auto Mode zu nutzen.

Erstellen Sie zunächst eine Datei mit dem Namen `storage-class.yaml`:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: auto-ebs-sc
  annotations:
    storageclass.kubernetes.io/is-default-class: "true"
allowedTopologies:
- matchLabelExpressions:
  - key: eks.amazonaws.com/compute-type
    values:
    - auto
provisioner: ebs.csi.eks.amazonaws.com
volumeBindingMode: WaitForFirstConsumer
parameters:
  type: gp3
  encrypted: "true"
```

Wenden Sie anschließend die Speicherklasse auf Ihren Cluster an.

```
kubectl apply -f storage-class.yaml
```

Zentrale Komponenten:

- `provisioner: ebs.csi.eks.amazonaws.com` – Verwendet EKS Auto Mode
- `allowedTopologies` – Durch die Angabe von `matchLabelExpressions` für die Übereinstimmung mit `eks.amazonaws.com/compute-type:auto` wird sichergestellt, dass Ihre Pods, die ein Volume benötigen, das automatisch im Automatikmodus bereitgestellt wird, nicht auf Nicht-Automatik-Knoten geplant werden.
- `volumeBindingMode: WaitForFirstConsumer` – Verzögert die Volume-Erstellung, bis ein Pod sie benötigt
- `type: gp3` – Gibt den EBS-Volume-Typ an
- `encrypted: "true"` – EBS verschlüsselt alle Volumes, die mit `StorageClass` erstellt wurden. EBS verwendet den `aws/ebs`-Standard-Schlüssel-Alias. Weitere Informationen finden Sie unter So funktioniert die Amazon-EBS-Verschlüsselung im Amazon-EBS-Benutzerhandbuch. Dieser Wert ist optional, wird jedoch empfohlen.
- `storageclass.kubernetes.io/is-default-class: "true"` – Kubernetes verwendet standardmäßig diese Speicherklasse, sofern Sie nicht für einen persistenten Volume-Anspruch

eine andere Volume-Klasse angeben. Dieser Wert ist optional. Seien Sie beim Festlegen dieses Werts vorsichtig, wenn Sie von einem anderen Speicher-Controller migrieren.

Selbstverwalteter KMS-Schlüssel zur Verschlüsselung von EBS-Volumes verwenden

Um einen selbstverwalteten KMS-Schlüssel zur Verschlüsselung von EBS-Volumes zu verwenden, die durch EKS Auto Mode automatisiert werden, müssen Sie folgende Schritte ausführen:

1. Erstellen Sie einen selbstverwalteten KMS-Schlüssel.
   - Weitere Informationen finden Sie unter Erstellen eines symmetrischen KMS-Verschlüsselungsschlüssels oder So verwendet Amazon Elastic Block Store (Amazon EBS) KMS im KMS-Benutzerhandbuch.
2. Erstellen Sie eine neue Richtlinie, die den Zugriff auf den KMS-Schlüssel zulässt.
   - Verwenden Sie die unten aufgeführte Beispiel-IAM-Richtlinie, um die Richtlinie zu erstellen. Geben Sie die ARN des neuen selbstverwalteten KMS-Schlüssels ein. Weitere Informationen finden Sie im AWS IAM-Benutzerhandbuch unter Rollen erstellen und Richtlinien anhängen (Konsole).
3. Fügen Sie die Richtlinie der EKS- Cluster-Rolle an.
   - Verwenden Sie die AWS Konsole, um den ARN der EKS-Clusterrolle zu finden. Die Rollen-Informationen sind im Abschnitt Übersicht einsehbar. Weitere Informationen finden Sie unter the section called “Cluster-IAM-Rolle”.
4. Aktualisieren Sie `StorageClass`, um auf die KMS-Schlüssel-ID im `parameters.kmsKeyId`-Feld zu verweisen.

Beispiel für eine selbstverwaltete KMS-IAM-Richtlinie

Aktualisieren Sie die folgenden Werte in der unten aufgeführten Richtlinie:

- `<account-id>`— Ihre AWS Konto-ID, z. B. `111122223333`
- `<aws-region>`— Die AWS Region Ihres Clusters, wie `us-west-2`

```
{
  "Version":"2012-10-17",
  "Id": "key-auto-policy-3",
  "Statement": [
      {
```

```
          "Sid": "Enable IAM User Permissions",
          "Effect": "Allow",
          "Principal": {
              "AWS": "arn:aws:iam::123456789012:root"
          },
          "Action": "kms:*",
          "Resource": "*"
      },
      {
        "Sid": "Allow access through EBS for all principals in the account that are
 authorized to use EBS",
        "Effect": "Allow",
        "Principal": {
            "AWS": "*"
        },
        "Action": [
            "kms:Encrypt",
            "kms:Decrypt",
            "kms:ReEncrypt*",
            "kms:GenerateDataKey*",
            "kms:CreateGrant",
            "kms:DescribeKey"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "kms:CallerAccount": "123456789012",
                "kms:ViaService": "ec2.us-east-1.amazonaws.com"
            }
        }
    }
  ]
}
```

Beispiel für ein selbstverwaltetes KMS **`StorageClass`**

```
parameters:
  type: gp3
  encrypted: "true"
  kmsKeyId: <custom-key-arn>
```

`StorageClass`-Parameter-Referenz

Allgemeine Informationen zu den `StorageClass`-Kubernetes-Ressourcen finden Sie unter Speicherklassen in der Kubernetes-Dokumentation

THe `parameters`Der Abschnitt der `StorageClass` Ressource ist spezifisch für AWS. Verwenden Sie die folgende Tabelle, um die verfügbaren Optionen zu überprüfen.

Parameters	Werte	Standard	Description
„csi.storage.k8s.io/fstype“	xfs, ext2, ext3, ext4	ext4	Dateisystemtyp, der während der Volume-Erstellung formatiert wird. Bei diesem Parameter wird zwischen Groß- und Kleinschreibung unterschieden!
„type“	io1, io2, gp2, gp3, sc1, st1, standard, sbp1, sbg1	gp3	EBS-Volume-Typ.
„iopsPerGB“			E/A-Vorgänge pro Sekunde pro GiB. Kann für IO1 IO2, und GP3 Volumen angegeben werden.
„Automatisch zulassen IOPSPerGB Increase“	true, false	false	Wenn `"true"` der CSI-Treiber die IOPS für ein Volume erhöht, wenn `iopsPerGB * <volume size>` es zu niedrig ist, um in den von unterstützten IOPS-Bereich zu passen. AWS

Parameters	Werte	Standard	Description
			Dadurch ist die dynamische Bereitstellung immer erfolgreich, selbst wenn der Benutzer eine zu geringe PVC-Kapazität oder einen zu geringen `iopsPerGB` -Wert angibt. Andererseits können zusätzliche Kosten entstehen, da solche Volumes höhere IOPS aufweisen als in `iopsPerGB` angefordert.
„iops“			E/A-Vorgänge pro Sekunde. Kann für IO1, IO2 und GP3 Volumes angegeben werden.
„throughput“		125	Durchsatz in MiB/s. Nur wirksam, wenn der Datenträgertyp gp3 angegeben ist.
„verschlüsselt“	true, false	false	Ob das Volume verschlüsselt werden soll oder nicht. Gültige Werte sind „wahr“ oder „falsch“.

Parameters	Werte	Standard	Description
„blockExpress“	true, false	false	Ermöglicht die Erstellung von io2-Block-Express-Volumes.
"kmsKeyId"			Die vollständige ARN des Schlüssels, der bei der Verschlüsselung des Volumes verwendet werden soll. Wenn nicht angegeben, AWS wird der Standard-KMS-Schlüssel für die Region verwendet, in der sich das Volume befindet. Wenn dieser nicht geändert wird, handelt es sich um einen automatisch generierten Schlüssel mit dem Namen `/aws/ebs`.
„blockSize“			Die Blockgröße, die beim Formatieren des zugrunde liegenden Dateisystems verwendet werden soll. Wird nur auf Linux-Knoten und mit fstype `ext2`, `ext3`, `ext4` oder `xfs` unterstützt.

Parameters	Werte	Standard	Description
„inodeSize“			Die Inode-Größe, die beim Formatier en des zugrunde liegenden Dateisyst ems verwendet werden soll. Wird nur auf Linux-Knoten und mit fstype `ext2`, `ext3`, `ext4` oder `xfs` unterstützt.
"bytesPerInode"			Die `bytes-per-inode`, die beim Formatieren des zugrunde liegenden Dateisystems verwendet werden sollen. Wird nur auf Linux-Knoten und mit fstype `ext2`, `ext3`, `ext4` unterstützt.
"numberOfInodes"			Die `number-of-inodes` , die beim Formatier en des zugrunde liegenden Dateisyst ems verwendet werden sollen. Wird nur auf Linux-Knoten und mit fstype `ext2`, `ext3`, `ext4` unterstüt zt.

Parameters	Werte	Standard	Description
„ext4BigAlloc“	true, false	false	Ändert das `ext4`-Dateisystem, um die gruppierte Block-Zuweisung zu verwenden, indem die `bigalloc`-Formatierungsoption aktiviert wird. Warnung: `bigalloc` wird möglicherweise nicht vollständig vom Linux-Kernel Ihres Knotens unterstützt.
„ClusterSizeext4“			Die Cluster-Größe, die beim Formatieren eines `ext4`-Dateisystems verwendet werden soll, wenn das `bigalloc`-Feature aktiviert ist. Hinweis: Der Parameter `ext4BigAlloc` muss auf „wahr“ gesetzt sein.

Weitere Informationen finden Sie unter dem AWS EBS-CSI-Treiber unter. GitHub

Überlegungen

> **Note**
>
> Sie können Workloads, die vom EKS-Automatikmodus abhängen, nur auf StorageClasses EKS-Automodus-Knoten bereitstellen. Wenn Sie über einen Cluster mit kombinierten Knotentypen verfügen, müssen Sie Ihre Workloads so konfigurieren, dass sie ausschließlich

in EKS-Auto-Mode-Knoten ausgeführt werden. Weitere Informationen finden Sie unter the section called “Kontrollbereitstellung”.

Die Blockspeicherfunktion des EKS Auto Mode unterscheidet sich vom EBS-CSI-Treiber.

- Statische Bereitstellung
  - Wenn Sie extern erstellte EBS-Volumes mit dem automatischen EKS-Modus verwenden möchten, müssen Sie manuell ein AWS Tag mit dem Schlüssel `eks:eks-cluster-name` und dem Wert des Clusternamens hinzufügen.
- Knoten-Startup-Taint
  - Es ist nicht möglich, das Node-Startup-Taint-Feature zu verwenden, um die Pod-Planung zu verhindern, bevor die Speicherkapazität bereit ist.
- Benutzerdefinierte Tags auf dynamisch bereitgestellten Volumes
  - Sie können das CLI-Flag „extra-tag“ nicht verwenden, um benutzerdefinierte Tags für dynamisch bereitgestellte EBS-Volumes zu konfigurieren.
  - Sie können `StorageClass`-Kennzeichnung verwenden, um benutzerdefinierte Tags hinzuzufügen. Der automatische EKS-Modus fügt den zugehörigen Ressourcen Tags hinzu. AWS Sie müssen die Cluster-IAM-Rolle für benutzerdefinierte Tags aktualisieren. Weitere Informationen finden Sie unter the section called “Benutzerdefinierte AWS Tags für EKS Auto-Ressourcen”.
- Detaillierte EBS-Leistungsmetriken
  - Sie können nicht auf Prometheus-Metriken für die detaillierte EBS-Leistung zugreifen

Add-On „CSI-Snapshot Controller“ installieren

EKS Auto Mode ist mit dem Amazon-EKS-Add-On „CSI Snapshot Controller“ kompatibel.

AWS schlägt vor, dass Sie dieses Add-on so konfigurieren, dass es auf dem integrierten `system` Knotenpool ausgeführt wird.

Weitere Informationen finden Sie unter:

- the section called “Kritische Add-Ons ausführen”
- the section called “Integrierte Knoten-Pool überprüfen”
- the section called “CSI-Snapshot-Controller”

So installieren Sie den Snapshot-Controller im System-Knoten-Pool

1. Öffnen Sie Ihren EKS-Cluster in der AWS Konsole
2. Wählen Sie auf der Registerkarte Add-Ons die Option Weitere Add-Ons abrufen
3. Wählen Sie den CSI Snapshot Controller und dann Weiter
4. Wählen Sie auf der Seite Einstellungen für ausgewählte Add-Ons konfigurieren die Option Optionale Konfigurationseinstellungen aus, um das Add-On-Konfigurationsschema anzuzeigen.
   a. Fügen Sie das folgende YAML ein, um den Snapshot-Controller dem `system`-Knoten-Pool zuzuordnen. Der Snapshot-Controller enthält eine Toleranz für den `CriticalAddonsOnly`-Taint.

```
{
        "nodeSelector": {
            "karpenter.sh/nodepool": "system"
        }
}
```

   b. Klicken Sie auf Weiter
5. Überprüfen Sie die Add-On-Konfiguration und wählen Sie dann Erstellen aus

EKS Auto Mode deaktivieren

Sie können EKS Auto Mode in einem vorhandenen EKS-Cluster deaktivieren. Hierbei handelt es sich um einen destruktiven Vorgang.

- EKS beendet alle vom EKS Auto Mode betriebenen EC2 Instanzen.
- EKS wird alle von EKS Auto Mode betriebenen Load Balancer löschen.
- EKS löscht keine EBS-Volumes, die von EKS Auto Mode bereitgestellt werden.

EKS Auto Mode wurde entwickelt, um die von ihm erstellten Ressourcen vollständig zu verwalten. Manuelle Eingriffe können dazu führen, dass EKS Auto Mode diese Ressourcen bei seiner Deaktivierung nicht vollständig bereinigt. Wenn Sie beispielsweise auf eine verwaltete Sicherheitsgruppe aus externen Sicherheitsgruppenregeln verwiesen haben und vergessen, diesen Verweis zu entfernen, bevor Sie EKS Auto Mode für einen Cluster deaktivieren, wird die verwaltete Sicherheitsgruppe verloren gehen (nicht gelöscht). Die folgenden Schritte beschreiben, wie Sie eine verloren gegangene Sicherheitsgruppe entfernen, falls dies passieren sollte.

Deaktivieren Sie den EKS-Automatikmodus (AWS Konsole)

1. Öffnen Sie Ihre Cluster-Übersichtsseite in der AWS-Managementkonsole.
2. Wählen Sie unter EKS Auto Mode die Option Verwalten aus.
3. Schalten Sie EKS Auto Mode auf `off` um.

Wenn eine verwaltete Sicherheitsgruppe am Ende dieses Vorgangs nicht gelöscht wird, können Sie sie mithilfe der Beschreibungen unter Sicherheitsgruppe löschen manuell löschen.

Deaktivieren Sie den EKS-Automatikmodus (AWS CLI)

Verwenden Sie den folgenden Befehl, um EKS Auto Mode in einem vorhandenen Cluster zu deaktivieren.

Sie müssen die `aws`-CLI installiert haben und mit ausreichenden Berechtigungen angemeldet sein, um EKS-Cluster zu verwalten. Weitere Informationen finden Sie unter Einrichten.

> **Note**
>
> Die Funktionen für Rechenleistung, Blockspeicher und Lastenausgleich müssen alle in derselben Anfrage aktiviert oder deaktiviert werden.

```
aws eks update-cluster-config \
 --name $CLUSTER_NAME \
 --compute-config enabled=false \
 --kubernetes-network-config '{"elasticLoadBalancing":{"enabled": false}}' \
 --storage-config '{"blockStorage":{"enabled": false}}'
```

Sie können wie folgt überprüfen, ob eine verloren gegangene Sicherheitsgruppe in EKS Auto Mode nach der Deaktivierung von EKS Auto Mode gelöscht werden konnte:

```
aws ec2 describe-security-groups \
    --filters Name=tag:eks:eks-cluster-name,Values=<cluster-Name> Name=tag-
key,Values=ingress.eks.amazonaws.com/resource,service.eks.amazonaws.com/resource --
query "SecurityGroups[*].[GroupName]"
```

So löschen Sie die Sicherheitsgruppe:

```
aws ec2 delete-security-group --group-name=<sg-name>
```

Aktualisierung der Kubernetes-Version eines Clusters in EKS Auto Mode

In diesem Thema wird erläutert, wie Sie die Kubernetes-Version Ihres Clusters im Automatikmodus aktualisieren können. Der Automatikmodus vereinfacht den Prozess der Versionsaktualisierung, indem er die Koordination der Aktualisierungen der Steuerebene mit dem Austausch von Knoten übernimmt. Gleichzeitig wird die Verfügbarkeit der Workload durch Budgets für Pod-Unterbrechung gewährleistet.

Bei der Aktualisierung eines Clusters im Automatikmodus werden viele Komponenten, die bisher manuell aktualisiert werden mussten, nun als Teil des Services verwaltet. Wenn Sie mit den automatisierten Aspekten des Aktualisierungsprozesses und Ihren Aufgaben vertraut sind, können Sie einen nahtlosen Versionswechsel für Ihren Cluster sicherstellen.

Weitere Informationen zu Aktualisierungen mit EKS Auto Mode

Nachdem Sie ein Upgrade der Steuerebene initiiert haben, aktualisiert EKS Auto Mode die Knoten in Ihrem Cluster. Wenn Knoten ablaufen, ersetzt EKS Auto Mode diese durch neue Knoten. Die neuen Knoten verfügen über die entsprechende neue Kubernetes-Version. EKS Auto Mode berücksichtigt beim Upgrade von Knoten die Budgets für Pod-Unterbrechung.

Darüber hinaus müssen Sie folgende Komponenten nicht mehr aktualisieren:

- Amazon VPC CNI
- AWS-Lastenverteilungs-Controller
- CoreDNS
- `kube-proxy`
- Karpenter
- AWS-EBS-CSI-Treiber

EKS Auto Mode ersetzt diese Komponenten durch Servicefunktionen.

Sie sind weiterhin für die Aktualisierung verantwortlich:

- Apps und Workloads, die in Ihrem Cluster bereitgestellt werden
- Selbstverwaltete Add-Ons und Controller

- Amazon-EKS-Add-Ons
  - Erfahren Sie, wie Sie the section called “Add-On aktualisieren”

Erfahren Sie mehr über Bewährte Methoden für Cluster-Upgrades

Cluster-Aktualisierung starten

Informationen zum Starten einer Cluster-Aktualisierung finden Sie unter the section called “Kubernetes-Version aktualisieren”.

Integriert aktivieren oder deaktivieren NodePools

Der EKS-Automatikmodus verfügt über zwei integrierte Funktionen NodePools. Sie können diese NodePools über die AWS Konsole, CLI oder API aktivieren oder deaktivieren.

Integrierte NodePool Referenz

- `system`
  - Das NodePool hat einen `CriticalAddonsOnly` Makel. Viele EKS-Add-Ons, wie beispielsweise CoreDNS, tolerieren diesen Taint. Verwenden Sie diesen System-KnotenPool, um Cluster-kritische Anwendungen zu trennen.
  - Unterstützt sowohl `amd64`- als auch `arm64`-Architekturen.
- `general-purpose`
  - Dies NodePool bietet Unterstützung für das Starten von Knoten für allgemeine Workloads in Ihrem Cluster.
  - Verwendet nur `amd64`-Architektur.

Beide sind integriert NodePools:

- Verwenden Sie das Standard-EKS NodeClass
- Verwenden Sie nur EC2 On-Demand-Kapazität
- Verwenden Sie die EC2 Instanzfamilien C, M und R
- EC2 Instances der Generation 5 oder neuer erforderlich

> **Note**
>
> Die Aktivierung von mindestens einer integrierten NodePool Komponente ist erforderlich, damit EKS den „Standard“ bereitstellen kann NodeClass. Wenn Sie alle integrierten Funktionen deaktivieren NodePools, müssen Sie eine benutzerdefinierte Version erstellen NodeClass und eine konfigurieren NodePool , um sie zu verwenden. Weitere Informationen zu finden NodeClasses Sie unterthe section called “Knotenklasse erstellen”.

Verfahren

Voraussetzungen

- Die neueste Version der AWS Befehlszeilenschnittstelle (AWS CLI) ist auf Ihrem Gerät installiert und konfiguriert. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version`. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle.
  - Melden Sie sich mit ausreichenden IAM-Berechtigungen bei der CLI an, um AWS Ressourcen wie IAM-Richtlinien, IAM-Rollen und EKS-Cluster zu erstellen.

Mit AWS CLI aktivieren

Verwenden Sie den folgenden Befehl, um beide integrierten Funktionen zu aktivieren NodePools:

```
aws eks update-cluster-config \
  --name <cluster-name> \
  --compute-config '{
    "nodeRoleArn": "<node-role-arn>",
    "nodePools": ["general-purpose", "system"],
    "enabled": true
  }' \
  --kubernetes-network-config '{
  "elasticLoadBalancing":{"enabled": true}
  }' \
  --storage-config '{
  "blockStorage":{"enabled": true}
  }'
```

Sie können den Befehl ändern, um den selektiv zu aktivieren. NodePools

Mit AWS CLI deaktivieren

Verwenden Sie den folgenden Befehl, um beide integrierten Funktionen zu deaktivieren NodePools:

```
aws eks update-cluster-config \
  --name <cluster-name> \
  --compute-config '{
  "enabled": true,
  "nodePools": []
  }' \
  --kubernetes-network-config '{
  "elasticLoadBalancing":{"enabled": true}}' \
  --storage-config '{
  "blockStorage":{"enabled": true}
  }'
```

Überprüfen, ob eine Workload in Knoten von EKS Auto Mode bereitgestellt wird

Bei der Ausführung von Workloads in einem EKS-Cluster mit EKS Auto Mode kann es erforderlich sein, zu steuern, ob bestimmte Workloads in Knoten von EKS Auto Mode oder anderen Rechentypen ausgeführt werden. In diesem Thema wird beschrieben, wie Sie mithilfe von Knoten-Selektoren und Affinitätsregeln sicherstellen können, dass Ihre Workloads in der vorgesehenen Recheninfrastruktur geplant werden.

Die Beispiele in diesem Thema veranschaulichen, wie Sie mithilfe des `eks.amazonaws.com/compute-type`-Labels die Bereitstellung von Workloads auf Knoten in EKS Auto Mode entweder erfordern oder verhindern können. Dies ist besonders nützlich für Cluster im gemischten Modus, in denen sowohl EKS Auto Mode als auch andere Rechentypen ausgeführt werden, beispielsweise selbstverwaltete Karpenter-Bereitsteller oder EKS-verwaltete Knotengruppen.

Knoten von EKS Auto Mode haben den Wert des Labels von `eks.amazonaws.com/compute-type` auf `auto` festgelegt. Mit diesem Label können Sie steuern, ob eine Workload in Knoten bereitgestellt wird, die von EKS Auto Mode verwaltet werden.

Voraussetzung: Eine Workload muss in Knoten von EKS Auto Mode bereitgestellt sein

> **Note**
>
> Dieser `nodeSelector`-Wert ist für EKS Auto Mode nicht erforderlich. Dieser `nodeSelector`-Wert ist nur relevant, wenn Sie einen Cluster im gemischten Modus ausführen, dessen Knotentypen nicht von EKS Auto Mode verwaltet werden. Beispielsweise können Sie in Ihrem Cluster über EKS-verwaltete Knotengruppen statische Rechenkapazität bereitstellen und über dynamische Rechenkapazität verfügen, die von EKS Auto Mode verwaltet wird.

Sie können diesen `nodeSelector` zu Bereitstellungen oder anderen Workloads hinzufügen, um zu verlangen, dass Kubernetes sie in Knoten von EKS Auto Mode plant.

```
apiVersion: apps/v1
kind: Deployment
spec:
  template:
    nodeSelector:
      eks.amazonaws.com/compute-type: auto
```

Voraussetzung: Ein Workload wird nicht in Knoten von EKS Auto Mode bereitgestellt

Sie können diesen `nodeAffinity` zu Bereitstellungen oder anderen Workloads hinzufügen, um zu verlangen, dass Kubernetes sie nicht in Knoten von EKS Auto Mode plant.

```
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: eks.amazonaws.com/compute-type
            operator: NotIn
            values:
            - auto
```

Kritische Add-Ons in dedizierten Instances ausführen

In diesem Thema erfahren Sie, wie Sie eine Workload mit einer `CriticalAddonsOnly`-Toleranz bereitstellen, sodass der EKS Auto Mode sie im `system`-Knoten-Pool plant.

Der integrierte `system`-Knoten-Pool des EKS Auto Mode ist für die Ausführung wichtiger Add-Ons in dedizierten Instances konzipiert. Diese Segmentierung stellt sicher, dass wichtige Komponenten über dedizierte Ressourcen verfügen und von allgemeinen Workloads isoliert sind, was die allgemeine Cluster-Stabilität und -Leistung verbessert.

Dieser Leitfaden zeigt, wie Add-Ons mithilfe von `CriticalAddonsOnly`-Toleranz und geeigneten Knotenselektoren im `system`-Knoten-Pool bereitgestellt werden können. Mit diesen Schritten können Sie sicherstellen, dass Ihre kritischen Anwendungen auf den dedizierten `system`-Knoten geplant werden, wobei Sie die Vorteile der Isolierung und Ressourcenzuweisung nutzen, welche die spezielle Knoten-Poolstruktur des EKS Auto Mode bietet.

EKS Auto Mode verfügt über zwei integrierte Knoten-Pools: `general-purpose` und `system`. Weitere Informationen finden Sie unter the section called “Integrierte Knoten-Pool überprüfen”.

Der Zweck des `system`-Knoten-Pools besteht darin, kritische Add-Ons auf verschiedene Knoten zu verteilen. Von dem `system`-Knoten-Pool bereitgestellte Knoten verfügen über einen `CriticalAddonsOnly`-Kubernetes-Taint. Kubernetes plant Pods nur dann auf diesen Knoten ein, wenn sie über eine entsprechende Toleranz verfügen. Weitere Informationen finden Sie unter Taints und Toleranzen in der Kubernetes-Dokumentation.

Voraussetzungen

- Cluster von EKS Auto Mode mit aktiviertem integrierten `system`-Knoten-Pool. Weitere Informationen finden Sie unter the section called “Integrierte Knoten-Pool überprüfen”.
- `kubectl` installiert und konfiguriert. Weitere Informationen finden Sie unter Einrichten.

Verfahren

Sehen Sie sich das YAML-Beispiel unten an. Beachten Sie die folgenden Konfigurationen:

- `nodeSelector` – Dies ordnet die Workload dem integrierten `system`-Knoten-Pool zu. Dieser Knoten-Pool muss über die AWS-API aktiviert werden. Weitere Informationen finden Sie unter the section called “Integrierte Knoten-Pool überprüfen”.

- `tolerations` – Diese Toleranz überwindet den `CriticalAddonsOnly`-Taint von Knoten im `system`-Knoten-Pool.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: sample-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: sample-app
  template:
    metadata:
      labels:
        app: sample-app
    spec:
      nodeSelector:
        karpenter.sh/nodepool: system
      tolerations:
      - key: "CriticalAddonsOnly"
        operator: "Exists"
      containers:
      - name: app
        image: nginx:latest
        resources:
          requests:
            cpu: "500m"
            memory: "512Mi"
```

Zur Aktualisierung einer Workload für die Ausführung im `system`-Knoten-Pool sind folgende Schritte erforderlich:

1. Aktualisieren Sie die vorhandene Workload, um die folgenden oben beschriebenen Konfigurationen hinzuzufügen:
   - `nodeSelector`
   - `tolerations`
2. Die aktualisierte Workload mit `kubectl apply` in Ihrem Cluster bereitstellen

Nach der Aktualisierung der Workload wird diese in dedizierten Knoten ausgeführt.

Verwendung von Netzwerkrichtlinien mit EKS Auto Mode

-Übersicht

Da Kunden ihre Anwendungsumgebungen mithilfe von EKS skalieren, wird die Isolierung des Netzwerkverkehrs immer wichtiger, um unbefugten Zugriff auf Ressourcen innerhalb und außerhalb des Clusters zu verhindern. Dies ist besonders wichtig in einer Umgebung mit mehreren Mandanten, in der mehrere voneinander unabhängige Workloads nebeneinander im Cluster ausgeführt werden. Mit Kubernetes-Netzwerkrichtlinien können Sie die Netzwerksicherheit für Ihre Kubernetes-Workloads und deren Integrationen mit clusterexternen Endpunkten verbessern. Der automatische Modus von EKS unterstützt verschiedene Arten von Netzwerkrichtlinien.

Isolierung der Schichten 3 und 4

Standard-Kubernetes-Netzwerkrichtlinien arbeiten auf den Ebenen 3 und 4 des OSI-Netzwerkmodells und ermöglichen Ihnen die Steuerung des Datenverkehrs auf IP-Adresse- oder Portebene innerhalb Ihres Amazon EKS-Clusters.

Anwendungsfälle

- Segmentieren Sie den Netzwerkverkehr zwischen Workloads, um sicherzustellen, dass nur verwandte Anwendungen miteinander kommunizieren können.
- Isolieren Sie Mandanten auf Namespace-Ebene mithilfe von Richtlinien, um die Netzwerktrennung durchzusetzen.

DNS-basierte Durchsetzung

Kunden stellen in der Regel Workloads in EKS bereit, die Teil einer breiteren verteilten Umgebung sind, von denen einige mit Systemen und Diensten außerhalb des Clusters kommunizieren müssen (Northbound-Verkehr). Diese Systeme und Dienste können sich in der AWS Cloud oder ganz außerhalb befinden AWS . Mit DNS-basierten Richtlinien (Domain Name System) können Sie Ihre Sicherheitslage verbessern, indem Sie einen stabileren und vorhersehbareren Ansatz verfolgen, um unbefugten Zugriff von Pods auf clusterexterne Ressourcen oder Endpunkte zu verhindern. Durch diesen Mechanismus entfällt die Notwendigkeit, bestimmte IP-Adressen manuell zu verfolgen und aufzulisten. Durch die Sicherung von Ressourcen mit einem DNS-basierten Ansatz haben Sie auch mehr Flexibilität bei der Aktualisierung der externen Infrastruktur, ohne dass Sie Ihre Sicherheitslage lockern oder die Netzwerkrichtlinien ändern müssen, wenn sich Änderungen an vorgelagerten

Servern und Hosts ergeben. Sie können ausgehenden Datenverkehr zu externen Endpunkten entweder mithilfe eines vollqualifizierten Domänennamens (FQDN) oder eines passenden Musters für einen DNS-Domainnamen filtern. Dies bietet Ihnen die zusätzliche Flexibilität, den Zugriff auf mehrere Subdomänen zu erweitern, die einem bestimmten externen Cluster-Endpunkt zugeordnet sind.

Anwendungsfälle

- Verwenden Sie standardmäßig einen DNS-basierten Ansatz zur Filterung des Zugriffs von einer Kubernetes-Umgebung auf clusterexterne Endpunkte.
- Sicherer Zugriff auf Dienste in einer Umgebung mit mehreren Mandanten. AWS
- Verwalten Sie den Netzwerkzugriff von Pods auf lokale Workloads in Ihren Hybrid-Cloud-Umgebungen.

Administratorregeln (oder Clusterregeln)

In einigen Fällen, z. B. in Szenarien mit mehreren Mandanten, müssen Kunden möglicherweise einen Netzwerksicherheitsstandard durchsetzen, der für den gesamten Cluster gilt. Anstatt wiederholt eine eigene Richtlinie für jeden Namespace zu definieren und zu verwalten, können Sie eine einzige Richtlinie verwenden, um die Netzwerkzugriffskontrollen für verschiedene Workloads im Cluster unabhängig von ihrem Namespace zentral zu verwalten. Mit diesen Richtlinientypen können Sie den Umfang der Durchsetzung Ihrer Netzwerkfilterregeln, die auf Ebene 3, Schicht 4 und bei der Verwendung von DNS-Regeln angewendet werden, erweitern.

Anwendungsfälle

- Verwalten Sie die Netzwerkzugriffskontrollen für alle (oder eine Teilmenge von) Workloads in Ihrem EKS-Cluster zentral.
- Definieren Sie einen standardmäßigen Netzwerksicherheitsstatus für den gesamten Cluster.
- Erweitern Sie die organisatorischen Sicherheitsstandards auf betrieblich effizientere Weise auf den gesamten Clusterbereich.

Erste Schritte

Voraussetzungen

- Ein Amazon-EKS-Cluster mit aktiviertem EKS Auto Mode
- Kubectl für die Verbindung mit Ihrem Cluster konfiguriert

Schritt 1: Netzwerkrichtlinien-Controller aktivieren

Um Netzwerkrichtlinien mit dem automatischen EKS-Modus verwenden zu können, müssen Sie zunächst den Network Policy Controller aktivieren, indem Sie a ConfigMap auf Ihren Cluster anwenden.

1. Erstellen Sie eine Datei mit dem Namen `enable-network-policy.yaml` und dem folgenden Inhalt:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: amazon-vpc-cni
  namespace: kube-system
data:
  enable-network-policy-controller: "true"
```

2. Wenden Sie das ConfigMap auf Ihren Cluster an:

```
kubectl apply -f enable-network-policy.yaml
```

Schritt 2: Netzwerkrichtlinien erstellen und testen

Ihr EKS-Auto-Mode-Cluster ist nun für den Support von Kubernetes-Netzwerkrichtlinien konfiguriert. Sie können dies mit the section called “Stars Policy-Demo” testen.

Schritt 3: Passen Sie die Konfiguration des Network Policy Agents in der Knotenklasse an (optional)

Sie können optional eine neue Knotenklasse erstellen, um das Standardverhalten des Network Policy Agent auf den Knoten zu ändern oder die Protokollierung von Netzwerkrichtlinienereignissen zu aktivieren. Führen Sie dazu die folgenden Schritte aus:

1. Erstellen oder bearbeiten Sie eine Knotenklassen-YAML-Datei (z. B. `nodeclass-network-policy.yaml`) mit dem folgenden Inhalt:

```
apiVersion: eks.amazonaws.com/v1
kind: NodeClass
metadata:
  name: network-policy-config
spec:
```

```
  # Optional: Changes default network policy behavior
  networkPolicy: DefaultAllow
  # Optional: Enables logging for network policy events
  networkPolicyEventLogs: Enabled
  # Include other Node Class configurations as needed
```

2. Wenden Sie die Knotenklassen-Konfiguration auf Ihren Cluster an:

```
kubectl apply -f nodeclass-network-policy.yaml
```

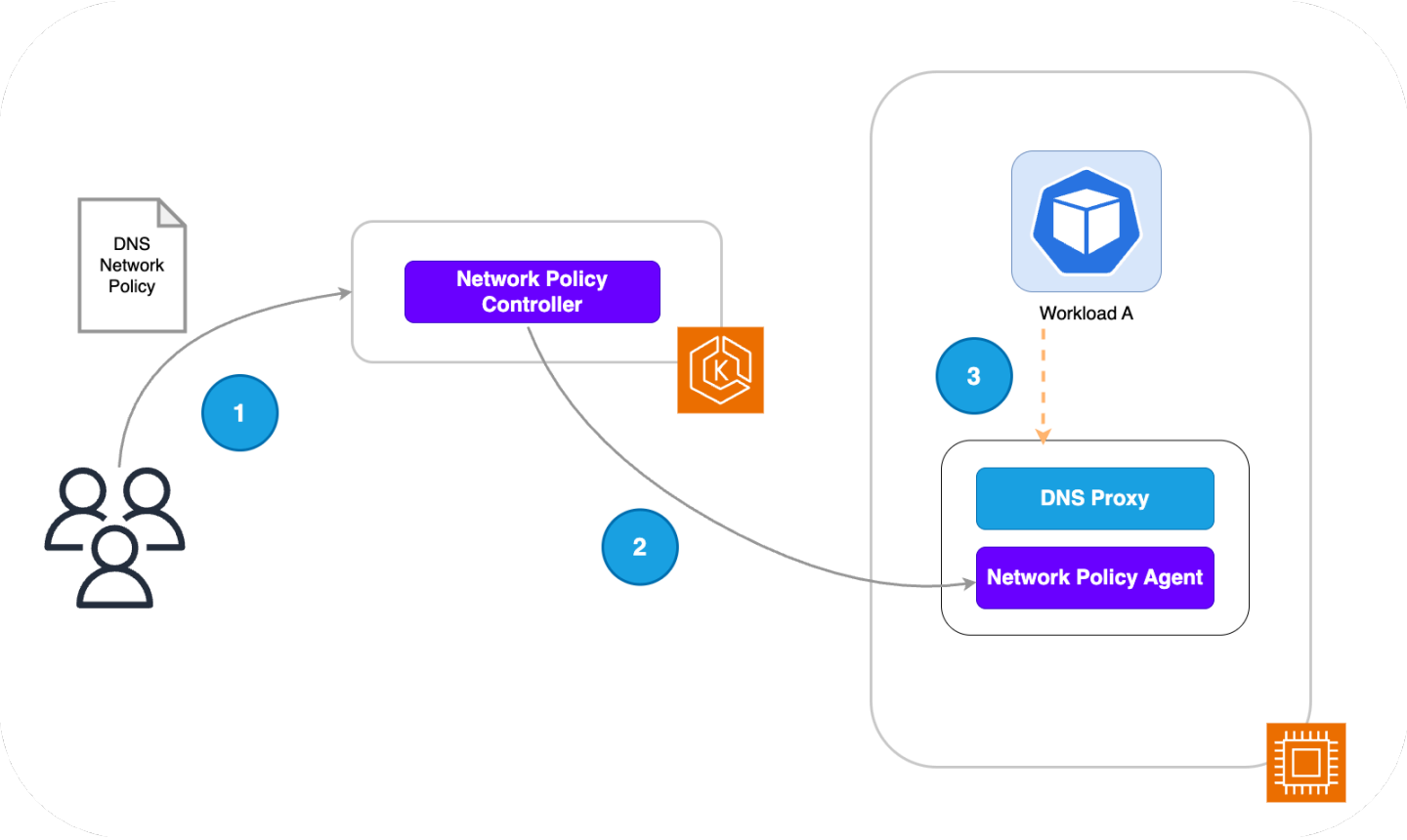
3. Überprüfen Sie, ob die Knotenklasse erstellt wurde:

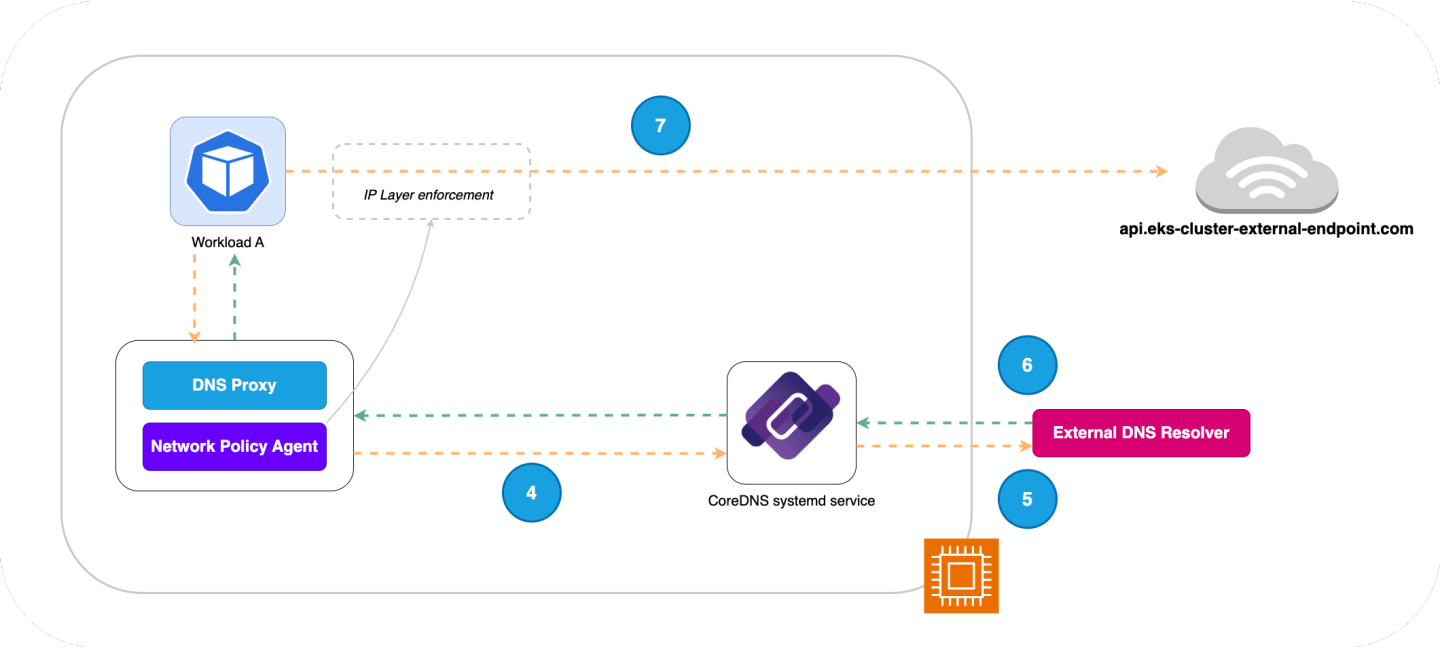
```
kubectl get nodeclass network-policy-config
```

4. Aktualisieren Sie Ihren Knoten-Pool, um diese Knotenklasse zu verwenden. Weitere Informationen finden Sie unter the section called “Knoten-Pool erstellen”.

Funktionsweise

DNS-basierte Netzwerkrichtlinie





1. Das Plattformteam wendet eine DNS-basierte Richtlinie auf den EKS-Cluster an.
2. Der Network Policy Controller ist dafür verantwortlich, die Erstellung von Richtlinien innerhalb des Clusters zu überwachen und anschließend die Richtlinienendpunkte abzustimmen. In diesem Anwendungsfall weist der Netzwerkrichtlinien-Controller den Node-Agent an, DNS-Anfragen auf der Grundlage der Domänen zu filtern, die in der erstellten Richtlinie zugelassen sind. Domänennamen werden mithilfe des FQDN oder eines Domainnamens, der einem in der Kubernetes-Ressourcenkonfiguration definierten Muster entspricht, auf die Zulassungsliste gesetzt.
3. Workload A versucht, die IP für einen clusterexternen Endpunkt aufzulösen. Die DNS-Anfrage durchläuft zunächst einen Proxy, der solche Anfragen anhand der in der Netzwerkrichtlinie festgelegten Zulassungsliste filtert.
4. Sobald die DNS-Anfrage die Zulassungsliste des DNS-Filters durchlaufen hat, wird sie an CoreDNS weitergeleitet.
5. CoreDNS wiederum sendet die Anfrage an den externen DNS-Resolver (Amazon Route 53 Resolver), um die Liste der IP-Adressen hinter dem Domainnamen abzurufen.
6. Die IPs mit TTL aufgelösten Daten werden in der Antwort auf die DNS-Anfrage zurückgegeben. Diese IPs werden dann in eine eBPF-Map geschrieben, die im nächsten Schritt für die Durchsetzung der IP-Schicht verwendet wird.
7. Die an die Pod-Veth-Schnittstelle angeschlossenen eBPF-Sonden filtern dann den ausgehenden Datenverkehr von Workload A zum clusterexternen Endpunkt auf der Grundlage der geltenden Regeln. Dadurch wird sichergestellt, dass Pods nur clusterexternen Datenverkehr an die

aufgelisteten Domänen senden können. IPs Ihre Gültigkeit IPs basiert auf der TTL, die vom externen DNS-Resolver (Amazon Route 53 Resolver) abgerufen wurde.

Verwendung der Anwendungsnetzwerkrichtlinie

Die `ApplicationNetworkPolicy` kombiniert die Funktionen von Standard-Kubernetes-Netzwerkrichtlinien mit DNS-basierter Filterung auf Namespace-Ebene unter Verwendung einer einzigen Custom Resource Definition (CRD). Daher `ApplicationNetworkPolicy` kann der verwendet werden für:

1. Definition von Einschränkungen auf den Ebenen 3 und 4 des Netzwerkstapels mithilfe von IP-Blöcken und Portnummern.
2. Definition von Regeln, die auf Schicht 7 des Netzwerkstapels angewendet werden, und ermöglichen es Ihnen, den Datenverkehr anhand dieser Kriterien zu filtern FQDNs.

Wichtiger Hinweis: DNS-basierte Regeln, die mithilfe von definiert wurden, `ApplicationNetworkPolicy` gelten nur für Workloads, die in im EKS Auto Mode gestarteten Instances ausgeführt werden. EC2

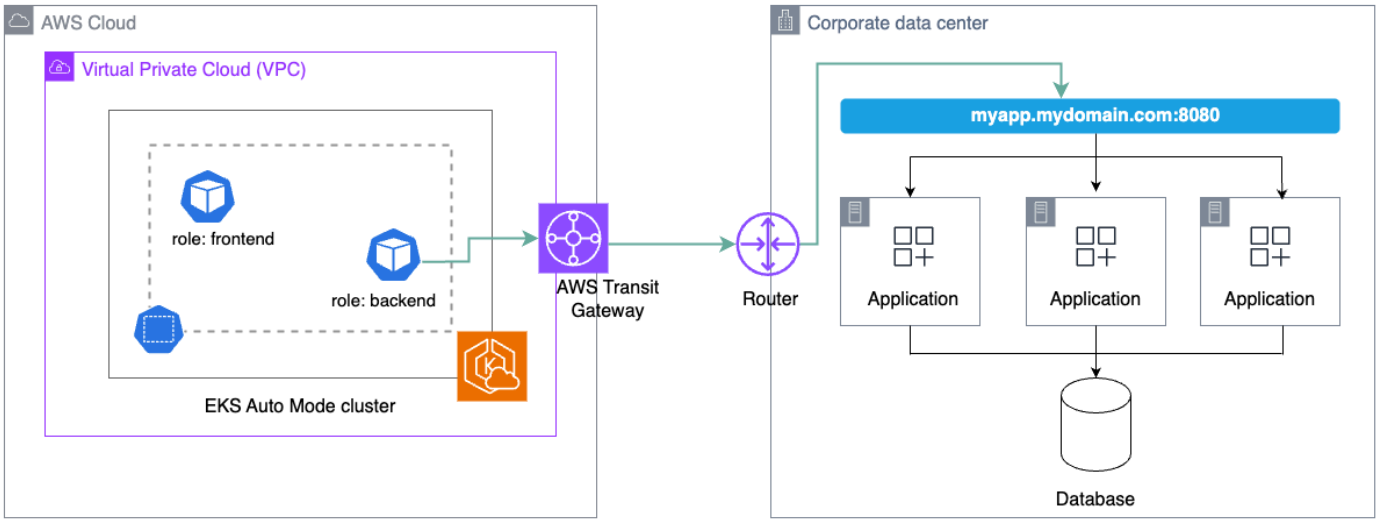
Beispiel

Sie haben einen Workload in Ihrem EKS-Auto-Mode-Cluster, der mit einer lokalen Anwendung kommunizieren muss, die sich hinter einem Load Balancer mit einem DNS-Namen befindet. Sie könnten dies mithilfe der folgenden Netzwerkrichtlinie erreichen:

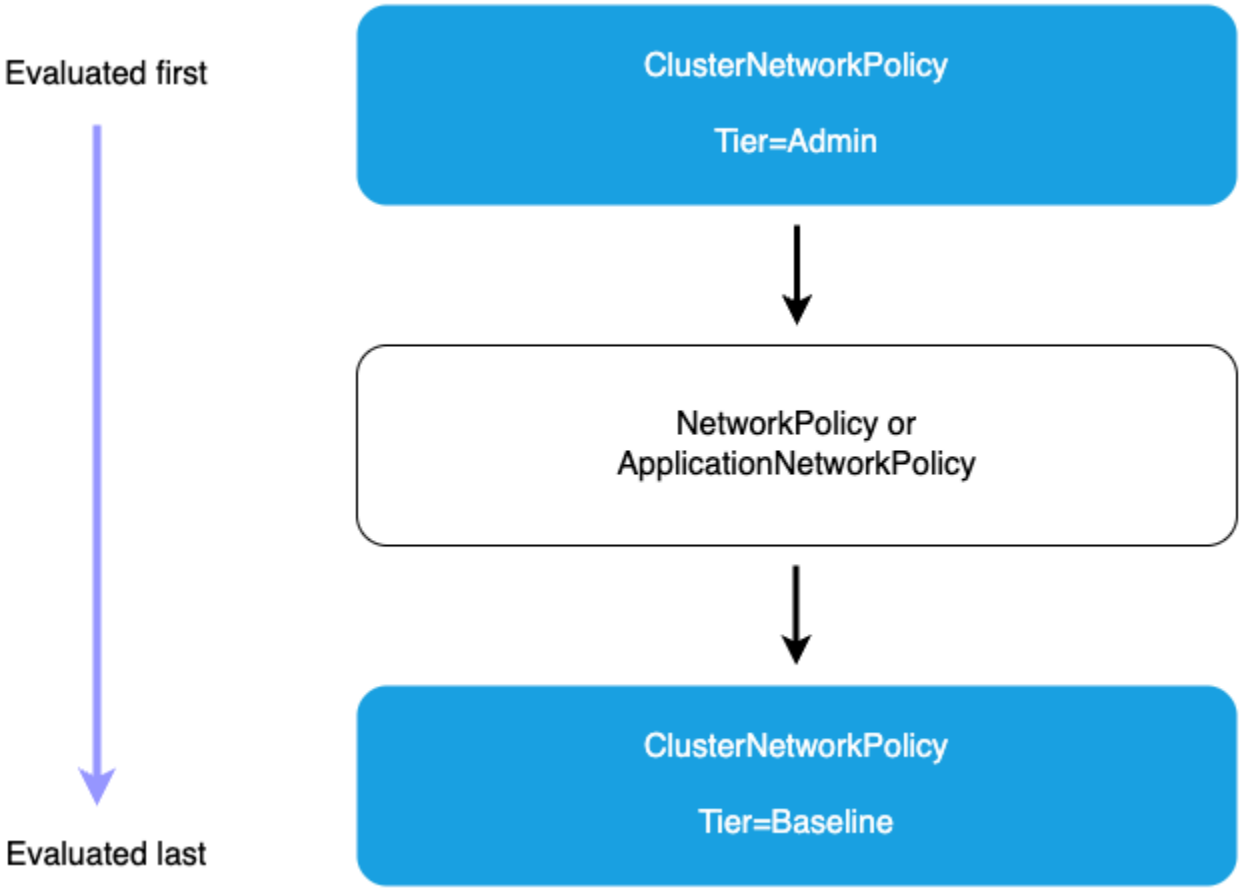
```
apiVersion: networking.k8s.aws/v1alpha1
kind: ApplicationNetworkPolicy
metadata:
  name: my-onprem-app-egress
  namespace: galaxy
spec:
  podSelector:
    matchLabels:
      role: backend
  policyTypes:
  - Egress
  egress:
  - to:
    - domainNames:
      - "myapp.mydomain.com"
```

```
    ports:
    - protocol: TCP
      port: 8080
```

Auf Kubernetes-Netzwerkebene würde dies den Ausgang von allen Pods im „Galaxy“ - Namespace ermöglichen, die mit gekennzeichnet sind, `role: backend` um eine Verbindung zum Domainnamen myapp.mydomain.com am TCP-Port 8080 herzustellen. Darüber hinaus müssten Sie die Netzwerkkonnektivität für den ausgehenden Datenverkehr von Ihrer VPC zu Ihrem Unternehmensrechenzentrum einrichten.



Netzwerkrichtlinie für Administratoren (oder Cluster)



Verwendung der Cluster-Netzwerkrichtlinie

Bei Verwendung von werden die Richtlinien der Admin-Ebene zuerst bewertet und können nicht außer Kraft gesetzt werden. `ClusterNetworkPolicy` Wenn die Richtlinien auf Admin-Ebene evaluiert wurden, werden die angewendeten Netzwerksegmentierungsregeln anhand der standardmäßigen Namespace-Richtlinien ausgeführt. Dies kann entweder mit oder erreicht werden. `ApplicationNetworkPolicy` `NetworkPolicy` Schließlich werden die Baseline-Tier-Regeln, die die Standard-Netzwerkeinschränkungen für Cluster-Workloads definieren, durchgesetzt. Diese Regeln der Baseline-Stufe können bei Bedarf durch die Richtlinien für den Namespace-Bereich außer Kraft gesetzt werden.

Beispiel

Sie haben eine Anwendung in Ihrem Cluster, die Sie von anderen Mandanten-Workloads isolieren möchten. Sie können den Clusterverkehr von anderen Namespaces explizit blockieren, um den Netzwerkzugriff auf den sensiblen Workload-Namespace zu verhindern.

```
apiVersion: networking.k8s.aws/v1alpha1
kind: ClusterNetworkPolicy
metadata:
  name: protect-sensitive-workload
spec:
  tier: Admin
  priority: 10
  subject:
    namespaces:
      matchLabels:
        kubernetes.io/metadata.name: earth
  ingress:
    - action: Deny
      from:
      - namespaces:
          matchLabels: {} # Match all namespaces.
      name: select-all-deny-all
```

Überlegungen

Verstehen Sie die Reihenfolge der Richtlinienbewertung

Die in EKS unterstützten Funktionen für Netzwerkrichtlinien werden in einer bestimmten Reihenfolge bewertet, um ein vorhersehbares und sicheres Datenverkehrsmanagement zu gewährleisten. Daher ist es wichtig, den Bewertungsablauf zu verstehen, um eine effektive Netzwerksicherheit für Ihre Umgebung zu entwickeln.

1. Richtlinien auf Admin-Ebene (zuerst bewertet): ClusterNetworkPolicies Die gesamte Admin-Ebene wird vor allen anderen Richtlinien bewertet. Innerhalb der Admin-Ebene werden Richtlinien in der Reihenfolge ihrer Priorität verarbeitet (Nummer mit der niedrigsten Priorität zuerst). Der Aktionstyp bestimmt, was als Nächstes passiert.
   - Aktion ablehnen (höchste Priorität): Wenn eine Admin-Richtlinie mit der Aktion „Ablehnen“ dem Verkehr entspricht, wird dieser Verkehr sofort blockiert, unabhängig von allen anderen Richtlinien. Es werden keine weiteren ClusterNetworkPolicy NetworkPolicy Regeln verarbeitet. Dadurch wird sichergestellt, dass unternehmensweite Sicherheitskontrollen nicht durch Richtlinien auf Namespace-Ebene außer Kraft gesetzt werden können.
   - Aktion zulassen: Nach der Auswertung der Ablehnungsregeln werden Administratorrichtlinien mit Zulassen-Aktionen in der Reihenfolge ihrer Priorität verarbeitet (die niedrigste Priorität zuerst). Wenn eine Aktion „Zulassen“ zutrifft, wird der Datenverkehr akzeptiert und es findet keine weitere Richtlinienbewertung statt. Diese Richtlinien können auf der Grundlage von

Label-Selektoren Zugriff auf mehrere Namespaces gewähren und ermöglichen so eine zentrale Kontrolle darüber, welche Workloads auf bestimmte Ressourcen zugreifen können.

- Aktion weiterleiten: Bei der Weiterleitung von Aktionen in den Richtlinien der Admin-Ebene wird die Entscheidungsfindung an untergeordnete Ebenen delegiert. Wenn der Traffic mit einer Pass-Regel übereinstimmt, werden bei der Evaluierung alle verbleibenden Regeln auf Admin-Ebene für diesen Traffic übersprungen und direkt zur NetworkPolicy Stufe weitergeleitet. Auf diese Weise können Administratoren die Kontrolle über bestimmte Datenverkehrsmuster explizit an Anwendungsteams delegieren. Sie können beispielsweise Pass-Regeln verwenden, um die Verwaltung des Datenverkehrs innerhalb des Namespaces an Namespace-Administratoren zu delegieren und gleichzeitig strenge Kontrollen des externen Zugriffs aufrechtzuerhalten.

2. Netzwerkrichtlinienebene: Wenn keine Richtlinie auf Admin-Ebene mit „Verweigern“ oder „Zulassen“ übereinstimmt oder wenn eine Pass-Aktion gefunden wurde, werden als Nächstes Ressourcen mit ApplicationNetworkPolicy Namespace-Bereich und herkömmliche Ressourcen bewertet. NetworkPolicy Diese Richtlinien ermöglichen eine detaillierte Steuerung innerhalb der einzelnen Namespaces und werden von Anwendungsteams verwaltet. Namespace-Richtlinien können nur restriktiver sein als Admin-Richtlinien. Sie können die Ablehnungsentscheidung einer Admin-Richtlinie nicht außer Kraft setzen, aber sie können den Datenverkehr weiter einschränken, der durch Admin-Richtlinien zugelassen oder weitergeleitet wurde.
3. Administratorrichtlinien der Baseline-Stufe: Wenn keine Admin- oder Namespace-bezogenen Richtlinien dem Traffic entsprechen, wird die Baseline-Stufe bewertet. ClusterNetworkPolicies Diese bieten standardmäßige Sicherheitsvorkehrungen, die durch Richtlinien im Namespace-Bereich außer Kraft gesetzt werden können. So können Administratoren unternehmensweite Standardeinstellungen festlegen und den Teams gleichzeitig die Flexibilität geben, sie nach Bedarf anzupassen. Basisrichtlinien werden in der Reihenfolge ihrer Priorität bewertet (niedrigste Priorität zuerst).
4. Standardverweigerung (wenn keine Richtlinien zutreffen): Dieses deny-by-default Verhalten stellt sicher, dass nur explizit zugelassene Verbindungen zugelassen werden, wodurch ein hohes Maß an Sicherheit gewährleistet wird.

Anwendung des Prinzips der geringsten Rechte

- Beginnen Sie mit restriktiven Richtlinien und fügen Sie nach Bedarf schrittweise Berechtigungen hinzu. Implementieren Sie zunächst deny-by-default Richtlinien auf Clusterebene und fügen Sie dann schrittweise Zulassungsregeln hinzu, wenn Sie legitime Konnektivitätsanforderungen überprüfen. Dieser Ansatz zwingt die Teams, jede externe Verbindung ausdrücklich zu begründen, wodurch eine sicherere und überprüfbarere Umgebung geschaffen wird.

- Regelmäßige Überprüfung und Entfernung ungenutzter Richtlinienregeln — Netzwerkrichtlinien können sich im Laufe der Zeit ansammeln, wenn sich Anwendungen weiterentwickeln, sodass veraltete Regeln zurückbleiben, die Ihre Angriffsfläche unnötig erweitern. Führen Sie einen regelmäßigen Überprüfungsprozess durch, um nicht mehr benötigte Richtlinienregeln zu identifizieren und zu entfernen. So stellen Sie sicher, dass Ihre Sicherheitsvorkehrungen weiterhin streng und durchführbar sind.
- Verwenden Sie nach Möglichkeit spezifische Domainnamen statt allgemeiner Muster. Platzhaltermuster wie z. B. `*.amazonaws.com` bieten zwar Komfort, gewähren aber auch Zugriff auf eine breite Palette von Diensten. Geben Sie, wann immer möglich, genaue Domainnamen an, z. B. `s3.us-west-2.amazonaws.com` um den Zugriff nur auf die spezifischen Dienste zu beschränken, die Ihre Anwendungen benötigen, wodurch das Risiko einer seitlichen Verlagerung verringert wird, wenn eine Arbeitslast beeinträchtigt wird.

Verwendung von DNS-basierten Richtlinien in EKS

- DNS-basierte Regeln, die mithilfe von definiert wurden, `ApplicationNetworkPolicy` gelten nur für Workloads, die in im automatischen Modus von EKS gestarteten Instanzen ausgeführt werden. EC2 Wenn Sie einen Cluster im gemischten Modus ausführen (der sowohl aus EKS Auto- als auch aus Nicht-EKS Auto-Worker-Knoten besteht), sind Ihre DNS-basierten Regeln nur in den EKS-Workerknoten im automatischen Modus (verwaltete Instanzen) wirksam. EC2

Validierung Ihrer DNS-Richtlinien

- Verwenden Sie zum Testen Staging-Cluster, die die Topologie des Produktionsnetzwerks widerspiegeln. Ihre Staging-Umgebung sollte die Netzwerkarchitektur, die externen Abhängigkeiten und die Konnektivitätsmuster der Produktion replizieren, um genaue Richtlinientests zu gewährleisten. Dazu gehören passende VPC-Konfigurationen, DNS-Auflösungsverhalten und Zugriff auf dieselben externen Dienste, die Ihre Produktions-Workloads benötigen.
- Implementieren Sie automatisierte Tests für kritische Netzwerkpfade — Erstellen Sie automatisierte Tests, die die Konnektivität zu wichtigen externen Diensten als Teil Ihrer CI/CD Pipeline validieren. Mit diesen Tests sollte überprüft werden, ob legitime Datenströme zulässig sind, während nicht autorisierte Verbindungen blockiert werden. Auf diese Weise soll kontinuierlich überprüft werden, ob Ihre Netzwerkrichtlinien bei der Weiterentwicklung Ihrer Infrastruktur den richtigen Sicherheitsstatus beibehalten.
- Überwachen Sie das Anwendungsverhalten nach Richtlinienänderungen — Nach der Implementierung neuer oder geänderter Netzwerkrichtlinien in der Produktion sollten Sie

die Anwendungsprotokolle, Fehlerraten und Leistungskennzahlen genau überwachen, um Verbindungsprobleme schnell zu identifizieren. Richten Sie klare Rollback-Verfahren ein, damit Sie Richtlinienänderungen schnell rückgängig machen können, falls sie zu unerwartetem Anwendungsverhalten oder Betriebsunterbrechungen führen.

Interaktion mit der Amazon Route 53 DNS-Firewall

Die Admin- und Netzwerkrichtlinien von EKS werden zunächst auf Pod-Ebene bewertet, wenn der Datenverkehr initiiert wird. Wenn eine EKS-Netzwerkrichtlinie den Ausgang zu einer bestimmten Domäne zulässt, führt der Pod dann eine DNS-Abfrage durch, die den Route 53 53-Resolver erreicht. Zu diesem Zeitpunkt werden die DNS-Firewallregeln für Route 53 ausgewertet. Wenn die DNS-Firewall die Domänenabfrage blockiert, schlägt die DNS-Auflösung fehl und die Verbindung kann nicht hergestellt werden, obwohl die EKS-Netzwerkrichtlinie dies zulässt. Dadurch entstehen ergänzende Sicherheitsebenen: Die DNS-basierten Netzwerkrichtlinien von EKS bieten Ausgangskontrolle auf Pod-Ebene für anwendungsspezifische Zugriffsanforderungen und mehrinstanzenfähige Sicherheitsgrenzen, während die DNS-Firewall VPC-weiten Schutz vor bekannten bösartigen Domänen bietet und unternehmensweite Blocklisten durchsetzt.

Kennzeichnung von Subnetzen für EKS Auto Mode

Wenn Sie die Lastausgleichsfunktion von EKS Auto Mode verwenden, müssen Sie Ihren VPC-Subnetzen AWS Tags hinzufügen.

Hintergrund

Diese Tags kennzeichnen Subnetze als mit dem Cluster verbunden und, noch wichtiger, sie geben an, ob es sich um ein öffentliches oder privates Subnetz handelt.

Öffentliche Subnetze verfügen über einen direkten Internetzugang über ein Internet-Gateway. Sie werden für Ressourcen verwendet, die öffentlich zugänglich sein müssen, wie beispielsweise Load Balancer.

Private Subnetze verfügen über keinen direkten Internetzugang und verwenden NAT-Gateways für ausgehenden Datenverkehr. Sie werden für interne Ressourcen wie EKS-Knoten verwendet, die keine öffentlichen Ressourcen benötigen. IPs

Weitere Informationen zu NAT-Gateways und Internet-Gateways finden Sie unter Verbinden Ihrer VPC mit anderen Netzwerken im Benutzerhandbuch zu Amazon Virtual Private Cloud (VPC).

Anforderung

Derzeit müssen Subnetze, die für den Lastausgleich im EKS Auto Mode verwendet werden, über eines der folgenden Tags verfügen.

Öffentliche Subnetze

Öffentliche Subnetze werden für Load Balancer mit Internetverbindung verwendet. Diese Subnetze müssen die folgenden Tags aufweisen:

Key (Schlüssel)	Value (Wert)
`kubernetes.io/role/elb`	1 oder ``

Private Subnetze

Private Subnetze werden für interne Load Balancer verwendet. Diese Subnetze müssen die folgenden Tags aufweisen:

Key (Schlüssel)	Value (Wert)
`kubernetes.io/role/internal-elb`	1 oder ``

Verfahren

Ermitteln Sie zunächst, welche Subnetze öffentlich sind (mit Internetzugang) und welche privat sind (mit NAT-Gateway). Sie benötigen Berechtigungen, um VPC-Ressourcen zu ändern.

AWS-Managementkonsole

1. Amazon-VPC-Konsole öffnen und zu „Subnetze“ navigieren
2. Das zu kennzeichnende Subnetz auswählen
3. Registerkarte „Tags“ und anschließend „Tag hinzufügen“ auswählen
4. Das entsprechende Tag hinzufügen:
   - Für öffentliche Subnetze: Schlüssel = `kubernetes.io/role/elb`
   - Für private Subnetze: Schlüssel = `kubernetes.io/role/internal-elb`

5. Wert auf 1 festlegen oder leer lassen
6. Speichern und für verbleibende Subnetze wiederholen

AWS CLI

Für öffentliche Subnetze:

```
aws ec2 create-tags \
    --resources subnet-ID \
    --tags Key=kubernetes.io/role/elb,Value=1
```

Für private Subnetze:

```
aws ec2 create-tags \
    --resources subnet-ID \
    --tags Key=kubernetes.io/role/internal-elb,Value=1
```

Ersetzen Sie `subnet-ID` durch Ihre tatsächliche Subnetz-ID.

CIS-Compliance-Berichte aus Kubernetes-Knoten mithilfe von kubectl debug generieren

In diesem Thema wird beschrieben, wie Sie mithilfe des `kubectl debug`-Befehls CIS-Konformitätsberichte (Center for Internet Security) für Amazon-EKS-Knoten generieren können. Mit diesem Befehl können Sie vorübergehend einen Debugging-Container in einem Kubernetes-Knoten erstellen und CIS-Konformitätsprüfungen mit dem `apiclient`-Tool durchführen. Das `apiclient`-Tool ist Teil von Bottlerocket OS, dem Betriebssystem, das von Knoten in EKS Auto Mode verwendet wird.

Voraussetzungen

Stellen Sie vor Beginn sicher, dass Sie über Folgendes verfügen:

- Zugriff auf einen Amazon-EKS-Cluster mit konfiguriertem `kubectl` (Version muss mindestens v1.32.0 sein. `kubectl version` eingeben, um dies zu überprüfen).
- Die entsprechenden IAM-Berechtigungen zum Debuggen von Knoten.
- Ein gültiges Profil, das Debugging-Vorgänge zulässt (z. B. `sysadmin`).

Weitere Informationen zur Verwendung von Debugging-Profilen mit `kubectl` finden Sie unter Debugging eines Pods oder Knotens bei Anwendung eines Profils in der Kubernetes-Dokumentation.

Verfahren

1. Bestimmen Sie die AWS-Instance-ID des Knotens, in dem Sie den Bericht ausführen möchten. Verwenden Sie den folgenden Befehl, um die Knoten im Cluster aufzulisten. Die Instance-ID befindet sich in der Spalte „Name“ und beginnt mit `i-`:

```
kubectl get nodes
```

```
NAME                  STATUS   ROLES    AGE   VERSION
i-0ea0ba0f8ef9ad609   Ready    <none>   62s   v1.30.10-eks-1a9dacd
```

2. Führen Sie den folgenden Befehl aus und ersetzen Sie `<instance-id>` durch die Instance-ID des Knotens, den Sie abfragen möchten:

```
kubectl debug node/<instance-id> -it --profile=sysadmin --image=public.ecr.aws/
amazonlinux/amazonlinux:2023 -- bash -c "yum install -q -y util-linux-core; nsenter -
t 1 -m apiclient report cis --level 1 --format text"
```

   Zu den Komponenten dieses Befehls gehören:

   - `kubectl debug node/<instance-id>` – Erstellt eine Debugging-Sitzung in der angegebenen EC2-Instance-ID.
   - `-it` – Weist ein TTY (Kommandozeilen-Shell) zu und hält die Standardeingabe für die interaktive Nutzung geöffnet.
   - `--profile=sysadmin` – Verwendet das angegebene `kubectl`-Profil mit den entsprechenden Berechtigungen.
   - `--image=public.ecr.aws/amazonlinux/amazonlinux:2023` – Verwendet `amazonlinux:2023` als Container-Image zum Debuggen.
   - `bash -c "…"` – Führt die folgenden Befehle in einer Bash-Shell aus:
     - `yum install -q -y util-linux-core` – Installiert das erforderliche Service-Programmpaket im Hintergrund.
     - `nsenter -t 1 -m` – Führt `nsenter` aus, um in den Namespace des Host-Prozesses (PID 1) zu gelangen.

- `apiclient report cis --level 1 --format text` – Führt den CIS-Konformitätsbericht auf Ebene 1 mit Textausgabe aus.

3. Überprüfen Sie die Textausgabe des Berichts.

Auswertung der Ausgabe

Der Befehl generiert einen textbasierten Bericht, der den Konformitätsstatus verschiedener CIS-Steuerungen anzeigt. Die Ausgabe enthält:

- Einzelne CIS-Kontroll-IDs
- Beschreibung jeder Steuerung
- Status „Bestanden“, „Nicht bestanden“ oder „Übersprungen“ für jede Prüfung
- Details, die etwaige Compliance-Probleme erklären

Nachfolgend finden Sie ein Beispiel für die Ausgabe des Berichts, der in einer Bottlerocket-Instance ausgeführt wurde:

```
Benchmark name:  CIS Bottlerocket Benchmark
Version:         v1.0.0
Reference:       https://www.cisecurity.org/benchmark/bottlerocket
Benchmark level: 1
Start time:      2025-04-11T01:40:39.055623436Z

[SKIP] 1.2.1     Ensure software update repositories are configured (Manual)
[PASS] 1.3.1     Ensure dm-verity is configured (Automatic)[PASS] 1.4.1     Ensure
 setuid programs do not create core dumps (Automatic)
[PASS] 1.4.2     Ensure address space layout randomization (ASLR) is enabled
 (Automatic)
[PASS] 1.4.3     Ensure unprivileged eBPF is disabled (Automatic)
[PASS] 1.5.1     Ensure SELinux is configured (Automatic)
[SKIP] 1.6       Ensure updates, patches, and additional security software are
 installed (Manual)
[PASS] 2.1.1.1   Ensure chrony is configured (Automatic)
[PASS] 3.2.5     Ensure broadcast ICMP requests are ignored (Automatic)
[PASS] 3.2.6     Ensure bogus ICMP responses are ignored (Automatic)
[PASS] 3.2.7     Ensure TCP SYN Cookies is enabled (Automatic)
[SKIP] 3.4.1.3   Ensure IPv4 outbound and established connections are configured
 (Manual)
```

```
[SKIP] 3.4.2.3   Ensure IPv6 outbound and established connections are configured
 (Manual)
[PASS] 4.1.1.1   Ensure journald is configured to write logs to persistent disk
 (Automatic)
[PASS] 4.1.2     Ensure permissions on journal files are configured (Automatic)

Passed:          11
Failed:          0
Skipped:         4
Total checks:    15
```

Informationen zum Benchmark finden Sie unter Kubernetes Benchmark vom Center for Internet Security (CIS).

Zugehörige Ressourcen

- Bottlerocket CIS Benchmark in der Bottlerocket-OS-Dokumentation.
- Debugging von ausgeführten Pods in der Kubernetes-Dokumentation.
- Kubernetes Benchmark vom Center for Internet Security (CIS)

EBS-Volume-Verschlüsselung mit vom Kunden verwalteten KMS-Schlüsseln für EKS Auto Mode aktivieren

Sie können das flüchtige Root-Volume für Instances in EKS Auto Mode mit einem vom Kunden verwalteten KMS-Schlüssel verschlüsseln.

Amazon EKS Auto Mode verwendet serviceverknüpfte Rollen, um Berechtigungen an andere AWS Services zu delegieren, wenn verschlüsselte EBS-Volumes für Ihre Kubernetes-Cluster verwaltet werden. In diesem Thema wird beschrieben, wie Sie die Schlüsselrichtlinie einrichten, die Sie benötigen, wenn Sie einen vom Kunden verwalteten Schlüssel für die Amazon-EBS-Verschlüsselung mit EKS Auto Mode angeben.

Überlegungen:

- Der automatische Modus von EKS benötigt keine zusätzliche Autorisierung, um den AWS verwalteten Standardschlüssel zum Schutz der verschlüsselten Volumes in Ihrem Konto zu verwenden.

- In diesem Thema wird das Verschlüsseln kurzlebiger Volumes, also der Root-Volumes für Instances, behandelt. EC2 Weitere Informationen zur Verschlüsselung von für Workloads verwendeten Daten-Volumes finden Sie unter the section called “Erstellen StorageClass”.

-Übersicht

Die folgenden AWS KMS-Schlüssel können für die Amazon EBS-Root-Volume-Verschlüsselung verwendet werden, wenn EKS Auto Mode Instances startet:

- AWS -verwalteter Schlüssel – Ein Verschlüsselungsschlüssel in Ihrem Konto, den Amazon EBS erstellt, besitzt und verwaltet. Dies ist der Standardverschlüsselungsschlüssel für ein neues Konto.
- Kundenverwalteter Schlüssel – Ein benutzerdefinierter Verschlüsselungsschlüssel, den Sie erstellen, besitzen und verwalten.

> **Note**
>
> Der Schlüssel muss symmetrisch sein. Amazon EBS unterstützt keine asymmetrischen vom Kunden verwalteten Schlüssel.

Schritt 1: Schlüsselrichtlinie konfigurieren

Ihre KMS-Schlüssel müssen über eine Schlüsselrichtlinie verfügen, die es EKS Auto Mode ermöglicht, Instances mit Amazon-EBS-Volumes zu starten, die mit einem vom Kunden verwalteten Schlüssel verschlüsselt sind.

Konfigurieren Sie Ihre Schlüsselrichtlinie mit der folgenden Struktur:

> **Note**
>
> Diese Richtlinie umfasst nur Berechtigungen für EKS Auto Mode. Die Schlüsselrichtlinie benötigt möglicherweise zusätzliche Berechtigungen, wenn andere Identitäten den Schlüssel verwenden oder Berechtigungen verwalten müssen.

```
{
    "Version":"2012-10-17",
    "Id": "MyKeyPolicy",
```

```
    "Statement": [
        {
            "Sid": "Allow use of the key",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::123456789012:role/ClusterServiceRole"
                ]
            },
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "kms:DescribeKey"
            ],
            "Resource": "*"
        },
        {
            "Sid": "Allow attachment of persistent resources",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::123456789012:role/ClusterServiceRole"
                ]
            },
            "Action": [
                "kms:CreateGrant",
                "kms:ListGrants",
                "kms:RevokeGrant"
            ],
            "Resource": "*",
            "Condition": {
                "Bool": {
                    "kms:GrantIsForAWSResource": "true"
                }
            }
        }
    ]
}
```

Stellen Sie sicher, dass Sie es <account-id> durch Ihre tatsächliche AWS Konto-ID ersetzen.

Beim Konfigurieren der Schlüsselrichtlinie:

- Die `ClusterServiceRole` muss über die erforderlichen IAM-Berechtigungen verfügen, um den KMS-Schlüssel für Verschlüsselungsvorgänge zu verwenden
- Die `kms:GrantIsForAWSResource` Bedingung stellt sicher, dass Zuschüsse nur für AWS Dienstleistungen gewährt werden können

Schritt 2: Konfigurieren Sie NodeClass mit Ihrem vom Kunden verwalteten Schlüssel

Nachdem Sie die Schlüsselrichtlinie konfiguriert haben, verweisen Sie in Ihrer NodeClass EKS-Konfiguration für den automatischen Modus auf den KMS-Schlüssel:

```
apiVersion: eks.amazonaws.com/v1
kind: NodeClass
metadata:
  name: my-node-class
spec:
  # Insert existing configuration

  ephemeralStorage:
    size: "80Gi"  # Range: 1-59000Gi or 1-64000G or 1-58Ti or 1-64T
    iops: 3000    # Range: 3000-16000
    throughput: 125  # Range: 125-1000

    # KMS key for encryption
    kmsKeyID: "arn:aws: kms:<region>:<account-id>:key/<key-id>"
```

Ersetzen Sie die Platzhalterwerte durch Ihre tatsächlichen Werte:

- `<region>`mit Ihrer AWS Region
- `<account-id>`mit deiner AWS Konto-ID
- `<key-id>` mit Ihrer KMS-Schlüssel-ID

Sie können den KMS-Schlüssel in einem der folgenden Formate angeben:

- KMS-Schlüssel-ID: `1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d`
- KMS-Schlüssel-ARN: `arn:aws: kms:us-west-2:111122223333:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d`
- Schlüssel-Alias-Name: `alias/eks-auto-mode-key`

- Schlüssel-Alias-ARN: `arn:aws: kms:us-west-2:111122223333:alias/eks-auto-mode-key`

Wenden Sie die NodeClass Konfiguration mit kubectl an:

```
kubectl apply -f nodeclass.yaml
```

Verwandte Ressourcen

- Knotenklasse für Amazon EKS erstellen
- Weitere Informationen finden Sie im AWS Key Management Service Developer Guide
  - Berechtigungen für AWS Dienste in wichtigen Richtlinien
  - Schlüsselrichtlinie ändern
  - Zuschüsse in AWS KMS

Organisationssteuerung für EKS Auto Mode aktualisieren

Einige Organisationssteuerungen können die korrekte Funktion von EKS Auto Mode beeinträchtigen. In diesem Fall müssen Sie diese Steuerelemente aktualisieren, damit EKS Auto Mode über die erforderlichen Berechtigungen verfügt, um EC2 Instances in Ihrem Namen zu verwalten.

Der automatische Modus von EKS verwendet eine Servicerolle für den Start der EC2 Instances, die die EKS Auto Mode Nodes unterstützen. Eine Servicerolle ist eine IAM-Rolle, die in Ihrem Konto erstellt wird und die ein Service übernimmt, um in Ihrem Namen Aktionen auszuführen. Die Dienststeuerungsrichtlinien (SCPs) gelten immer für Aktionen, die mit Servicerollen ausgeführt werden. Dadurch kann ein SCP die Vorgänge im Automatikmodus unterbinden. Am häufigsten kommt es vor, wenn ein SCP verwendet wird, um die Amazon Machine Images (AMIs) einzuschränken, die gestartet werden können. Damit der EKS-Automodus funktioniert, ändern Sie den SCP so, dass der Start AMIs von EKS-Automodus-Konten aus möglich ist.

Sie können auch die AMIs Funktion „EC2 Zugelassen“ verwenden, um die Sichtbarkeit von AMIs in anderen Konten einzuschränken. Wenn Sie dieses Feature verwenden, müssen Sie die Image-Kriterien erweitern, um auch die AMI-Konten in EKS Auto Mode in den relevanten Regionen einzubeziehen.

Beispiel SCP, um alle AMIs außer dem EKS-Automatikmodus zu blockieren AMIs

Die nachfolgende SCP verhindert den Aufruf von `ec2:RunInstances`, es sei denn, die AMI gehört zum AMI-Konto in EKS Auto Mode für us-west-2 oder us-east-1.

> **Note**
>
> Es ist wichtig, den Kontextschlüssel `ec2:Owner` nicht zu verwenden. Amazon ist Eigentümer der AMI-Konten in EKS Auto Mode, und der Wert für diesen Schlüssel ist stets `amazon`. Die Konstruktion eines SCP, das den Start ermöglicht, AMIs wenn der `ec2:Owner` IS `amazon` das Starten aller Amazon-eigenen AMI ermöglicht, nicht nur der AMIs für den EKS-Automatikmodus. *

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAMI",
      "Effect": "Deny",
      "Action": "ec2:RunInstances",
      "Resource": "arn:*:ec2:*::image/ami-*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "767397842682",
            "992382739861"
          ]
        }
      }
    }
  ]
}
```

AMI-Konten in EKS Auto Mode

AWS Konten, die je nach Region variieren, hosten EKS Auto Mode public AMIs.

AWS Region	Account

af-south-1	471112993317
ap-east-1	590183728416
ap-northeast-1	851725346105
ap-northeast-2	992382805010
ap-northeast-3	891377407544
ap-south-1	975049899075
ap-south-2	590183737426
ap-southeast-1	339712723301
ap-southeast-2	58264376476
ap-southeast-3	471112941769
ap-southeast-4	590183863144
ap-southeast-5	654654202513
ap-southeast-7	533267217478
ca-central-1	992382439851
ca-west-1	767397959864
eu-central-1	891376953411
eu-central-2	381492036002
eu-north-1	339712696471
eu-south-1	975049955519
eu-south-2	471112620929
eu-west-1	381492008532

eu-west-2	590184142468
eu-west-3	891376969258
il-central-1	590183797093
me-central-1	637423494195
me-south-1	905418070398
mx-central-1	211125506622
sa-east-1	339712709251
us-east-1	992382739861
us-east-2	975050179949
us-west-1	975050035094
us-west-2	767397842682

Öffentliche IP-Adresse zuweisen

Wenn `ec2:RunInstances` aufgerufen wird, wird das `AssociatePublicIpAddress`-Feld für einen Instance-Start automatisch durch den Typ des Subnetzes bestimmt, in dem die Instance gestartet wird. Ein SCP kann verwendet werden, um sicherzustellen, dass dieser Wert unabhängig vom Typ des Subnetzes, in dem der Start erfolgt, explizit auf „false“ gesetzt wird. In diesem Fall `spec.advancedNetworking.associatePublicIPAddress` kann das NodeClass Feld auch auf false gesetzt werden, um die Anforderungen des SCP zu erfüllen.

```
{
      "Sid": "DenyPublicEC2IPAddesses",
      "Effect": "Deny",
      "Action": "ec2:RunInstances",
      "Resource": "arn:aws:ec2:*:*:network-interface/*",
      "Condition": {
          "BoolIfExists": {
              "ec2:AssociatePublicIpAddress": "true"
          }
      }
```

```
}
```

Steuern Sie die Bereitstellung von Workloads in Kapazitätsreservierungen mit EKS Auto Mode.

Sie können die Bereitstellung von Workloads in Kapazitätsreservierungen steuern. Der automatische EKS-Modus unterstützt EC2 On-Demand-Kapazitätsreservierungen (ODCRs) und EC2 Kapazitätsblöcke für ML.

> **Tip**
>
> Standardmäßig startet der automatische EKS-Modus automatisch offene Blöcke ODCRs und ML-Kapazitätsblöcke. Bei Verwendung `capacityReservationSelectorTerms` in der NodeClass Definition verwendet der automatische EKS-Modus nicht mehr automatisch offene Kapazitätsreservierungen.

EC2 Kapazitätsreservierungen auf Abruf (ODCRs)

EC2 Kapazitätsreservierungen auf Abruf (ODCRs) ermöglichen es Ihnen, Rechenkapazität für Ihre EC2 Amazon-Instances in einer bestimmten Availability Zone für einen beliebigen Zeitraum zu reservieren. Bei Verwendung von EKS Auto Mode möchten Sie möglicherweise steuern, ob Ihre Kubernetes-Workloads in diesen reservierten Instances bereitgestellt werden, um die Auslastung der im Voraus erworbenen Kapazität zu maximieren oder um sicherzustellen, dass wichtige Workloads Zugriff auf garantierte Ressourcen haben.

Standardmäßig wird der automatische EKS-Modus automatisch geöffnet ODCRs. Durch die Konfiguration `capacityReservationSelectorTerms` auf a können Sie NodeClass jedoch explizit steuern, welche ODCRs Workloads verwendet werden. Knoten, die mithilfe von configured bereitgestellt wurden, haben ODCRs Vorrang vor On-Demand-Nodes `karpenter.sh/capacity-type: reserved` und Spot-Nodes und haben auch weiterhin Priorität. Sobald diese Funktion aktiviert ist, verwendet der automatische EKS-Modus nicht mehr automatisch die Option „Öffnen“ ODCRs. Sie müssen explizit durch ein ausgewählt werden NodeClass, sodass Sie die Kapazitätsreservierungsnutzung in Ihrem gesamten Cluster genau steuern können.

> **Warning**
>
> Wenn Sie NodeClass in `capacityReservationSelectorTerms` einem Cluster eine Konfiguration vornehmen, verwendet EKS Auto Mode nicht mehr automatisch open ODCRs für alle NodeClass im Cluster.

Beispiel NodeClass

```
apiVersion: eks.amazonaws.com/v1
kind: NodeClass
spec:
  # Optional: Selects upon on-demand capacity reservations and capacity blocks
  # for EKS Auto Mode to prioritize.
  capacityReservationSelectorTerms:
    - id: cr-56fac701cc1951b03
    # Alternative Approaches
    - tags:
        app: "my-app"
      # Optional owning account ID filter
      owner: "012345678901"
```

Dieses Beispiel NodeClass zeigt zwei Ansätze für die Auswahl ODCRs. Die erste Methode verweist direkt über die ID (`cr-56fac701cc1951b03`) auf einen bestimmten ODCR. Die zweite Methode verwendet eine Tag-basierte Auswahl, ODCRs wobei das Targeting anhand des Tags `Name: "targeted-odcr"` erfolgt. Sie können optional auch nach dem AWS Konto filtern, dem die Reservierung gehört. Dies ist besonders nützlich in kontoübergreifenden Szenarien oder bei der Arbeit mit Reservierungen für gemeinsame Kapazitäten.

EC2 Kapazitätsblöcke für ML

Kapazitätsblöcke für ML reservieren GPU-basierte beschleunigte Rechen-Instances für einen zukünftigen Zeitpunkt, um Ihre kurzfristigen Machine-Learning-Workloads (ML) zu unterstützen. Instances, die innerhalb eines Kapazitätsblocks ausgeführt werden, werden innerhalb von Amazon automatisch nahe beieinander platziert EC2 UltraClusters, um blockierungsfreie Netzwerke im Petabit-Bereich mit niedriger Latenz zu gewährleisten.

Weitere Informationen zu den unterstützten Plattformen und Instance-Typen finden Sie unter Capacity Blocks for ML im Benutzerhandbuch. EC2

Sie können einen automatischen EKS-Modus erstellen NodeClass , der einen Kapazitätsblock für ML verwendet, ähnlich einem ODCR (weiter oben beschrieben).

Die folgenden Beispieldefinitionen erstellen drei Ressourcen:

1. A NodeClass , das auf Ihre Kapazitätsblock-Reservierung verweist
2. Eine NodePool , die den NodeClass und verwendet, verleiht ihm einen Makel
3. Eine Pod-Spezifikation, die den Taint toleriert und GPU-Ressourcen anfordert

Beispiel NodeClass

Dies NodeClass verweist anhand seiner Reservierungs-ID auf einen bestimmten Kapazitätsblock für ML. Sie können diese ID von der EC2 Konsole abrufen.

```
apiVersion: eks.amazonaws.com/v1
kind: NodeClass
metadata:
  name: gpu
spec:
  # Specify your Capacity Block reservation ID
  capacityReservationSelectorTerms:
    - id: cr-56fac701cc1951b03
```

Weitere Informationen finden Sie unter the section called “Knotenklasse erstellen”.

Beispiel NodePool

Dies NodePool verweist auf die gpu NodeClass und spezifiziert eine wichtige Konfiguration:

- Er nutzt ausschliesslich reservierte Kapazität, indem er `karpenter.sh/capacity-type: reserved` festlegt
- Es werden bestimmte GPU-Instance-Familien angefordert, die für ML-Workloads geeignet sind
- Es wird ein `nvidia.com/gpu` Taint angewendet, um sicherzustellen, dass nur GPU-Workloads auf diesen Knoten geplant werden

```
apiVersion: karpenter.sh/v1
kind: NodePool
metadata:
```

```
  name: gpu
spec:
  template:
    spec:
      nodeClassRef:
        group: eks.amazonaws.com
        kind: NodeClass
        name: gpu
      requirements:
        - key: eks.amazonaws.com/instance-family
          operator: In
          values:
            - g6
            - p4d
            - p4de
            - p5
            - p5e
            - p5en
            - p6
            - p6-b200
        - key: karpenter.sh/capacity-type
          operator: In
          values:
            - reserved
            # Enable other capacity types
            # - on-demand
            # - spot
      taints:
        - effect: NoSchedule
          key: nvidia.com/gpu
```

Weitere Informationen finden Sie unter the section called “Knoten-Pool erstellen”.

Beispiel Pod

Dieses Beispiel-Pod veranschaulicht, wie Sie eine Workload für die Ausführung in Ihren Kapazitätsblock-Knoten konfigurieren:

- Es verwendet einen NodeSelector, um auf bestimmte GPU-Typen abzuzielen (in diesem Fall H200) GPUs
- Es beinhaltet eine Toleranz für den Makel, der durch den `nvidia.com/gpu` NodePool
- Es fordert explizit GPU-Ressourcen unter Verwendung des `nvidia.com/gpu`-Ressourcentyps an

```
apiVersion: v1
kind: Pod
metadata:
  name: nvidia-smi
spec:
  nodeSelector:
    # Select specific GPU type - uncomment as needed
    # eks.amazonaws.com/instance-gpu-name: l4
    # eks.amazonaws.com/instance-gpu-name: a100
    eks.amazonaws.com/instance-gpu-name: h200
    # eks.amazonaws.com/instance-gpu-name: b200
    eks.amazonaws.com/compute-type: auto
  restartPolicy: OnFailure
  containers:
  - name: nvidia-smi
    image: public.ecr.aws/amazonlinux/amazonlinux:2023-minimal
    args:
    - "nvidia-smi"
    resources:
      requests:
        # Uncomment if needed
        # memory: "30Gi"
        # cpu: "3500m"
        nvidia.com/gpu: 1
      limits:
        # Uncomment if needed
        # memory: "30Gi"
        nvidia.com/gpu: 1
  tolerations:
  - key: nvidia.com/gpu
    effect: NoSchedule
    operator: Exists
```

Weitere Informationen finden Sie unter Dry run in der Kubernetes-Dokumentation.

Verwandte Ressourcen

- Kapazitätsblöcke für ML im EC2 Amazon-Benutzerhandbuch
- Capacity Blocks im EC2 Amazon-Benutzerhandbuch finden und kaufen
- Rechenressourcen für AI/ML Workloads auf Amazon EKS verwalten
- GPU-Ressourcenoptimierung und Kostenmanagement im EKS-Leitfaden für bewährte Verfahren

Stellen Sie EKS-Automodus-Knoten in Local Zones bereit

Der automatische Modus von EKS bietet eine vereinfachte Clusterverwaltung mit automatischer Knotenbereitstellung. AWS Local Zones erweitern die AWS Infrastruktur auf geografische Standorte, die näher an Ihren Endbenutzern liegen, und reduzieren so die Latenz für latenzempfindliche Anwendungen. Dieses Handbuch führt Sie durch den Prozess der Bereitstellung von EKS-Automodus-Knoten in AWS Local Zones, sodass Sie containerisierte Anwendungen mit geringerer Latenz für Benutzer in bestimmten geografischen Gebieten ausführen können.

In diesem Leitfaden wird auch gezeigt, wie Sie mithilfe von Kubernetes-Taints and Tolerations sicherstellen können, dass nur bestimmte Workloads auf Ihren Local-Zone-Knoten ausgeführt werden. So können Sie die Kosten kontrollieren und die Ressourcennutzung optimieren.

Voraussetzungen

Bevor Sie mit der Bereitstellung von EKS-Automodus-Knoten in Local Zones beginnen, stellen Sie sicher, dass die folgenden Voraussetzungen erfüllt sind:

- Ein vorhandener EKS-Auto-Mode-Cluster
- Sie haben sich in Ihrem Konto für die lokale Zone angemeldet AWS

Schritt 1: Subnetz für die lokale Zone erstellen

Der erste Schritt bei der Bereitstellung von EKS-Automodus-Knoten in einer lokalen Zone besteht darin, ein Subnetz in dieser lokalen Zone zu erstellen. Dieses Subnetz stellt die Netzwerkinfrastruktur für Ihre Knoten bereit und ermöglicht ihnen, mit dem Rest Ihrer VPC zu kommunizieren. Folgen Sie den Anweisungen zum Erstellen eines Subnetzes für eine lokale Zone (im Benutzerhandbuch für AWS Local Zones), um ein Subnetz in der von Ihnen ausgewählten lokalen Zone zu erstellen.

> **Tip**
>
> Notieren Sie sich den Namen Ihres lokalen Zonen-Subnetzes.

Schritt 2: Erstellen Sie das NodeClass Subnetz für die lokale Zone

Nachdem Sie Ihr Subnetz für die lokale Zone erstellt haben, müssen Sie ein Subnetz definieren, NodeClass das auf dieses Subnetz verweist. Das NodeClass ist eine benutzerdefinierte

Kubernetes-Ressource, die die Infrastrukturattribute für Ihre Knoten spezifiziert, einschließlich der zu verwendenden Subnetze, Sicherheitsgruppen und Speicherkonfigurationen. Im folgenden Beispiel erstellen wir eine NodeClass sogenannte „lokale Zone“, die anhand ihres Namens auf ein Subnetz der lokalen Zone abzielt. Sie können auch die Subnetz-ID verwenden. Sie müssen diese Konfiguration an Ihr lokales Zonen-Subnetz anpassen.

Weitere Informationen finden Sie unter the section called “Knotenklasse erstellen”.

```
apiVersion: eks.amazonaws.com/v1
kind: NodeClass
metadata:
  name: local-zone
spec:
  subnetSelectorTerms:
    - id: <local-subnet-id>
```

Schritt 3: Create NodePool with NodeClass und Taint

Nachdem Sie es NodeClass konfiguriert haben, müssen Sie jetzt eine erstellen NodePool , die dies NodeClass verwendet. A NodePool definiert die Recheneigenschaften Ihrer Knoten, einschließlich der Instanztypen. Der NodePool verwendet das NodeClass als Referenz, um zu bestimmen, wo Instances gestartet werden sollen.

Im folgenden Beispiel erstellen wir eine, NodePool die auf unsere „lokale Zone“ verweist. NodeClass Außerdem fügen wir den Knoten einen Tain hinzu, um sicherzustellen, dass nur Pods mit einer entsprechenden Toleranz auf diesen Local-Zone-Knoten geplant werden können. Dies ist besonders wichtig für Knoten in der lokalen Zone, die in der Regel höhere Kosten verursachen und nur von Workloads verwendet werden sollten, die speziell von der reduzierten Latenz profitieren.

Weitere Informationen finden Sie unter the section called “Knoten-Pool erstellen”.

```
apiVersion: karpenter.sh/v1
kind: NodePool
metadata:
  name: my-node-pool
spec:
  template:
    metadata:
      labels:
        node-type: local-zone
```

```
    spec:
      nodeClassRef:
        group: eks.amazonaws.com
        kind: NodeClass
        name: local-zone
      taints:
        - key: "aws.amazon.com/local-zone"
          value: "true"
          effect: NoSchedule

      requirements:
        - key: "eks.amazonaws.com/instance-category"
          operator: In
          values: ["c", "m", "r"]
        - key: "eks.amazonaws.com/instance-cpu"
          operator: In
          values: ["4", "8", "16", "32"]
```

Der Nachteil von Schlüssel `aws.amazon.com/local-zone` und Effekt `NoSchedule` stellt sicher, dass Pods ohne entsprechende Toleranzen nicht auf diesen Knoten eingeplant werden. Dadurch wird verhindert, dass reguläre Workloads versehentlich in der lokalen Zone ausgeführt werden, was zu unerwarteten Kosten führen könnte.

Schritt 4: Stellen Sie Workloads mit Toleranz und Knotenaffinität bereit

Für eine optimale Kontrolle der Workload-Platzierung auf Knoten in der lokalen Zone sollten Sie beide taints/tolerations und die Knotenaffinität zusammen verwenden. Dieser kombinierte Ansatz bietet die folgenden Vorteile:

1. Kostenkontrolle: Dieser Fehler stellt sicher, dass nur Pods mit ausdrücklichen Toleranzen potenziell teure Local-Zone-Ressourcen nutzen können.
2. Garantierte Platzierung: Durch die Knotenaffinität wird sichergestellt, dass Ihre latenzempfindlichen Anwendungen ausschließlich in der lokalen Zone und nicht auf regulären Clusterknoten ausgeführt werden.

Hier ist ein Beispiel für eine Bereitstellung, die speziell für die Ausführung auf Knoten der lokalen Zone konfiguriert wurde:

```
apiVersion: apps/v1
kind: Deployment
```

```
metadata:
  name: low-latency-app
  namespace: default
spec:
  replicas: 2
  selector:
    matchLabels:
      app: low-latency-app
  template:
    metadata:
      labels:
        app: low-latency-app
    spec:
      tolerations:
      - key: "aws.amazon.com/local-zone"
        operator: "Equal"
        value: "true"
        effect: "NoSchedule"
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
              - key: "node-type"
                operator: "In"
                values: ["local-zone"]
      containers:
      - name: application
        image: my-low-latency-app:latest
        resources:
          limits:
            cpu: "1"
            memory: "1Gi"
          requests:
            cpu: "500m"
            memory: "512Mi"
```

Diese Bereitstellung hat zwei wichtige Planungskonfigurationen:

1. Diese Toleranz ermöglicht es, die Pods auf Knoten mit dem `aws.amazon.com/local-zone` Makel zu planen.
2. Die Anforderung an die Knotenaffinität stellt sicher, dass diese Pods nur auf Knoten mit dem Label ausgeführt werden. `node-type: local-zone`

Zusammen stellen sie sicher, dass Ihre latenzempfindliche Anwendung nur auf Knoten der lokalen Zone ausgeführt wird und reguläre Anwendungen die Ressourcen der lokalen Zone nicht verbrauchen, sofern sie nicht ausdrücklich dafür konfiguriert wurden.

Schritt 5: Überprüfen Sie mit der Konsole AWS

Nachdem Sie Ihre NodeClass NodePool, und Bereitstellungen eingerichtet haben, sollten Sie überprüfen, ob die Knoten wie erwartet in Ihrer lokalen Zone bereitgestellt werden und ob Ihre Workloads auf ihnen ausgeführt werden. Sie können die AWS Managementkonsole verwenden, um zu überprüfen, ob EC2 Instances im richtigen Subnetz der lokalen Zone gestartet werden.

Darüber hinaus können Sie anhand `kubectl get nodes -o wide` der Kubernetes-Knotenliste überprüfen, ob die Knoten Ihrem Cluster mit den richtigen Labels und Taints beitreten:

```
kubectl get nodes -o wide
kubectl describe node <node-name> | grep -A 5 Taints
```

Sie können auch überprüfen, ob Ihre Workload-Pods auf den Knoten der lokalen Zone geplant sind:

```
kubectl get pods -o wide
```

Durch diesen Ansatz wird sichergestellt, dass auf diesen Knoten nur Workloads geplant werden, die den Taint Local Zone speziell tolerieren. Auf diese Weise können Sie die Kosten kontrollieren und die Ressourcen Ihrer lokalen Zone so effizient wie möglich nutzen.

Erweiterte Sicherheitseinstellungen für Knoten konfigurieren

In diesem Thema wird beschrieben, wie Sie mithilfe der `advancedSecurity` Spezifikation in Ihrer Node Class erweiterte Sicherheitseinstellungen für Amazon EKS Auto Mode-Knoten konfigurieren.

Voraussetzungen

Stellen Sie vor Beginn sicher, dass Sie über Folgendes verfügen:

- Ein Cluster in Amazon EKS Auto Mode. Weitere Informationen finden Sie unter the section called “Cluster erstellen”.
- `kubectl` installiert und konfiguriert. Weitere Informationen finden Sie unter Einrichten.
- Verständnis der Node Class-Konfiguration. Weitere Informationen finden Sie unter the section called “Knotenklasse erstellen”.

Konfigurieren Sie erweiterte Sicherheitseinstellungen

Um erweiterte Sicherheitseinstellungen für Ihre Knoten zu konfigurieren, legen Sie die `advancedSecurity` Felder in Ihrer Node Class-Spezifikation fest:

```
apiVersion: eks.amazonaws.com/v1
kind: NodeClass
metadata:
  name: security-hardened
spec:
  role: MyNodeRole

  subnetSelectorTerms:
    - tags:
        Name: "private-subnet"

  securityGroupSelectorTerms:
    - tags:
        Name: "eks-cluster-sg"

  advancedSecurity:
    # Enable FIPS-compliant AMIs (US regions only)
    fips: true

    # Configure kernel lockdown mode
    kernelLockdown: "integrity"
```

Wenden Sie diese Konfiguration an:

```
kubectl apply -f nodeclass.yaml
```

Verweisen Sie in Ihrer Node-Pool-Konfiguration auf diese Node-Klasse. Weitere Informationen finden Sie unter the section called “Knoten-Pool erstellen”.

Beschreibungen der Felder

- `fips`(boolean, optional): Wenn auf gesetzt`true`, werden Knoten mithilfe von FIPS AMIs 140-2-validierten kryptografischen Modulen bereitgestellt. Mit dieser Einstellung wird FIPS-konform ausgewählt. Kunden sind für die Verwaltung ihrer Compliance-Anforderungen AMIs verantwortlich. Weitere Informationen finden Sie unter FIPS-Konformität.AWS Standard: `false`.

- `kernelLockdown`(Zeichenfolge, optional): Steuert den Modus des Kernel-Lockdown-Sicherheitsmoduls. Zulässige Werte:
  - `integrity`: Blockiert Methoden zum Überschreiben des Kernelspeichers oder zum Ändern des Kernelcodes. Verhindert das Laden von Kernelmodulen ohne Vorzeichen.
  - `none`: Deaktiviert den Kernel-Lockdown-Schutz.

    Weitere Informationen finden Sie in der Dokumentation zum Lockdown des Linux-Kernels.

Überlegungen

- FIPS-konform AMIs sind in den Regionen AWS USA Ost/West, AWS GovCloud (USA) und AWS Kanada (Zentral/West) verfügbar. Weitere Informationen finden Sie unter FIPS-Konformität.AWS
- Stellen Sie bei der Verwendung sicher`kernelLockdown: "integrity"`, dass Ihre Workloads nicht das Laden von unsignierten Kernelmodulen oder das Ändern des Kernelspeichers erfordern.

Zugehörige Ressourcen

- the section called “Knotenklasse erstellen”- Vollständiger Leitfaden zur Konfiguration von Node Class
- the section called “Knoten-Pool erstellen”- Konfiguration des Knotenpools

Weitere Informationen zur Funktionsweise von EKS Auto Mode

In diesem Kapitel erfahren Sie, wie die Komponenten von Amazon-EKS-Auto-Mode-Clustern funktionieren.

Themen

- Weitere Informationen zu von Amazon EKS Auto Mode verwalteten Instances
- Weitere Informationen zu Identität und Zugriff in EKS Auto Mode
- Weitere Informationen zu VPC-Netzwerke und Load Balancing in EKS Auto Mode

Weitere Informationen zu von Amazon EKS Auto Mode verwalteten Instances

In diesem Thema wird erklärt, wie Amazon EKS Auto Mode EC2 Amazon-Instances in Ihrem EKS-Cluster verwaltet. Wenn Sie den automatischen EKS-Modus aktivieren, werden die Rechenressourcen Ihres Clusters automatisch von EKS bereitgestellt und verwaltet, wodurch sich die Art und Weise ändert, wie Sie mit den EC2 Instances interagieren, die als Knoten in Ihrem Cluster dienen.

Das Verständnis der Funktionsweise von Amazon EKS Auto Mode bei der Verwaltung von Instances ist für die Planung Ihrer Workload-Bereitstellungsstrategie und Ihrer Betriebsabläufe von entscheidender Bedeutung. Im Gegensatz zu herkömmlichen EC2 Instanzen oder verwalteten Knotengruppen folgen diese Instanzen einem anderen Lebenszyklusmodell, bei dem EKS die Verantwortung für viele betriebliche Aspekte übernimmt und gleichzeitig bestimmte Zugriffs- und Anpassungsarten einschränkt.

Amazon EKS Auto Mode automatisiert Routineaufgaben zur Erstellung neuer EC2 Instances und fügt sie als Knoten an Ihren EKS-Cluster an. Der automatische Modus von EKS erkennt, wenn ein Workload nicht auf bestehende Knoten passt, und erstellt eine neue EC2 Instance.

Amazon EKS Auto Mode ist für das Erstellen, Löschen und Patchen von EC2 Instances verantwortlich. Sie sind für die auf der Instance bereitgestellten Container und Pods verantwortlich.

EC2 Mit EKS Auto Mode erstellte Instances unterscheiden sich von anderen EC2 Instances, es handelt sich um verwaltete Instances. Diese verwalteten Instances sind Eigentum von EKS und unterliegen strengeren Beschränkungen. Sie können nicht direkt auf von EKS Auto Mode verwaltete Instances zugreifen oder Software darauf installieren.

AWS schlägt vor, entweder den EKS Auto Mode oder den selbstverwalteten Karpenter auszuführen. Sie können beide während einer Migration oder in einer erweiterten Konfiguration installieren. Wenn Sie beide installiert haben, konfigurieren Sie Ihre Knoten-Pools so, dass Workloads entweder Karpenter oder EKS Auto Mode zugeordnet sind.

Weitere Informationen finden Sie unter Von Amazon EC2 verwaltete Instances im EC2 Amazon-Benutzerhandbuch.

Vergleichstabelle

EC2 Standardinstanz	Von EKS Auto Mode verwaltete Instance
Sie sind für das Patchen und Aktualisieren der Instance verantwortlich.	AWS patcht und aktualisiert die Instanz automatisch.
EKS übernimmt keine Verantwortung für die Software auf der Instance.	EKS ist für bestimmte Software auf der Instance verantwortlich, wie `kubelet`, die Container-Laufzeitumgebung und das Betriebssystem.
Sie können die EC2 Instanz mithilfe der EC2 API löschen.	EKS bestimmt die Anzahl der in Ihrem Konto bereitgestellten Instances. Wenn Sie eine Workload löschen, reduziert EKS die Anzahl der Instances in Ihrem Konto.
Sie können SSH verwenden, um auf die EC2 Instance zuzugreifen.	Sie können Pods und Container auf der verwalteten Instance bereitstellen.
Sie bestimmen das Betriebssystem und das Image (AMI).	AWS bestimmt das Betriebssystem und das Image.
Sie können Workloads bereitstellen, die auf Windows- oder Ubuntu-Funktionalität basieren.	Sie können Container auf Linux-Basis bereitstellen, jedoch ohne spezifische Betriebssystemabhängigkeiten.
Sie legen fest, welcher Instance-Typ und welche Instance-Familie gestartet werden soll.	AWS bestimmt, welcher Instance-Typ und welche Familie gestartet werden sollen. Sie können einen Knoten-Pool verwenden, um die Instance-Typen einzuschränken, aus denen EKS Auto Mode auswählt.

Die folgende Funktionalität funktioniert sowohl für verwaltete Instances als auch für EC2 Standard-Instances:

- Sie können die Instanz in der AWS Konsole anzeigen.
- Sie können den Instance-Speicher als flüchtigen Speicher für Workloads verwenden.

AMI-Support

AWS Bestimmt im EKS-Automatikmodus das Image (AMI), das für Ihre Rechenknoten verwendet wird. AWS überwacht die Einführung neuer EKS Auto Mode AMI-Versionen. Wenn Sie Probleme mit der Workload im Zusammenhang mit einer AMI-Version haben, erstellen Sie einen Support-Fall. Weitere Informationen finden Sie unter Erstellen von Supportfällen und Fallmanagement im AWS Support-Benutzerhandbuch.

Im Allgemeinen veröffentlicht EKS jede Woche eine neue AMI, die CVE- und Sicherheitskorrekturen enthält.

Unterstützte Instance-Referenz für EKS Auto Mode

EKS Auto Mode erstellt ausschließlich Instances von unterstützten Typen, die eine Mindestgröße aufweisen.

EKS Auto Mode unterstützt die folgenden Instance-Typen:

Familie	Instance-Typen
Für Datenverarbeitung optimiert (C)	c8i, c8i-flex, c8gd, c8gn, c8g, c7a, c7g, c7gn, c7gn, c7gd, c7i, c7i-flex, c6a, c6g, c6i, c6gn, c6id, c6in, c6gd, c5, c5a, c5d, c5ad, c5n, c4
Allzweck (M)	m8i, m8i-flex, m8a, m8gn, m8gb, m8gd, m8g, m7i, m7a, m7g, m7g, m7i-flex, m6a, m6i, m6in, m6in, m6in, m6idn, m6gd, m5, m5a, m5ad, m5n, m5dn, m5d, m5zn, m4
Speicheroptimiert (R)	r8i, r8i-flex, r8gn, r8gb, r8gd, r8g, r7a, r7iz, r7iz, r7gd, r7i, r7g, r6a, r6i, r6id, r6in, r6idn, r6g, r6gd, r5, r5n, r5a, r5dn, r5b, r5ad, r5ad, r5n d, r4
Mit Spitzenlast (T)	t4g, t3, t3a, t2
Hoher Arbeitsspeicher (Z/X)	z1d, x8g, x2gd
Speicheroptimiert (I/D)	i8ge, i7i, i8g, i7ie, i4g, i4i, i3, i3en, is4gen, d3, d3en, im4gn

Familie	Instance-Typen
P/G/Inf/T rnBeschleunigtes Rechnen ()	p5, p4d, p4de, p3, p3dn, gr6, g6, g6e, g5g, g5, g4dn, inf2, inf1, trn1, trn1n
Hochleistungs-Datenverarbeitung (X2)	x2iezn, x2iedn, x2idn

Darüber hinaus erstellt der automatische Modus von EKS nur EC2 Instanzen, die die folgenden Anforderungen erfüllen:

- Mehr als 1 CPU
- Die Instance-Größe ist nicht Nano, Mikro oder klein

Weitere Informationen finden Sie unter Benennungskonventionen für EC2 Amazon-Instance-Typen.

Instance-Metadatenservice

- Im automatischen Modus von EKS gilt standardmäßig ein IMDSv2 Hop-Limit von 1, was den bewährten AWS Sicherheitsmethoden entspricht.
- Diese Standardkonfiguration kann im Automatikmodus nicht geändert werden.
- Geben Sie bei Add-Ons, für die normalerweise IMDS-Zugriff erforderlich ist, während der Installation Parameter (z. B. AWS Region) an, um IMDS-Abfragen zu vermeiden. Weitere Informationen finden Sie unter the section called “Felder, die angepasst werden können”.
- Wenn ein Pod im Automatikmodus unbedingt IMDS-Zugriff benötigt, muss der Pod für die Ausführung mit `hostNetwork: true` konfiguriert werden. Dadurch kann der Pod direkt auf den Instance-Metadaten-Service zugreifen.
- Berücksichtigen Sie die Sicherheitsauswirkungen, wenn Sie Pods Zugriff auf Instance-Metadaten gewähren.

Weitere Informationen zum Amazon EC2 Instance Metadata Service (IMDS) finden Sie unter Konfiguration der Optionen für den Instance-Metadaten-Service im EC2 Amazon-Benutzerhandbuch.

Überlegungen

- Wenn der konfigurierte flüchtige Speicher in der kleiner NodeClass ist als der NVMe lokale Speicher für die Instance, macht der automatische EKS-Modus eine manuelle Konfiguration überflüssig, indem automatisch die folgenden Aktionen ausgeführt werden:
  - Verwendet ein kleineres (20 GiB) Amazon-EBS-Daten-Volume, um die Kosten zu senken.
  - Formatiert und konfiguriert den NVMe lokalen Speicher für die Verwendung kurzlebiger Daten. Dies beinhaltet die Einrichtung eines RAID 0-Arrays, falls mehrere Laufwerke vorhanden sind. NVMe
- Wenn `ephemeralStorage.size` die lokale NVMe Kapazität erreicht oder diese überschreitet, werden die folgenden Aktionen ausgeführt:
  - Automatikmodus überspringt das kleine EBS-Volume.
  - Die NVMe Laufwerke sind direkt für Ihre Arbeitslast verfügbar.
- Der automatische Modus von Amazon EKS unterstützt die folgenden Aktionen des AWS Fault Injection Service nicht:
  - `ec2:RebootInstances`
  - `ec2:SendSpotInstanceInterruptions`
  - `ec2:StartInstances`
  - `ec2:StopInstances`
  - `ec2:TerminateInstances`
  - `ec2:PauseVolumeIO`
- Amazon EKS Auto Mode unterstützt EKS-Pod-Aktionen des AWS Fault Injection Service. Weitere Informationen finden Sie unter Managing Fault Injection Service Experiments und Use the AWS FIS aws:eks:pod actions im Resilience Hub-Benutzerhandbuch. AWS
- Sie brauchen das `Neuron Device Plugin` nicht in EKS-Auto-Mode-Knoten zu installieren.

  Wenn Sie andere Knotentypen in Ihrem Cluster haben, müssen Sie das Neuron-Geräte-Plugin so konfigurieren, dass es nicht in Knoten im Automatikmodus ausgeführt wird. Weitere Informationen finden Sie unter the section called “Kontrollbereitstellung”.

Weitere Informationen zu Identität und Zugriff in EKS Auto Mode

In diesem Thema werden die Rollen und Berechtigungen des Identity and Access Management (IAM) beschrieben, die für die Verwendung von EKS Auto Mode erforderlich sind. EKS Auto Mode

verwendet zwei primäre IAM-Rollen: eine Cluster-IAM-Rolle und eine Knoten-IAM-Rolle. Diese Rollen arbeiten in Verbindung mit EKS Pod Identity und EKS-Zugriffseinträgen, um eine umfassende Zugriffsverwaltung für Ihre EKS-Cluster zu gewährleisten.

Wenn Sie den automatischen EKS-Modus konfigurieren, müssen Sie diese IAM-Rollen mit bestimmten Berechtigungen einrichten, die es AWS Diensten ermöglichen, mit Ihren Clusterressourcen zu interagieren. Dazu gehören Berechtigungen für die Verwaltung von Rechenressourcen, Speicher-Volumes, Load Balancern und Netzwerkkomponenten. Das Verständnis dieser Rollenkonfigurationen ist für den ordnungsgemäßen Betrieb und die Sicherheit des Clusters von entscheidender Bedeutung.

Im automatischen EKS-Modus werden AWS IAM-Rollen über EKS-Zugriffseinträge automatisch den Kubernetes-Berechtigungen zugeordnet, sodass keine manuelle Konfiguration oder benutzerdefinierte Bindungen erforderlich sind. `aws-auth` ConfigMaps Wenn Sie einen neuen Auto-Mode-Cluster erstellen, erstellt EKS automatisch die entsprechenden Kubernetes-Berechtigungen mithilfe von Access-Einträgen und stellt so sicher, dass AWS Dienste und Clusterkomponenten sowohl innerhalb des Kubernetes-Autorisierungssystems als auch innerhalb des Kubernetes-Autorisierungssystems über die AWS entsprechenden Zugriffsebenen verfügen. Diese automatisierte Integration reduziert die Komplexität der Konfiguration und trägt dazu bei, Probleme im Zusammenhang mit Berechtigungen zu vermeiden, die bei der Verwaltung von EKS-Clustern häufig auftreten.

Cluster-IAM-Rolle

Die Cluster-IAM-Rolle ist eine AWS Identity and Access Management (IAM) -Rolle, die von Amazon EKS zur Verwaltung von Berechtigungen für Kubernetes-Cluster verwendet wird. Diese Rolle gewährt Amazon EKS die erforderlichen Berechtigungen für die Interaktion mit anderen AWS Services im Namen Ihres Clusters und wird mithilfe von EKS-Zugriffseinträgen automatisch mit Kubernetes-Berechtigungen konfiguriert.

- Sie müssen dieser Rolle AWS IAM-Richtlinien zuordnen.
- EKS Auto Mode weist dieser Rolle automatisch Kubernetes-Berechtigungen mithilfe von EKS-Zugriffseinträgen zu.
- Schlägt AWS vor, im automatischen Modus von EKS eine einzelne Cluster-IAM-Rolle pro AWS Konto zu erstellen.
- AWS schlägt vor, diese Rolle `AmazonEKSAutoClusterRole` zu benennen.
- Diese Rolle erfordert Berechtigungen für mehrere AWS Dienste zur Verwaltung von Ressourcen, einschließlich EBS-Volumes, Elastic Load Balancers und EC2 Instances.

- Die vorgeschlagene Konfiguration für diese Rolle umfasst mehrere AWS verwaltete IAM-Richtlinien, die sich auf die verschiedenen Funktionen von EKS Auto Mode beziehen.
  - `AmazonEKSComputePolicy`
  - `AmazonEKSBlockStoragePolicy`
  - `AmazonEKSLoadBalancingPolicy`
  - `AmazonEKSNetworkingPolicy`
  - `AmazonEKSClusterPolicy`

Weitere Informationen zur Cluster-IAM-Rolle und den AWS verwalteten IAM-Richtlinien finden Sie unter:

- the section called “ AWS verwaltete Richtlinien”
- the section called “Cluster-IAM-Rolle”

Weitere Informationen zum Kubernetes-Zugriff finden Sie unter:

- the section called “Überprüfung von Zugriffsrichtlinien”

Knoten-IAM-Rolle

Die Node-IAM-Rolle ist eine AWS Identity and Access Management (IAM) -Rolle, die von Amazon EKS verwendet wird, um Berechtigungen für Worker-Knoten in Kubernetes-Clustern zu verwalten. Diese Rolle gewährt EC2 Instances, die als Kubernetes-Knoten ausgeführt werden, die erforderlichen Berechtigungen für die Interaktion mit AWS Diensten und Ressourcen. Sie wird mithilfe von EKS-Zugriffseinträgen automatisch mit Kubernetes-RBAC-Berechtigungen konfiguriert.

- Sie müssen dieser Rolle IAM-Richtlinien zuordnen AWS .
- EKS Auto Mode weist dieser Rolle automatisch Kubernetes-RBAC-Berechtigungen zu, indem er EKS-Zugriffseinträge verwendet.
- AWS schlägt vor, diese Rolle `AmazonEKSAutoNodeRole` zu benennen.
- Schlägt AWS vor, im automatischen Modus von EKS eine einzelne Node-IAM-Rolle pro AWS Konto zu erstellen.
- Diese Rolle verfügt über eingeschränkte Berechtigungen. Zu den wichtigsten Berechtigungen gehören das Übernehmen einer Pod-Identity-Rolle und das Abrufen von Images aus ECR.
- AWS schlägt die folgenden AWS verwalteten IAM-Richtlinien vor:

- `AmazonEKSWorkerNodeMinimalPolicy`
- `AmazonEC2ContainerRegistryPullOnly`

Weitere Informationen zur Cluster-IAM-Rolle und zu AWS verwalteten IAM-Richtlinien finden Sie unter:

- the section called “ AWS verwaltete Richtlinien”
- the section called “Knoten-IAM-Rolle”

Weitere Informationen zum Kubernetes-Zugriff finden Sie unter:

- the section called “Überprüfung von Zugriffsrichtlinien”

Servicegebundene Rolle

Amazon EKS verwendet für bestimmte Vorgänge eine serviceverknüpfte Rolle (SLR). Eine servicegebundene Rolle ist eine einzigartige Art von IAM-Rolle, die direkt mit Amazon EKS verknüpft ist. Servicebezogene Rollen sind von Amazon EKS vordefiniert und beinhalten alle Berechtigungen, die der Service benötigt, um andere AWS Services in Ihrem Namen aufzurufen.

AWS erstellt und konfiguriert die Spiegelreflexkamera automatisch. Sie können eine SLR erst löschen, nachdem Sie die zugehörigen Ressourcen gelöscht haben. Dies schützt Ihre Amazon-EKS-Ressourcen, da Sie nicht versehentlich die Zugriffsberechtigung für die Ressourcen entfernen können.

Die SLR-Richtlinie gewährt Amazon EKS die Erlaubnis, zentrale Infrastrukturkomponenten zu beobachten und zu löschen: EC2 Ressourcen (Instances, Netzwerkschnittstellen, Sicherheitsgruppen), ELB-Ressourcen (Load Balancer, Zielgruppen), CloudWatch Funktionen (Protokollierung und Metriken) und IAM-Rollen mit dem Präfix „eks“. Sie ermöglicht auch private Endpunktnetzwerke durch VPC/hosted Zonenzuweisung und beinhaltet Berechtigungen für die EventBridge Überwachung und Bereinigung von mit EKS markierten Ressourcen.

Weitere Informationen finden Sie unter:

- the section called “AWS verwaltete Richtlinie: Amazon EKSService RolePolicy”
- the section called “Serviceverknüpfte Rollenberechtigungen für Amazon EKS”

Benutzerdefinierte AWS Tags für EKS Auto-Ressourcen

Standardmäßig erlauben die verwalteten Richtlinien im Zusammenhang mit dem automatischen Modus von EKS nicht das Anwenden von benutzerdefinierten Tags auf im automatischen Modus bereitgestellte AWS Ressourcen. Wenn Sie benutzerdefinierte Tags auf AWS Ressourcen anwenden möchten, müssen Sie der Cluster-IAM-Rolle zusätzliche Berechtigungen zuweisen, die über ausreichende Berechtigungen verfügen, um Tags auf AWS Ressourcen zu erstellen und zu ändern. Nachfolgend finden Sie ein Beispiel für eine Richtlinie, die uneingeschränkten Zugriff auf Tags gewährt:

Beispiel für eine benutzerdefinierte Tag-Richtlinie anzeigen

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Compute",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateFleet",
                "ec2:RunInstances",
                "ec2:CreateLaunchTemplate"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/eks:eks-cluster-name": "${aws:PrincipalTag/eks:eks-
cluster-name}"
                },
                "StringLike": {
                    "aws:RequestTag/eks:kubernetes-node-class-name": "*",
                    "aws:RequestTag/eks:kubernetes-node-pool-name": "*"
                }
            }
        },
        {
            "Sid": "Storage",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateVolume",
                "ec2:CreateSnapshot"
            ],
```

```
            "Resource": [
                "arn:aws:ec2:*:*:volume/*",
                "arn:aws:ec2:*:*:snapshot/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/eks:eks-cluster-name": "${aws:PrincipalTag/eks:eks-
cluster-name}"
                }
            }
        },
        {
            "Sid": "Networking",
            "Effect": "Allow",
            "Action": "ec2:CreateNetworkInterface",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/eks:eks-cluster-name": "${aws:PrincipalTag/eks:eks-
cluster-name}"
                },
                "StringLike": {
                    "aws:RequestTag/eks:kubernetes-cni-node-name": "*"
                }
            }
        },
        {
            "Sid": "LoadBalancer",
            "Effect": "Allow",
            "Action": [
                "elasticloadbalancing:CreateLoadBalancer",
                "elasticloadbalancing:CreateTargetGroup",
                "elasticloadbalancing:CreateListener",
                "elasticloadbalancing:CreateRule",
                "ec2:CreateSecurityGroup"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/eks:eks-cluster-name": "${aws:PrincipalTag/eks:eks-
cluster-name}"
                }
            }
        },
```

```
        {
            "Sid": "ShieldProtection",
            "Effect": "Allow",
            "Action": [
                "shield:CreateProtection"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/eks:eks-cluster-name": "${aws:PrincipalTag/eks:eks-
cluster-name}"
                }
            }
        },
        {
            "Sid": "ShieldTagResource",
            "Effect": "Allow",
            "Action": [
                "shield:TagResource"
            ],
            "Resource": "arn:aws:shield::*:protection/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/eks:eks-cluster-name": "${aws:PrincipalTag/eks:eks-
cluster-name}"
                }
            }
        }
    ]
}
```

Zugriffsrichtlinien-Referenz

Weitere Informationen zu den von EKS Auto Mode verwendeten Kubernetes-Berechtigungen finden Sie unter the section called “Überprüfung von Zugriffsrichtlinien”.

Weitere Informationen zu VPC-Netzwerke und Load Balancing in EKS Auto Mode

In diesem Thema wird erläutert, wie Sie die Netzwerk- und Load-Balancer-Features der Virtual Private Cloud (VPC) in EKS Auto Mode konfigurieren. Obwohl EKS Auto Mode die meisten Netzwerkkomponenten automatisch verwaltet, können Sie bestimmte Aspekte der

Netzwerkkonfiguration Ihres Clusters dennoch über `NodeClass`-Ressourcen und Load-Balancer-Annotationen anpassen.

Wenn Sie EKS Auto Mode verwenden, AWS verwaltet es die VPC Container Network Interface (CNI) -Konfiguration und die Load Balancer-Bereitstellung für Ihren Cluster. Sie können das Netzwerkverhalten beeinflussen, indem Sie `NodeClass`-Objekte definieren und bestimmte Annotationen auf Ihre Service- und Ingress-Ressourcen anwenden, während Sie gleichzeitig das automatisierte Betriebsmodell beibehalten, das EKS Auto Mode bereitstellt.

Netzwerkfähigkeit

EKS Auto Mode verfügt über eine neue Netzwerkfunktion, die das Netzwerk von Knoten und Pods verwaltet. Sie können es konfigurieren, indem Sie ein `NodeClass`-Kubernetes-Objekt erstellen.

Die Konfigurationsoptionen für das vorherige AWS VPC-CNI gelten nicht für den EKS-Automatikmodus.

Konfiguration des Netzwerks mit einer **`NodeClass`**

Die `NodeClass`-Ressource in EKS Auto Mode ermöglicht es Ihnen, bestimmte Aspekte der Netzwerkfunktionen anzupassen. Über `NodeClass` können Sie Sicherheitsgruppen auswählen, die Platzierung von Knoten in VPC-Subnetzen steuern, SNAT-Richtlinien festlegen, Netzwerkrichtlinien konfigurieren und die Protokollierung von Netzwerkereignissen aktivieren. Dieser Ansatz behält das automatisierte Betriebsmodell von EKS Auto Mode bei und bietet gleichzeitig Flexibilität für die Anpassung des Netzwerks.

Sie können eine `NodeClass` verwenden, um:

- Sicherheitsgruppe für Knoten auswählen
- Steuern, wie Knoten in VPC-Subnetzen platziert werden
- SNAT-Richtlinie für den Knoten auf `random` oder `disabled` festlegen
- Aktivieren Sie die Kubernetes-Netzwerkrichtlinien, einschließlich:
  - Einstellen der Netzwerkrichtlinie auf „Standardmäßig ablehnen“ oder „Standardmäßig zulassen“
  - Aktivieren Sie die Protokollierung von Netzwerkereignissen in einer Datei.
- Trennen Sie den Pod-Datenverkehr vom Knotendatenverkehr, indem Sie Pods an verschiedene Subnetze anfügen.

Erfahren Sie, wie Sie ein Amazon EKS erstellen NodeClass.

Überlegungen

EKS Auto Mode unterstützt:

- EKS-Netzwerkrichtlinien.
- Die `HostPort`- und `HostNetwork`-Optionen für Kubernetes-Pods.
- Knoten und Pods in öffentlichen oder privaten Subnetzen.
- Zwischenspeichern von DNS-Abfragen auf dem Knoten.

EKS Auto Mode unterstützt Folgendes nicht:

- Sicherheitsgruppen pro Pod (SGPP).
- Benutzerdefinierte Netzwerke in `ENIConfig`. Sie können Pods in mehreren Subnetzen platzieren oder sie mit the section called “Subnetz-Auswahl für Pods” exklusiv vom Knoten-Datenverkehr isolieren.
- Warm-IP-, Warm-Präfix- und Warm-ENI-Konfigurationen.
- Konfiguration für minimale IP-Ziele.
- Andere Konfigurationen, die von der AWS Open-Source-VPC CNI unterstützt werden.
- Konfigurationen der Netzwerkrichtlinien, wie beispielsweise die Anpassung des Conntrack-Timers (Standardwert ist 300 Sekunden).
- Netzwerkereignisprotokolle exportieren nach. CloudWatch

Netzwerk-Ressourcenmanagement

Der automatische Modus von EKS kümmert sich um die Verwaltung von Präfixen, IP-Adressen und Netzwerkschnittstellen, indem er die NodeClass Ressourcen für Netzwerkkonfigurationen überwacht. Der Service führt mehrere wichtige Vorgänge automatisch aus:

Präfix-Delegierung

Der automatische Modus von EKS verwendet standardmäßig die Präfix-Delegierung (/28-Präfixe) für Pod-Netzwerke und verwaltet einen vordefinierten Warmpool von IP-Ressourcen, der auf der Grundlage der Anzahl der geplanten Pods skaliert wird. Wenn eine Fragmentierung des Pod-Subnetzes erkannt wird, stellt der automatische Modus sekundäre IP-Adressen (/32) bereit. Aufgrund dieses standardmäßigen Pod-Netzwerkalgorithmus berechnet der automatische Modus die maximale Anzahl von Pods pro Knoten auf der Grundlage der Anzahl ENIs und der IPs unterstützten pro

Instance-Typ (wobei der schlimmste Fall einer Fragmentierung vorausgesetzt wird). Weitere Informationen zu Max ENIs und IPs pro Instance-Typ finden Sie unter Maximale IP-Adressen pro Netzwerkschnittstelle im EC2 Benutzerhandbuch. Bei Instance-Familien der neueren Generation (Nitro v6 und höher) gibt es im Allgemeinen mehr ENIs IPs pro Instance-Typ, und der automatische Modus passt die Berechnung der maximalen Anzahl an Pods entsprechend an.

Für IPv6 Cluster wird nur die Präfix-Delegierung verwendet, und der automatische Modus verwendet immer ein maximales Pod-Limit von 110 Pods pro Knoten.

Abkühlungs-Management

Der Dienst implementiert einen Cooldown-Pool für Präfixe oder sekundäre IPv4 Adressen, die nicht mehr verwendet werden. Nach Ablauf der Ruhephase werden diese Ressourcen wieder an die VPC zurückgegeben. Wenn Pods diese Ressourcen jedoch während der Ruhephase wiederverwenden, werden sie aus dem Abkühlungs-Pool wiederhergestellt.

IPv6 Support

Für IPv6 Cluster stellt EKS Auto Mode ein `/80` IPv6 Präfix pro Knoten auf der primären Netzwerkschnittstelle bereit.

Der Service gewährleistet außerdem die ordnungsgemäße Verwaltung und Garbage Collection aller Netzwerkschnittstellen.

Lastausgleich

Sie konfigurieren AWS Elastic Load Balancer, die vom EKS Auto Mode bereitgestellt werden, mithilfe von Anmerkungen zu Service- und Ingress-Ressourcen.

Für weitere Informationen siehe the section called “Ingress-Klasse erstellen” oder the section called “Service erstellen”.

Überlegungen zum Load Balancing mit EKS Auto Mode

- Der Standard- Zielmodus ist der IP-Modus, nicht der Instance-Modus.
- EKS Auto Mode unterstützt nur den Sicherheitsgruppenmodus für Network Load Balancers.
- AWS unterstützt nicht die Migration von Load Balancern vom selbstverwalteten AWS Load Balancer-Controller zur Verwaltung durch EKS Auto Mode.
- Das Feld `networking.ingress.ipBlock` wird in der `TargetGroupBinding`-Spezifikation nicht unterstützt.

- Wenn Ihre Worker-Knoten benutzerdefinierte Sicherheitsgruppen verwenden (kein `eks-cluster-sg-` -Benennungsmuster), benötigt Ihre Cluster-Rolle zusätzliche IAM-Berechtigungen. Die standardmäßige von EKS verwaltete Richtlinie berechtigt EKS lediglich zur Änderung von Sicherheitsgruppen mit dem Namen `eks-cluster-sg-`. Ohne die Erlaubnis, Ihre benutzerdefinierten Sicherheitsgruppen zu ändern, kann EKS die erforderlichen Eingangsregeln nicht hinzufügen, die es ermöglichen, dass der ALB/NLB Datenverkehr Ihre Pods erreicht.

Überlegungen zu CoreDNS

Der automatische EKS-Modus verwendet nicht die herkömmliche CoreDNS-Bereitstellung, um die DNS-Auflösung innerhalb des Clusters bereitzustellen. Stattdessen verwenden Auto-Mode-Knoten CoreDNS, das als Systemdienst direkt auf jedem Knoten ausgeführt wird. Wenn Sie einen herkömmlichen Cluster in den automatischen Modus umstellen, können Sie die CoreDNS-Bereitstellung aus Ihrem Cluster entfernen, sobald Ihre Workloads auf die Knoten im automatischen Modus verschoben wurden.

> **Important**
>
> Wenn Sie planen, einen Cluster mit Knoten im automatischen Modus und ohne automatischen Modus zu verwalten, müssen Sie die CoreDNS-Bereitstellung beibehalten. Knoten ohne automatischen Modus verlassen sich bei der DNS-Auflösung auf die CoreDNS CoreDNS-Pods, da sie nicht auf den DNS-Dienst auf Knotenebene zugreifen können, den der automatische Modus bereitstellt.

Fehlerbehebung mit EKS Auto Mode

AWS Übernimmt mit dem automatischen Modus von EKS mehr Verantwortung für EC2 Instances in Ihrem AWS Konto. EKS übernimmt die Verantwortung für die Container-Laufzeit auf den Knoten, das Betriebssystem auf den Knoten und bestimmte Controller. Dazu gehören ein Blockspeicher-Controller, ein Load-Balancing-Controller und ein Compute-Controller.

Sie müssen Kubernetes verwenden AWS , um Fehler bei Knoten APIs zu beheben. Sie können:

- Verwenden Sie eine Kubernetes-`NodeDiagnostic`-Ressource, um Knotenprotokolle mithilfe von the section called “Knotenüberwachungsagent” abzurufen. Weitere Schritte finden Sie unter the section called “Knotenprotokolle abrufen”.

- Verwenden Sie den AWS EC2 CLI-Befehl`get-console-output`, um die Konsolenausgabe von Knoten abzurufen. Weitere Schritte finden Sie unter the section called “Rufen Sie mithilfe der AWS EC2 CLI die Konsolenausgabe von einer EC2 verwalteten Instanz ab”.
- Verwenden Sie Debugging-Container von Kubernetes, um Knotenprotokolle abzurufen. Weitere Schritte finden Sie unter the section called “Knotenprotokolle mithilfe von Debug-Containern und der `kubectl`-CLI abrufen”.

> **Note**
>
> Der automatische Modus von EKS verwendet EC2 verwaltete Instanzen. Sie können nicht direkt auf EC2 verwaltete Instanzen zugreifen, auch nicht über SSH.

Möglicherweise treten die folgenden Probleme auf, für die es spezifische Lösungen für EKS-Auto-Mode-Komponenten gibt:

- Pods, die im Status `Pending` hängen geblieben sind und nicht auf Knoten im Automatikmodus geplant werden. Lösungen finden Sie unter the section called “Fehlerbehebung bei fehlgeschlagener Pod-Planung in Knoten im Automatikmodus”.
- EC2 verwaltete Instanzen, die dem Cluster nicht als Kubernetes-Knoten beitreten. Lösungen finden Sie unter the section called “Fehlerbehebung für Knoten, die nicht dem Cluster beitreten”.
- Fehler und Probleme mit `NodePools`, `PersistentVolumes` und `Services`, welche die im EKS Auto Mode enthaltenen Controller verwenden. Lösungen finden Sie unter the section called “Fehlerbehebung für im Automatikmodus enthaltene Controller”.
- Die verbesserte Pod-Sicherheit verhindert die gemeinsame Nutzung von Volumes über Pods hinweg. Lösungen finden Sie unter the section called “Freigabe von Volumes über Pods hinweg”.

Sie können die folgenden Methoden zur Fehlerbehebung bei EKS-Auto-Mode-Komponenten anwenden:

- the section called “Rufen Sie mithilfe der AWS EC2 CLI die Konsolenausgabe von einer EC2 verwalteten Instanz ab”
- the section called “Knotenprotokolle mithilfe von Debug-Containern und der `kubectl`-CLI abrufen”
- the section called “Zeigen Sie die mit dem automatischen Modus von EKS verknüpften Ressourcen in der AWS Konsole an”

- the section called “IAM-Fehler in Ihrem AWS Konto anzeigen”
- the section called “Erkennung von Problemen mit der Knotenkonnektivität mit `VPC Reachability Analyzer`”

Knotenüberwachungsagent

EKS Auto Mode umfasst den Amazon-EKS-Knoten-Überwachungsagent. Mit diesem Agenten können Sie Informationen zur Fehlerbehebung und zum Debugging von Knoten anzeigen. Der Knoten-Überwachungsagent veröffentlicht Kubernetes `events` und Knoten `conditions`. Weitere Informationen finden Sie unter the section called “Knotenintegrität”.

Rufen Sie mithilfe der AWS EC2 CLI die Konsolenausgabe von einer EC2 verwalteten Instanz ab

Dieses Verfahren unterstützt bei der Fehlerbehebung von Problemen beim Systemstart oder auf Kernel-Ebene.

Zunächst müssen Sie die EC2 Instanz-ID der Instanz ermitteln, die Ihrem Workload zugeordnet ist. Verwenden Sie zweitens die AWS CLI, um die Konsolenausgabe abzurufen.

1. Bestätigen Sie, dass Sie `kubectl` installiert und mit Ihrem Cluster verbunden haben
2. (Optional) Namen einer Kubernetes-Bereitstellung für die Auflistung der zugehörigen Pods verwenden

```
kubectl get pods -l app=<deployment-name>
```

3. Verwenden Sie den Namen des Kubernetes-Pods, um die EC2 Instanz-ID des zugehörigen Knotens zu ermitteln.

```
kubectl get pod <pod-name> -o wide
```

4. Verwenden Sie die EC2 Instanz-ID, um die Konsolenausgabe abzurufen.

```
aws ec2 get-console-output --instance-id <instance id> --latest --output text
```

Knotenprotokolle mithilfe von Debug-Containern und der **`kubectl`**-CLI abrufen

Die empfohlene Methode zum Abrufen von Protokollen von einem EKS-Auto-Mode-Knoten ist die Verwendung der `NodeDiagnostic`-Ressource. Diese Schritte finden Sie unter the section called “Knotenprotokolle abrufen”.

Sie können Protokolle jedoch auch live von einer Instance streamen, indem Sie den Befehl `kubectl debug node` verwenden. Dieser Befehl startet einen neuen Pod auf dem Knoten, den Sie debuggen möchten, und den Sie dann interaktiv verwenden können.

1. Starten Sie einen Debug-Container. Der folgende Befehl verwendet `i-0123456789012345`6 für die Instance-ID des Knotens, `-it` weist `tty` zu und fügt `stdin` für die interaktive Verwendung an und verwendet das `sysadmin`-Profil aus der kubeconfig-Datei.

   ```
   kubectl debug node/i-0123456789012345`6 -it --profile=sysadmin --image=public.ecr.aws/
   amazonlinux/amazonlinux:2023
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   Creating debugging pod node-debugger-i-0123456789012345`6-nxb9c with container
    debugger on node i-0123456789012345`6.
   If you don't see a command prompt, try pressing enter.
   bash-5.2#
   ```

2. Von der Shell aus können Sie nun `util-linux-core` installieren, das den Befehl `nsenter` bereitstellt. Verwenden Sie `nsenter`, um den Mount-Namespace von PID 1 (`init`) auf dem Host aufzurufen, und führen Sie den Befehl `journalctl` aus, um Protokolle von `kubelet` zu streamen:

   ```
   yum install -y util-linux-core
   nsenter -t 1 -m journalctl -f -u kubelet
   ```

Aus Sicherheitsgründen installiert das Amazon-Linux-Container-Image standardmäßig nicht viele Binärdateien. Mit dem Befehl `yum whatprovides` können Sie das Paket identifizieren, das installiert werden muss, um eine bestimmte Binärdatei bereitzustellen.

```
yum whatprovides ps
```

```
Last metadata expiration check: 0:03:36 ago on Thu Jan 16 14:49:17 2025.
procps-ng-3.3.17-1.amzn2023.0.2.x86_64 : System and process monitoring utilities
Repo        : @System
Matched from:
Filename    : /usr/bin/ps
Provide    : /bin/ps

procps-ng-3.3.17-1.amzn2023.0.2.x86_64 : System and process monitoring utilities
Repo        : amazonlinux
Matched from:
Filename    : /usr/bin/ps
Provide    : /bin/ps
```

Zeigen Sie die mit dem automatischen Modus von EKS verknüpften Ressourcen in der AWS Konsole an

Sie können die AWS Konsole verwenden, um den Status der Ressourcen anzuzeigen, die Ihrem EKS-Auto-Mode-Cluster zugeordnet sind.

- EBS-Volumes
  - Anzeigen der EKS-Auto-Mode-Volumes durch Suche nach dem Tag-Schlüssel `eks:eks-cluster-name`
- Load Balancer
  - Anzeigen der Load Balancer in EKS Auto Mode durch Suche nach dem Tag-Schlüssel `eks:eks-cluster-name`
- EC2 Instances
  - Anzeigen der EKS-Auto-Mode-Instances durch Suche nach dem Tag-Schlüssel `eks:eks-cluster-name`

IAM-Fehler in Ihrem AWS Konto anzeigen

1. Navigieren Sie zur Konsole CloudTrail
2. Auswahl von „Ereignisverlauf“ im linken Navigationsbereich
3. Fehlercode-Filter anwenden:

- AccessDenied
- UnauthorizedOperation
- InvalidClientTokenId

Suchen Sie nach Fehlern im Zusammenhang mit Ihrem EKS-Cluster. Verwenden Sie die Fehlermeldungen, um Ihre EKS-Zugriffseinträge, die Cluster-IAM-Rolle oder die Knoten-IAM-Rolle zu aktualisieren. Möglicherweise müssen Sie diesen Rollen eine neue Richtlinie mit Berechtigungen für den EKS-Automodus zuordnen.

Fehlerbehebung bei fehlgeschlagener Pod-Planung in Knoten im Automatikmodus

Wenn Pods im `Pending`-Status verbleiben und nicht auf einen Automatikmodus-Knoten geplant werden, überprüfen Sie, ob Ihr Pod- oder Bereitstellungs-Manifest ein `nodeSelector` enthält. Wenn `nodeSelector` vorhanden ist, stellen Sie sicher, dass `eks.amazonaws.com/compute-type: auto` verwendet wird, um auf Knoten geplant zu werden, die von EKS Auto Mode erstellt wurden. Weitere Informationen zu den Knotenbezeichnungen, die von EKS Auto Mode verwendet werden, finden Sie unter the section called “Kontrollbereitstellung”.

Fehlerbehebung für Knoten, die nicht dem Cluster beitreten

Der automatische EKS-Modus konfiguriert automatisch neue EC2 Instances mit den richtigen Informationen für den Beitritt zum Cluster, einschließlich des Cluster-Endpunkts und der Cluster-Zertifizierungsstelle (CA). Es kann jedoch vorkommen, dass diese Instances dennoch nicht als Knoten dem EKS-Cluster beitreten können. Führen Sie die folgenden Befehle aus, um Instances, die nicht dem Cluster beigetreten sind, zu identifizieren:

1. Führen Sie `kubectl get nodeclaim` aus, um nach `NodeClaims` zu suchen, die `Ready = False` sind.

```
kubectl get nodeclaim
```

2. Führen Sie `kubectl describe nodeclaim <node_claim>` aus und überprüfen Sie unter Status, ob Probleme vorliegen, die den Knoten daran hindern, dem Cluster beizutreten.

```
kubectl describe nodeclaim <node_claim>
```

Gängige Fehlermeldungen:

`Error getting launch template configs`

Dieser Fehler kann auftreten, wenn Sie benutzerdefinierte Tags im `NodeClass` mit den Standardberechtigungen der Cluster-IAM-Rolle festlegen. Siehe the section called “Identität und Zugriff”.

`Error creating fleet`

Möglicherweise liegt ein Autorisierungsproblem beim `RunInstances` Aufrufen des Aufrufs über die EC2 API vor. AWS CloudTrail Suchen Sie nach Fehlern und the section called “IAM-Rolle für Cluster im Automatikmodus” nach den erforderlichen IAM-Berechtigungen.

Erkennung von Problemen mit der Knotenkonnektivität mit **`VPC Reachability Analyzer`**

> **Note**
>
> Für jede Analyse, die mit dem VPC Reachability Analyzer durchgeführt wird, wird Ihnen eine Gebühr berechnet. Details zu den Preisen finden Sie unter Amazon-VPC-Preise.

Ein Grund dafür, dass eine Instance dem Cluster nicht beigetreten ist, ist ein Problem mit der Netzwerkkonnektivität, das sie daran hindert, den API-Server zu erreichen. Zur Diagnose dieses Problems können Sie mit dem VPC Reachability Analyzer die Konnektivität zwischen einem Knoten, der dem Cluster nicht beitreten kann, und dem API-Server analysieren. Sie benötigen zwei Informationen:

- Instance-ID eines Knotens, der dem Cluster nicht beitreten kann
- IP-Adresse des Kubernetes-API-Server-Endpunkts

Um die Instanz-ID zu erhalten, müssen Sie einen Workload auf dem Cluster erstellen, damit der automatische Modus von EKS eine EC2 Instance startet. Dadurch wird auch ein `NodeClaim`-Objekt in Ihrem Cluster erstellt, das die Instance-ID enthält. Führen Sie `kubectl get nodeclaim -o yaml` aus, um alle `NodeClaims` in Ihrem Cluster auszudrucken. Jedes `NodeClaim` enthält die Instance-ID als Feld und erneut in der providerID:

```
kubectl get nodeclaim -o yaml
```

Eine Beispielausgabe sieht wie folgt aus.

```
    nodeName: i-01234567890123456
    providerID: aws:///us-west-2a/i-01234567890123456
```

Sie können Ihren Kubernetes-API-Server-Endpunkt ermitteln, indem Sie `kubectl get endpoint kubernetes -o yaml` ausführen. Die Adressen befinden sich im Adressfeld:

```
kubectl get endpoints kubernetes -o yaml
```

Eine Beispielausgabe sieht wie folgt aus.

```
apiVersion: v1
kind: Endpoints
metadata:
  name: kubernetes
  namespace: default
subsets:
- addresses:
  - ip: 10.0.143.233
  - ip: 10.0.152.17
  ports:
  - name: https
    port: 443
    protocol: TCP
```

Mit diesen beiden Informationen können Sie die S-Analyse durchführen. Navigieren Sie zunächst zum VPC Reachability Analyzer in AWS-Managementkonsole.

1. Klicken Sie auf „Pfad erstellen und analysieren“.
2. Geben Sie einen Namen für die Analyse an (z. B. „Fehler beim Zusammenschluss von Knoten”).
3. Wählen Sie für „Quelltyp“ die Option „Instances“ aus
4. Geben Sie die Instance-ID des fehlerhaften Knotens als „Quelle“ ein
5. Wählen Sie für „Pfadziel“ die Option „IP-Adresse“ aus
6. Geben Sie eine der IP-Adressen für den API-Server als „Zieladresse“ ein
7. Erweitern Sie den Abschnitt „Zusätzliche Konfiguration des Paket-Headers“.

8. Geben Sie als „Zielport“ 443 ein
9. Wählen Sie „Protokoll“ als TCP, falls dies noch nicht ausgewählt ist
10. Klicken Sie auf „Pfad erstellen und analysieren“.
11. Der Test kann einige Minuten in Anspruh nehmen. Wenn die Analyseergebnisse eine fehlgeschlagene Erreichbarkeit anzeigen, wird angegeben, wo der Fehler im Netzwerkpfad aufgetreten ist, sodass Sie das Problem beheben können.

Freigabe von Volumes über Pods hinweg

Die EKS-Automodus-Knoten sind SELinux im Modus „Enforcing“ konfiguriert, was für mehr Isolation zwischen Pods sorgt, die auf demselben Knoten ausgeführt werden. Wenn diese Option aktiviert SELinux ist, erhalten die meisten nicht privilegierten Pods automatisch ein eigenes Sicherheitslabel (Multi-Category Security, MCS). Dieses MCS-Label ist für jeden Pod einzigartig und soll sicherstellen, dass ein Prozess in einem Pod keinen Prozess in einem anderen Pod oder auf dem Host manipulieren kann. Selbst wenn ein gekennzeichneter Pod als Root ausgeführt wird und Zugriff auf das Host-Dateisystem hat, kann er keine Dateien manipulieren, keine vertraulichen Systemaufrufe im Host ausführen, nicht auf die Container-Laufzeitumgebung zugreifen und kein geheimes Schlüsselmaterial von Kubelet abrufen.

Aus diesem Grund können Probleme auftreten, wenn Sie versuchen, Daten zwischen Pods auszutauschen. Beispielsweise erlaubt ein `PersistentVolumeClaim` mit einem Zugriffsmodus von `ReadWriteOnce` immer noch nicht, dass mehrere Pods gleichzeitig auf das Volume zugreifen.

Um diese Freigabe zwischen Pods zu ermöglichen, können Sie das `seLinuxOptions` des Pods verwenden, um auf diesen Pods dasselbe MCS-Label zu konfigurieren. In diesem Beispiel weisen wir dem Pod die drei Kategorien `c123,c456,c789` zu. Dies steht nicht im Widerspruch zu den Kategorien, die den Pods auf dem Knoten automatisch zugewiesen werden, da ihnen nur zwei Kategorien zugewiesen werden.

```
securityContext:
  seLinuxOptions:
    level: "s0:c123,c456,c789"
```

Anzeige von Karpenter-Ereignissen in den Protokollen der Steuerebene

Für EKS-Cluster mit aktivierten Steuerebenen-Protokollen können Sie durch Abfragen der Protokolle Einblicke in die Aktionen und Entscheidungsprozesse von Karpenter gewinnen. Dies kann besonders

nützlich sein, um Probleme in EKS Auto Mode im Zusammenhang mit der Bereitstellung, Skalierung und Beendigung von Knoten zu beheben. Verwenden Sie die folgende Logs Insights-Abfrage, um Ereignisse im Zusammenhang mit Karpenter anzuzeigen: CloudWatch

```
fields @timestamp, @message
| filter @logStream like /kube-apiserver-audit/
| filter @message like 'DisruptionBlocked'
or @message like 'DisruptionLaunching'
or @message like 'DisruptionTerminating'
or @message like 'DisruptionWaitingReadiness'
or @message like 'Unconsolidatable'
or @message like 'FailedScheduling'
or @message like 'NoCompatibleInstanceTypes'
or @message like 'NodeRepairBlocked'
or @message like 'Disrupted'
or @message like 'Evicted'
or @message like 'FailedDraining'
or @message like 'TerminationGracePeriodExpiring'
or @message like 'TerminationFailed'
or @message like 'FailedConsistencyCheck'
or @message like 'InsufficientCapacityError'
or @message like 'UnregisteredTaintMissing'
or @message like 'NodeClassNotReady'
| sort @timestamp desc
```

Diese Abfrage filtert nach bestimmten Karpenter-bezogenen Ereignissen in den Überwachungsprotokollen des Kube-API-Servers. Zu den Ereignissen zählen verschiedene Störungszustände, Planungsfehler, Kapazitätsprobleme und knotenbezogene Probleme. Durch die Analyse dieser Protokolle erhalten Sie ein besseres Verständnis für Folgendes:

- Warum Karpenter bestimmte Maßnahmen ergreift.
- Alle Probleme, die eine ordnungsgemäße Bereitstellung, Skalierung oder Beendigung von Knoten verhindern.
- Mögliche Kapazitäts- oder Kompatibilitätsprobleme mit Instance-Typen.
- Ereignisse im Knoten-Lebenszyklus wie Störungen, Bereinigungen oder Beendigungen.

So verwenden Sie diese Abfrage:

1. Navigieren Sie zur Konsole CloudWatch

2. Wählen Sie im linken Navigationsbereich „Protokolleinblicke“ aus.
3. Wählen Sie die Protokollgruppe für die Protokolle der Steuerebene Ihres EKS-Clusters aus
4. Fügen Sie die Abfrage in den Abfrage-Editor ein
5. Passen Sie den Zeitraum nach Bedarf an
6. Run the query

Die Ergebnisse zeigen Ihnen eine Zeitleiste mit Karpenter-bezogenen Ereignissen, die Ihnen bei der Fehlerbehebung und beim Verständnis des Verhaltens von EKS Auto Mode in Ihrem Cluster behilflich sein können. Um die Aktionen von Karpenter auf einem bestimmten Knoten zu überprüfen, können Sie den folgenden Zeilenfilter mit der Instance-ID zur oben genannten Abfrage hinzufügen:

```
|filter @message like /[.replaceable]`i-12345678910123456`/
```

> **Note**
>
> Um diese Abfrage zu verwenden, muss die Protokollierung der Steuerebene in Ihrem EKS-Cluster aktiviert sein. Falls Sie dies noch nicht getan haben, lesen Sie the section called “Protokolle der Steuerebene”.

Fehlerbehebung für im Automatikmodus enthaltene Controller

Bei Problemen mit einem Controller sollten Sie Folgendes prüfen:

- Ob die dem Controller zugeordneten Ressourcen korrekt formatiert und gültig sind.
- Wenn die AWS IAM- und Kubernetes-RBAC-Ressourcen für Ihren Cluster ordnungsgemäß konfiguriert sind. Weitere Informationen finden Sie unter the section called “Identität und Zugriff”.

Zugehörige Ressourcen

In diesen Artikeln von AWS re:POST finden Sie weitere Schritte zur Fehlerbehebung:

- Wie behebt man häufige Skalierungsprobleme in EKS Auto Mode?
- Wie behebt man Probleme bei der Bereitstellung benutzerdefinierter Knoten-Pools und Knotenklassen in Amazon EKS Auto Mode?

- Wie behebt man Probleme mit integrierten Knoten-Pools in EKS Auto Mode, deren Status unbekannt ist?

Versionshinweise zu EKS Auto Mode überprüfen

Diese Seite dokumentiert Aktualisierungen von Amazon EKS Auto Mode. Auf dieser Seite finden Sie regelmäßig Ankündigungen zu neuen Features, Fehlerbehebungen, bekannten Problemen und veralteten Funktionen.

Um Benachrichtigungen über alle Änderungen an den Quelldateien dieser spezifischen Dokumentationsseite zu erhalten, können Sie die folgende URL mit einem RSS-Reader abonnieren:

```
https://github.com/awsdocs/amazon-eks-user-guide/commits/mainline/latest/ug/automode/auto-change.adoc.atom
```

19. Dezember 2025

Feature: Unterstützung für den sekundären IP-Modus hinzugefügt, der sekundäre IP-Adressen anstelle von Präfixen für Auto-Knoten bereitstellt. In diesem Modus wird eine einzige sekundäre IP als Minimal beibehalten IPTarget und IP-Ressourcen für Kunden eingespart, die keine weiteren sekundären IP-Adressen IPs oder Präfixe aufwärmen müssen. Weitere Informationen erhalten Sie unter the section called “Knotenklassen-Spezifikation” und the section called “Sekundärer IP-Modus für Pods”.

19. November 2025

Funktion: Paralleles Ziehen und Entpacken von Seekable OCI (SOCI) für Instances der G-, P- und Trn-Familie mit lokalem Speicher aktiviert. NVMe SOCI Parallel Pull and Unpack wird für diese Instance-Familien mit EKS Auto Mode immer verwendet, und es sind keine Konfigurationsänderungen erforderlich, um ihn zu aktivieren. Weitere Informationen zu SOCI finden Sie im Launch-Blog.

19. November 2025

Feature: Es wurde Unterstützung für Knotenpools mit statischer Kapazität hinzugefügt, die eine feste Anzahl von Knoten verwalten. Weitere Informationen finden Sie unter the section called “Knotenpools mit statischer Kapazität”.

23. Oktober 2025

Feature: Benutzer mit Clustern in US-Regionen können jetzt beantragen, FIPS-kompatibel zu verwenden, AMIs indem sie dies `spec.advancedSecurity.fips` in ihrer NodeClass Definition angeben.

1. Oktober 2025

Feature: Der automatische EKS-Modus unterstützt jetzt die Bereitstellung von Knoten in AWS Local Zones. Weitere Informationen finden Sie unter the section called “Local Zones verwenden”.

30. September 2025

Feature: Unterstützung für InstanceProfile wurde hinzugefügt, NodeClass `spec.instanceProfile` was sich gegenseitig aus dem `spec.role` Feld ausschließt.

29. September 2025

DRA wird derzeit nicht vom EKS Auto Mode unterstützt.

10. September 2025

Aufgabe: Vom Auto-Mode-Compute-Controller ausgelöste Ereignisse verwenden jetzt den Namen `eks-auto-mode/compute` statt `karpenter`.

24. August 2025

Fehlerbehebung: VPCs Die Verwendung eines DHCP-Optionssatzes mit einem benutzerdefinierten Domainnamen, der Großbuchstaben enthielt, führte dazu, dass Knoten dem Cluster nicht beitreten konnten, da ein ungültiger Hostname generiert wurde. Dieses Problem wurde behoben, und Domain-Namen mit Großbuchstaben funktionieren nun korrekt.

15. August 2025

Fehlerbehebung: Der Pod Identity Agent überwacht jetzt nur noch die IPv4 Link Local-Adresse in einem IPv4 EKS-Cluster, um Probleme zu vermeiden, bei denen der Pod die Adresse nicht erreichen kann. IPv6

06. August 2025

Feature: Es wurde eine neue Konfiguration hinzugefügt NodeClass `spec.advancedNetworking.associatePublicIPAddress`, mit der verhindert werden kann, dass öffentliche IP-Adressen EKS-Automodus-Knoten zugewiesen werden

30. Juni 2025

Feature: Der Auto-Modus verwendet NodeClass jetzt den konfigurierten benutzerdefinierten KMS-Schlüssel, um zusätzlich zum Datenvolumen auch das schreibgeschützte Root-Volume der Instanz zu verschlüsseln. read/write Zuvor wurde der benutzerdefinierte KMS-Schlüssel nur zum Verschlüsseln des Daten-Volumes verwendet.

20. Juni 2025

Funktion: Support für die Steuerung der Bereitstellung von Workloads in EC2 On-Demand-Kapazitätsreservierungen (ODCRs). Dadurch wird der optionale Schlüssel `capacityReservationSelectorTerms` hinzugefügt NodeClass, sodass Sie explizit steuern können, welche ODCRs Workloads verwenden. Weitere Informationen finden Sie unter the section called “Steuerung von Kapazitätsreservierungen”.

13. Juni 2025

Feature: Unterstützung für separate Pod-Subnetze in `NodeClass`. Dadurch werden die optionalen Schlüssel `` `podSubnetSelectorTerms `` und `podSecurityGroupSelectorTerms` hinzugefügt, um die Subnetze und Sicherheitsgruppen für die Pods festzulegen. Weitere Informationen finden Sie unter the section called “Subnetz-Auswahl für Pods”.

30. April 2025

Feature: Unterstützung für weitergeleitete Netzwerk-Proxys in `NodeClass`. Dadurch wird der optionale Schlüssel `advancedNetworking` zum Festlegen Ihres HTTPS-Proxys hinzugefügt. Weitere Informationen finden Sie unter the section called “Knotenklassen-Spezifikation”.

18. April 2025

Feature: Unterstützung für die Auflösung von .local-Domains (in der Regel für Multicast-DNS reserviert) über Unicast-DNS.

11. April 2025

Feature: `certificateBundles` und `ephemeralStorage.kmsKeyID` zu `NodeClass` hinzugefügt. Weitere Informationen finden Sie unter the section called “Knotenklassen-Spezifikation”.

Feature: Verbesserte Image-Abrufgeschwindigkeit, insbesondere für Instance-Typen mit lokalem Instance-Speicher, welche die schnellere Image-Dekomprimierung nutzen können.

Fehlerbehebung: Es wurde ein Race-Problem behoben FailedCreatePodSandBox , das zu einem Fehler beim Wählen führte: dial tcp 127.0.0. 1:50051: connect: connection failed to, was manchmal dazu führte, dass Pods direkt beim Start auf einen Node planten.

4. April 2025

Feature: `registryPullQPS` von 5 auf 25 und `registryBurst` von 10 auf 50 erhöhen, um die vom Client erzwungene Drosselung des Image-Pulls zu reduzieren (`Failed to pull image xyz: pull QPS exceeded`)

31. März 2025

Fehlerbehebung: Behebt ein Problem, bei dem, wenn ein Kern-DNS-Pod in einem Knoten im Automatikmodus ausgeführt wird, DNS-Anfragen von Pods auf dem Knoten diesen Kern-DNS-Pod anstelle des lokalen DNS-Servers des Knotens erreichen. DNS-Anfragen von Pods in einem Knoten im Automatikmodus werden nun stets an den lokalen DNS-Server des Knotens weitergeleitet.

21. März 2025

Fehlerbehebung: Knoten im Automatikmodus lösen `kube-dns.kube-system.svc.cluster.local` nun korrekt auf, wenn kein `kube-dns`-Service im Cluster installiert ist. Behebt GitHub Problem #2546.

14. März 2025

Feature: `IPv4`-Ausgang in `IPv6`-Clustern aktiviert. Der `IPv4`-Datenverkehr, der aus `IPv6`-Clustern im Automatikmodus ausgegeben wird, wird nun automatisch in die `v4`-Adresse der primären ENI des Knotens übersetzt.

Lebenszyklus und Konfiguration des Amazon-EKS-Clusters

Dieser Abschnitt enthält detaillierte Anleitungen zum gesamten Lebenszyklus-Management von Kubernetes-Clustern und zu verschiedenen Möglichkeiten, diese zu konfigurieren. Er behandelt die Erstellung von Clustern, die Überwachung des Cluster-Zustands, die Aktualisierung von Kubernetes-Versionen und das Löschen von Clustern. Es enthält auch fortgeschrittene Themen zur Konfiguration des Endpunktzugriffs, zur enable/disable Windows-Unterstützung, zur Einrichtung privater Cluster, zur Implementierung von Autoscaling und zur Verbesserung der Ausfallsicherheit durch Zonenverschiebung für die Verkehrsumleitung in Multi-AZ-Setups.

Themen

- Einen Cluster für Amazon EKS Auto Mode erstellen
- Amazon-EKS-Cluster erstellen
- Von Amazon EKS bereitgestellte Kontrollebene
- Vorbereitung auf Kubernetes-Versionsupgrades und Beheben von Fehlkonfigurationen mit Cluster-Einblicken
- Aktualisierung des vorhandenen Clusters auf die neue Kubernetes-Version
- Einen Cluster löschen
- Cluster-API-Server-Endpunkt
- Bereitstellung von Windows-Knoten in EKS-Clustern
- Windows-Support deaktivieren
- Bereitstellung privater Cluster mit eingeschränktem Internetzugang
- Skalierung von Cluster-Computing mit Karpenter und Cluster Autoscaler
- Weitere Informationen zur Zonenverschiebung von Amazon Application Recovery Controller (ARC) in Amazon EKS
- Aktivierung der EKS-Zonenverschiebung zur Vermeidung beeinträchtigter Availability Zones

Einen Cluster für Amazon EKS Auto Mode erstellen

Dieses Thema enthält detaillierte Anweisungen zum Erstellen eines Clusters für Amazon EKS Auto Mode mit erweiterten Konfigurationsoptionen. Es behandelt Voraussetzungen, Netzwerkoptionen und Add-On-Konfigurationen. Der Prozess umfasst die Einrichtung von IAM-Rollen, die Konfiguration

von Cluster-Einstellungen, die Festlegung von Netzwerkparametern und die Auswahl von Add-Ons. Benutzer können Cluster entweder mit der AWS-Managementkonsole oder der AWS CLI erstellen, wobei step-by-step Anleitungen für beide Methoden bereitgestellt werden.

Benutzer, die einen weniger komplexen Einrichtungsprozess wünschen, finden nachfolgend vereinfachte Schritte zur Cluster-Erstellung:

- the section called “eksctl CLI”
- the section called “ AWS CLI”
- the section called “Management-Konsole”

Dieser Leitfaden zur erweiterten Konfiguration richtet sich an Benutzer, die eine differenziertere Kontrolle über die Einrichtung ihres EKS-Auto-Mode-Clusters benötigen und mit den Konzepten und Anforderungen von Amazon EKS vertraut sind. Bevor Sie mit der erweiterten Konfiguration fortfahren, stellen Sie sicher, dass Sie alle Voraussetzungen erfüllt haben und die Netzwerk- und IAM-Anforderungen für EKS-Auto-Mode-Cluster gründlich verstanden haben.

EKS Auto Mode erfordert zusätzliche IAM-Berechtigungen. Weitere Informationen finden Sie unter:

- the section called “IAM-Rollen für Cluster in EKS Auto Mode”
- the section called “Identität und Zugriff”

> **Note**
>
> Wenn Sie einen Cluster ohne EKS Auto Mode erstellen möchten, lesen Sie the section called “Erstellen eines Clusters”.
> Dieses Thema behandelt die erweiterte Konfiguration. Wenn Sie den EKS Auto Mode verwenden möchten, lesen Sie the section called “Cluster erstellen”.

Voraussetzungen

- Eine vorhandene VPC und Subnetze, die die Amazon-EKS-Anforderungen erfüllen. Bevor Sie einen Cluster für den Produktionseinsatz bereitstellen, empfehlen wir, dass Sie sich ein umfassendes Verständnis der VPC- und Subnetzanforderungen aneignen. Wenn Sie keine VPC und keine Subnetze haben, können Sie diese mithilfe einer von Amazon EKS bereitgestellten AWS CloudFormation Vorlage erstellen.

- Das `kubectl`-Befehlszeilen-Tool ist auf Ihrem Gerät oder in der AWS CloudShell installiert. Die Version kann mit der Kubernetes-Version Ihres Clusters identisch sein oder bis zu einer Nebenversion älter oder neuer sein. Wenn Ihre Clusterversion beispielsweise `1.29` ist, können Sie `kubectl`-Version `1.28`, `1.29`, oder `1.30` damit verwenden. Informationen zum Installieren oder Aktualisieren von `kubectl` finden Sie unter the section called “kubectl und eksctl einrichten”.
- Version `2.12.3` oder höher oder Version `1.27.160` oder höher der auf Ihrem Gerät installierten und konfigurierten AWS Befehlszeilenschnittstelle (AWS CLI) oder AWS CloudShell. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version`. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle.
- Ein IAM-Prinzipal mit Berechtigungen zum Erstellen und Ändern von EKS- und IAM-Ressourcen.

Cluster erstellen — AWS Konsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie Cluster hinzufügen und dann Erstellen aus.
3. Wählen Sie unter Konfigurationsoptionen die Option Benutzerdefinierte Konfiguration aus.
   - Dieses Thema behandelt die benutzerdefinierte Konfiguration. Informationen zur Schnellkonfiguration finden Sie unter the section called “Management-Konsole”.
4. Bestätigen Sie, dass EKS Auto Mode verwenden aktiviert ist.
   - Dieses Thema behandelt die Erstellung von Clustern mit EKS Auto Mode. Weitere Informationen zum Erstellen von Clustern ohne EKS Auto Mode finden Sie unter the section called “Erstellen eines Clusters”.
5. Füllen Sie auf der Seite Configure cluster (Cluster konfigurieren) die folgenden Felder aus:
   - Name – Ein Name für Ihren Cluster. Der Name darf nur alphanumerische Zeichen (Groß-/Kleinschreibung beachten), Bindestriche und Unterstriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein.
   - Cluster-IAM-Rolle — Wählen Sie die Amazon EKS-Cluster-IAM-Rolle aus, die Sie erstellt haben, damit die Kubernetes-Steuerebene AWS Ressourcen in Ihrem Namen verwalten kann. Wenn Sie zuvor noch keine Cluster-IAM-Rolle für den EKS Auto Mode erstellt haben, klicken Sie auf die Schaltfläche Empfohlene Rolle erstellen, um die Rolle mit den erforderlichen Berechtigungen in der IAM-Konsole zu erstellen.

- Kubernetes-Version – Die Kubernetes-Version, die für den Cluster verwendet wird. Wir empfehlen, die frühere Version auszuwählen, es sei denn, Sie benötigen eine ältere Version.
- Upgrade-Richtlinie – Die Kubernetes-Versionsrichtlinie, die Sie für Ihren Cluster festlegen möchten. Wenn Ihr Cluster nur mit einer Standard-Supportversion ausgeführt werden soll, wählen Sie Standard. Wenn Ihr Cluster nach Ablauf des Standard-Supports für eine Version in den erweiterten Support wechseln soll, wählen Sie Erweitert. Wenn Sie eine Kubernetes-Version auswählen, für die derzeit der erweiterte Support gilt, können Sie den Standard-Support nicht als Option auswählen.

6. Geben Sie im Abschnitt Auto Mode Compute der Seite für die Cluster-Konfiguration die folgenden Felder ein:
   - Knoten-Pools – Legen Sie fest, ob Sie die integrierten Knoten-Pools verwenden möchten. Weitere Informationen finden Sie unter the section called “Integrierte Knoten-Pool überprüfen”.
   - Knoten-IAM-Rolle – Wenn Sie einen der integrierten Knoten-Pools aktivieren, müssen Sie eine Knoten-IAM-Rolle auswählen. EKS Auto Mode weist diese Rolle neuen Knoten zu. Dieser Wert kann nach der Erstellung des Clusters nicht mehr geändert werden. Falls Sie noch keine Knoten-IAM-Rolle für EKS Auto Mode erstellt haben, klicken Sie auf die Schaltfläche „Empfohlene Rolle erstellen“, um die Rolle mit den erforderlichen Berechtigungen zu erstellen. Weitere Informationen über diese Rolle finden Sie unter the section called “Identität und Zugriff”.
7. Geben Sie im Abschnitt Cluster-Zugriff der Seite „Cluster konfigurieren“ die folgenden Felder ein:
   - Administrator-Zugriff für Bootstrap-Cluster – Der Cluster-Ersteller ist automatisch ein Kubernetes-Administrator. Wenn Sie dies deaktivieren möchten, wählen Sie Cluster-Administratorzugriff verbieten aus.
   - Cluster-Authentifizierungsmodus – EKS Auto Mode erfordert EKS-Zugriffseinträge, den EKS-API-Authentifizierungsmodus. Sie können den **`ConfigMap`** Authentifizierungsmodus optional aktivieren, indem Sie EKS-API und auswählen. ConfigMap
8. Geben Sie die restlichen Felder auf der Seite für die Cluster-Konfiguration ein:
   - Secrets-Verschlüsselung – (Optional) Aktivieren Sie die Secrets-Verschlüsselung von Kubernetes-Secrets mithilfe eines KMS-Schlüssels. Sie können diese auch aktivieren, nachdem Sie Ihren Cluster erstellt haben. Stellen Sie vor Aktivierung dieser Funktion sicher, dass Sie mit den Informationen unter the section called “Geheime Verschlüsselung aktivieren” vertraut sind.
   - ARC-Zonenverschiebung – EKS Auto Mode unterstützt keine Arc-Zonenverschiebung.
   - Tags – (Optional) Fügen Sie Ihrem Cluster beliebige Tags hinzu. Weitere Informationen finden Sie unter the section called “Markieren Ihrer -Ressourcen”.

Wenn Sie mit dieser Seite fertig sind, wählen Sie Weiter aus.

9. Wählen Sie auf der Seite Specify networking (Netzwerk angeben) die Werte für die folgenden Felder aus:

   - VPC – Wählen Sie eine vorhandene VPC aus, die die Anforderungen für Amazon EKS VPC erfüllt, um Ihren Cluster zu erstellen. Bevor Sie sich für eine VPC entscheiden, empfehlen wir, sich mit allen Anforderungen und Überlegungen in the section called “VPC- und Subnetz-Anforderungen” vertraut zu machen. Sie können nach der Cluster-Erstellung nicht mehr ändern, welche VPC Sie verwenden möchten. Wenn keine aufgeführt VPCs sind, müssen Sie zuerst eine erstellen. Weitere Informationen finden Sie unter the section called “Erstellen einer VPC”.
   - Subnetze – Standardmäßig sind alle im vorherigen Feld angegebenen Subnetze in der VPC vorausgewählt. Sie müssen mindestens zwei auswählen.

     Die von Ihnen gewählten Subnetze müssen die Anforderungen für Amazon-EKS-Subnetze erfüllen. Bevor Sie Subnetze auswählen, empfehlen wir, dass Sie sich mit allen Anforderungen und Überlegungen für Amazon-EKS-VPCs und -Subnetze vertraut machen.

     Security groups (Sicherheitsgruppen) – (Optional) Geben Sie eine oder mehrere Sicherheitsgruppen an, die Amazon EKS den erstellten Netzwerkschnittstellen zuordnen soll.

     Unabhängig davon, ob Sie Sicherheitsgruppen wählen oder nicht, erstellt Amazon EKS eine Sicherheitsgruppe, die die Kommunikation zwischen Ihrem Cluster und Ihrer VPC ermöglicht. Amazon EKS verknüpft diese Sicherheitsgruppe und alle, die Sie wählen, mit den erstellten Netzwerkschnittstellen. Weitere Informationen zu der Cluster-Sicherheitsgruppe, die Amazon EKS erstellt, finden Sie unter the section called “Anforderungen an Sicherheitsgruppen”. Sie können die Regeln in der von Amazon EKS erstellten Cluster-Sicherheitsgruppe ändern.
   - Cluster-IP-Adressfamilie wählen — Sie können entweder IPv4und wählen IPv6.

     Kubernetes weist Pods und Services standardmäßig `IPv4`-Adressen zu. Bevor Sie sich entscheiden, die `IPv6`-Familie zu verwenden, stellen Sie sicher, dass Sie mit allen Überlegungen und Anforderungen in den VPC-Anforderungen und -Überlegungen, the section called “Subnetz-Anforderungen und -Überlegungen”-, the section called “Anforderungen an Sicherheitsgruppen”- und the section called “IPv6”-Themen vertraut sind. Wenn Sie die `IPv6`-Familie auswählen, können Sie keinen Adressbereich für Kubernetes festlegen, aus dem `IPv6`-Serviceadressen zugewiesen werden sollen, wie dies bei der `IPv4`-Familie möglich ist. Kubernetes weist Serviceadressen aus dem eindeutigen lokalen Adressbereich (`fc00::/7`) zu.

- (Optional) Wählen Sie Configure Kubernetes Service IP address range (IP-Adressbereich des Kubernetes-Diensts konfigurieren) und geben Sie einen **IPv4**-Bereich für den Service an.

  Durch die Angabe Ihres eigenen Bereichs können Sie Konflikte zwischen Kubernetes-Services und anderen Netzwerken vermeiden, die mit Ihrer VPC verbunden sind. Geben Sie einen Bereich in CIDR-Notation ein. Beispiel: `10.2.0.0/16`.

  Der CIDR-Block muss die folgenden Anforderungen erfüllen:

  - Verwendet einen der folgenden Bereiche: `10.0.0.0/8`, `172.16.0.0/12`, oder `192.168.0.0/16`.
  - Hat eine Mindestgröße von `/24` und eine maximale Größe von `/12`.
  - Überschneidet sich nicht mit dem Bereich der VPC für Ihre Amazon-EKS-Ressourcen.

  Sie können diese Option nur angeben, wenn Sie die `IPv4`-Adressfamilie verwenden, und nur bei Erstellen des Clusters. Wenn Sie dies nicht angeben, weist Kubernetes Service-IP-Adressen aus den CIDR-Blöcken `10.100.0.0/16` oder `172.20.0.0/16` zu.

- Wählen Sie eine Option für den Cluster-Endpunktzugriff aus. Nachdem Ihr Cluster erstellt wurde, können Sie diese Option ändern. Bevor Sie eine nicht standardmäßige Option auswählen, sollten Sie sich mit den Optionen und deren Auswirkungen vertraut machen. Weitere Informationen finden Sie unter the section called “Cluster-Endpunktzugriff”.

  Wenn Sie mit dieser Seite fertig sind, wählen Sie Weiter aus.

10. (Optional) Auf der Seite Beobachtbarkeit konfigurieren können Sie Metriken und Optionen zur Steuerebenen-Protokollierung aktivieren. Standardmäßig sind alle Protokollierungstypen deaktiviert.

    - Weitere Informationen zur Prometheus-Metrik-Option finden Sie unter the section called “Schritt 1: Prometheus-Metriken aktivieren”.
    - Weitere Informationen zu den Optionen für die Steuerebenen-Protokollierung finden Sie unter the section called “Protokolle der Steuerebene”.
    - Wenn Sie mit dieser Seite fertig sind, wählen Sie Weiter aus.

11. Wählen Sie auf der Seite Select add-ons (Add-Ons auswählen) die Add-Ons aus, die Sie Ihrem Cluster hinzufügen möchten. Sie können so viele Amazon EKS-Add-Ons und AWS Marketplace-Add-Ons auswählen, wie Sie benötigen. Wenn die AWS Marketplace-Add-Ons, die Sie installieren möchten, nicht aufgeführt sind, können Sie auf die Seitennummerierung klicken, um zusätzliche Seitenergebnisse anzuzeigen, oder nach verfügbaren AWS Marketplace-Add-Ons suchen, indem Sie Text in das Suchfeld eingeben. Sie können auch nach Kategorie, Anbieter oder Preismodell

filtern und dann die Add-Ons aus den Suchergebnissen auswählen. Beim Erstellen eines Clusters können Sie alle Add-Ons, die EKS Pod Identities unterstützen, wie in the section called “Pod Identity” beschrieben, anzeigen, auswählen und installieren.

- EKS Auto Mode automatisiert die Funktionalität bestimmter Add-Ons. Wenn Sie planen, EKS-verwaltete Knotengruppen in Ihrem EKS-Auto-Mode-Cluster bereitzustellen, wählen Sie Zusätzliche Amazon-EKS-Add-Ons aus und überprüfen Sie die Optionen. Möglicherweise müssen Sie Add-Ons wie CoreDNS und Kube-Proxy installieren. EKS installiert die Add-Ons in diesem Abschnitt nur in selbstverwalteten Knoten und Knotengruppen.
- Wenn Sie mit dieser Seite fertig sind, wählen Sie Weiter aus.

12. Wählen Sie auf der Seite Einstellungen für ausgewählte Add-ons konfigurieren die Version aus, die Sie installieren möchten. Sie können nach der Clustererstellung jederzeit auf eine neuere Version aktualisieren.

    Bei Add-Ons, die EKS Pod Identities unterstützen, können Sie die Konsole verwenden, um die Rolle automatisch zu generieren, wobei der Name, die AWS verwaltete Richtlinie und die Vertrauensrichtlinie speziell für das Add-on vorausgefüllt sind. Sie können vorhandene Rollen wiederverwenden oder neue Rollen für unterstützte Add-Ons erstellen. Schritte zur Verwendung der Konsole zum Erstellen von Rollen für Add-Ons, die EKS Pod Identities unterstützen, finden Sie unter the section called “Add-on erstellen (Konsole)AWS”. Wenn ein Add-On EKS Pod Identity nicht unterstützt, wird nach der Erstellung des Clusters eine Meldung mit Anweisungen zur Verwendung des Assistenten zum Erstellen der IAM-Rollen für Servicekonten (IRSA) angezeigt.

    Sie können die Konfiguration jedes Add-Ons nach der Cluster-Erstellung aktualisieren. Weitere Informationen zum Konfigurieren eines Add-Ons finden Sie unter the section called “Add-On aktualisieren”. Wenn Sie mit dieser Seite fertig sind, wählen Sie Weiter aus.

13. Überprüfen Sie auf der Seite Review and create (Überprüfen und erstellen) die Informationen, die Sie auf den vorherigen Seiten eingegeben oder ausgewählt haben. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Wenn Sie zufrieden sind, klicken Sie auf Erstellen. Das Feld Status zeigt CREATING (WIRD ERSTELLT) an, während der Cluster bereitgestellt wird.

> **Note**
>
> Sie erhalten möglicherweise eine Fehlermeldung, dass eine der Availability Zones in Ihrer Anfrage nicht über genügend Kapazität zum Erstellen eines Amazon-EKS-Clusters verfügt. Wenn dies der Fall ist, enthält die Fehlerausgabe die Availability Zones, die einen neuen Cluster unterstützen können. Versuchen Sie, Ihren Cluster mit mindestens zwei Subnetzen

erneut zu erstellen, die sich in den unterstützten Availability Zones für Ihr Konto befinden. Weitere Informationen finden Sie unter the section called “Unzureichende Kapazität”.

Die Clusterbereitstellung dauert mehrere Minuten.

Cluster erstellen — AWS CLI

Die folgenden CLI-Anweisungen behandeln die Erstellung von IAM-Ressourcen und das Erstellen des Clusters.

Erstellung einer IAM-Rolle für einen Cluster in EKS Auto Mode

Schritt 1: Vertrauensrichtlinie erstellen

Erstellen Sie eine Vertrauensrichtlinie, mit welcher der Amazon-EKS-Service die Rolle übernehmen kann. Speichern Sie die Richtlinie als `trust-policy.json`:

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
```

Schritt 2: Erstellen der IAM-Rolle

Verwenden Sie die Vertrauensrichtlinie, um die Cluster-IAM-Rolle zu erstellen:

```
aws iam create-role \
    --role-name AmazonEKSAutoClusterRole \
    --assume-role-policy-document file://trust-policy.json
```

Schritt 3: Rollen-ARN notieren

Rufen Sie die ARN der neuen Rolle ab und speichern Sie sie für die Verwendung in den folgenden Schritten:

```
aws iam get-role --role-name AmazonEKSAutoClusterRole --query "Role.Arn" --output text
```

Schritt 4: Erforderliche Richtlinien hinzufügen

Fügen Sie der Cluster-IAM-Rolle die folgenden AWS verwalteten Richtlinien hinzu, um die erforderlichen Berechtigungen zu gewähren:

EKSClusterAmazon-Richtlinie:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoClusterRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSClusterPolicy
```

EKSComputeAmazon-Richtlinie:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoClusterRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSComputePolicy
```

Amazon EKSBlock StoragePolicy:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoClusterRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSBlockStoragePolicy
```

Amazon EKSLoad BalancingPolicy:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoClusterRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSLoadBalancingPolicy
```

EKSNetworkingAmazon-Richtlinie:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoClusterRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSNetworkingPolicy
```

Erstellung einer Knoten-IAM-Rolle in EKS Auto Mode

Schritt 1: Vertrauensrichtlinie erstellen

Erstellen Sie eine Vertrauensrichtlinie, mit welcher der Amazon-EKS-Service die Rolle übernehmen kann. Speichern Sie die Richtlinie als `node-trust-policy.json`:

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Schritt 2: Knoten-IAM-Rolle erstellen

Verwenden Sie die node-trust-policyJSON-Datei aus dem vorherigen Schritt, um zu definieren, welche Entitäten die Rolle übernehmen können. Führen Sie den folgenden Befehl aus, um die Knoten-IAM-Rolle zu erstellen:

```
aws iam create-role \
    --role-name AmazonEKSAutoNodeRole \
    --assume-role-policy-document file://node-trust-policy.json
```

Schritt 3: Rollen-ARN notieren

Rufen Sie nach dem Erstellen der Rolle die ARN der Knoten-IAM-Rolle ab und speichern Sie sie. Sie benötigen diese ARN in den nachfolgenden Schritten. Um die ARN abzurufen, verwenden Sie den folgenden Befehl.

```
aws iam get-role --role-name AmazonEKSAutoNodeRole --query "Role.Arn" --output text
```

Schritt 4: Erforderliche Richtlinien hinzufügen

Fügen Sie der Node-IAM-Rolle die folgenden AWS verwalteten Richtlinien hinzu, um die erforderlichen Berechtigungen bereitzustellen:

Amazon EKSWorker NodeMinimalPolicy:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoNodeRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSWorkerNodeMinimalPolicy
```

Amazon EC2 ContainerRegistryPullOnly:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoNodeRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEC2ContainerRegistryPullOnly
```

Cluster erstellen

1. Erstellen Sie den Cluster mit dem folgenden Befehl. Nehmen Sie vor der Ausführung des Befehls die folgenden Ersetzungen vor:
   - *region-code*Ersetzen Sie es durch die AWS Region, in der Sie Ihren Cluster erstellen möchten.
   - Ersetzen Sie *my-cluster* durch Ihren Cluster-Namen. Der Name darf nur alphanumerische Zeichen (Groß-/Kleinschreibung beachten), Bindestriche und Unterstriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein.
   - Ersetzen Sie *1.30* durch eine von Amazon EKS unterstützte Version.
   - Ersetzen Sie ihn *111122223333* durch Ihre Konto-ID
   - Wenn Sie unterschiedlich benannte IAM-Rollen für die Cluster- und Node-Rollen erstellt haben, ersetzen Sie die ARNs.
   - Ersetzen Sie `subnetIds` durch Ihre eigenen Werte. Sie können auch weitere IDs hinzufügen. Sie müssen mindestens zwei Subnetze IDs angeben.

     Die von Ihnen gewählten Subnetze müssen die Anforderungen für Amazon-EKS-Subnetze erfüllen. Bevor Sie Subnetze auswählen, empfehlen wir, dass Sie sich mit allen Anforderungen und Überlegungen für Amazon-EKS-VPCs und -Subnetze vertraut machen.
   - Wenn Sie keine Sicherheitsgruppen-ID angeben möchten, entfernen Sie `,securityGroupIds=sg-<ExampleID1>` aus dem Befehl. Wenn Sie eine oder mehrere Sicherheitsgruppen angeben möchten IDs, ersetzen Sie die Werte für `securityGroupIds` durch Ihre eigenen. Sie können auch weitere hinzufügen IDs.

Unabhängig davon, ob Sie Sicherheitsgruppen wählen oder nicht, erstellt Amazon EKS eine Sicherheitsgruppe, die die Kommunikation zwischen Ihrem Cluster und Ihrer VPC ermöglicht. Amazon EKS verknüpft diese Sicherheitsgruppe und alle, die Sie wählen, mit den erstellten Netzwerkschnittstellen. Weitere Informationen zu der Cluster-Sicherheitsgruppe, die Amazon EKS erstellt, finden Sie unter the section called “Anforderungen an Sicherheitsgruppen”. Sie können die Regeln in der von Amazon EKS erstellten Cluster-Sicherheitsgruppe ändern.

```
aws eks create-cluster \
  --region region-code \
  --name my-cluster \
  --kubernetes-version 1.30 \
  --role-arn arn:aws: iam::111122223333:role/AmazonEKSAutoClusterRole \
  --resources-vpc-config '{"subnetIds": ["subnet-ExampleID1","subnet-ExampleID2"],
 "securityGroupIds": ["sg-ExampleID1"], "endpointPublicAccess": true,
 "endpointPrivateAccess": true}' \
  --compute-config '{"enabled": true, "nodeRoleArn": "arn:aws:
 iam::111122223333:role/AmazonEKSAutoNodeRole", "nodePools": ["general-purpose",
 "system"]}' \
  --kubernetes-network-config '{"elasticLoadBalancing": {"enabled": true}}' \
  --storage-config '{"blockStorage": {"enabled": true}}' \
  --access-config '{"authenticationMode": "API"}'
```

> **Note**
>
> Sie erhalten möglicherweise eine Fehlermeldung, dass eine der Availability Zones in Ihrer Anfrage nicht über genügend Kapazität zum Erstellen eines Amazon-EKS-Clusters verfügt. Wenn dies der Fall ist, enthält die Fehlerausgabe die Availability Zones, die einen neuen Cluster unterstützen können. Versuchen Sie, Ihren Cluster mit mindestens zwei Subnetzen erneut zu erstellen, die sich in den unterstützten Availability Zones für Ihr Konto befinden. Weitere Informationen finden Sie unter the section called “Unzureichende Kapazität”.

Im Folgenden finden Sie optionale Einstellungen, die bei Bedarf dem vorherigen Befehl hinzugefügt werden müssen. Sie können diese Optionen nur aktivieren, wenn Sie den Cluster erstellen, nicht danach.

- Wenn Sie angeben möchten welchem IPv4 CIDR-Block (Classless Inter-Domain Routing) Kubernetes Service-IP-Adressen zuweist, geben Sie die Option `--kubernetes-network-config serviceIpv4Cidr=<cidr-block>` für den folgenden Befehl an.

  Durch die Angabe Ihres eigenen Bereichs können Sie Konflikte zwischen Kubernetes-Services und anderen Netzwerken vermeiden, die mit Ihrer VPC verbunden sind. Geben Sie einen Bereich in CIDR-Notation ein. Beispiel: `10.2.0.0/16`.

  Der CIDR-Block muss die folgenden Anforderungen erfüllen:

  - Verwendet einen der folgenden Bereiche: `10.0.0.0/8`, `172.16.0.0/12`, oder `192.168.0.0/16`.
  - Hat eine Mindestgröße von `/24` und eine maximale Größe von `/12`.
  - Überschneidet sich nicht mit dem Bereich der VPC für Ihre Amazon-EKS-Ressourcen.

    Sie können diese Option nur angeben, wenn Sie die IPv4-Adressfamilie verwenden, und nur bei Erstellen des Clusters. Wenn Sie dies nicht angeben, weist Kubernetes Service-IP-Adressen aus den CIDR-Blöcken `10.100.0.0/16` oder `172.20.0.0/16` zu.

- Wenn Sie einen Cluster erstellen und möchten, dass der Cluster Pods und Services IPv6-Adressen statt IPv4-Adressen zuweist, fügen Sie dem folgenden Befehl `--kubernetes-network-config ipFamily=ipv6` hinzu.

  Kubernetes weist Pods und Services standardmäßig IPv4-Adressen zu. Bevor Sie sich entscheiden, die IPv6-Familie zu verwenden, stellen Sie sicher, dass Sie mit allen Überlegungen und Anforderungen in den VPC-Anforderungen und -Überlegungen, the section called “Subnetz-Anforderungen und -Überlegungen”-, the section called “Anforderungen an Sicherheitsgruppen”- und the section called “IPv6”-Themen vertraut sind. Wenn Sie die IPv6-Familie auswählen, können Sie keinen Adressbereich für Kubernetes festlegen, aus dem IPv6-Serviceadressen zugewiesen werden sollen, wie dies bei der IPv4-Familie möglich ist. Kubernetes weist Serviceadressen aus dem eindeutigen lokalen Adressbereich (`fc00::/7`) zu.

2. Die Bereitstellung des Clusters dauert mehrere Minuten. Sie können den Status Ihres Clusters mit dem folgenden Befehl überprüfen.

```
aws eks describe-cluster --region region-code --name my-cluster --query
 "cluster.status"
```

Nächste Schritte

- the section called “Zugriff auf den Cluster mit kubectl”
- the section called “Zugriffseinträge”
- Aktivieren Sie die Secrets-Verschlüsselung für Ihren Cluster.
- Konfigurieren Sie die Protokollierung für Ihren Cluster.
- Fügen Sie Ihrem Cluster Knoten hinzu.

Amazon-EKS-Cluster erstellen

> **Note**
>
> Dieses Thema behandelt die Erstellung von Amazon EKS-Clustern ohne EKS Auto Mode. Detaillierte Anweisungen zum Erstellen eines EKS-Auto-Mode-Clusters finden Sie unter the section called “Automatik-Cluster erstellen”.
> Informationen zum Einstieg in EKS Auto Mode finden Sie unter the section called “Cluster erstellen (EKS Auto Mode)”.

Dieses Thema bietet einen Überblick über die verfügbaren Optionen und beschreibt, was zu beachten ist, wenn Sie einen Amazon-EKS-Cluster erstellen. Informationen zum Erstellen eines Clusters mit Ihrer On-Premises-Infrastruktur als Rechenleistung für Knoten finden Sie unter the section called “Cluster erstellen”. Wenn Sie zum ersten Mal einen Amazon-EKS-Cluster erstellen, empfehlen wir, entsprechend unseren Leitfäden in Erste Schritte vorzugehen. Diese Leitfäden helfen, einen einfachen Standard-Cluster zu erstellen, ohne auf alle verfügbaren Optionen einzugehen.

Voraussetzungen

- Eine vorhandene VPC und Subnetze, die die Amazon-EKS-Anforderungen erfüllen. Bevor Sie einen Cluster für den Produktionseinsatz bereitstellen, empfehlen wir, dass Sie sich ein umfassendes Verständnis der VPC- und Subnetzanforderungen aneignen. Wenn Sie keine VPC und keine Subnetze haben, können Sie diese mithilfe einer von Amazon EKS bereitgestellten AWS CloudFormation Vorlage erstellen.
- Das `kubectl`-Befehlszeilen-Tool ist auf Ihrem Gerät oder in der AWS CloudShell installiert. Die Version kann mit der Kubernetes-Version Ihres Clusters identisch sein oder bis zu einer

Nebenversion älter oder neuer sein. Informationen zum Installieren oder Aktualisieren von `kubectl` finden Sie unter the section called “kubectl und eksctl einrichten”.

- Version `2.12.3` oder höher oder Version `1.27.160` oder höher der auf Ihrem Gerät installierten und konfigurierten AWS Befehlszeilenschnittstelle (AWS CLI) oder AWS CloudShell. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Paketmanager wie `yum` `apt-get`, oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der AWS CLI. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Die AWS CLI-Version, in der installiert ist, AWS CloudShell kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zur Aktualisierung finden Sie im AWS CloudShell Benutzerhandbuch unter AWS CLI in Ihrem Home-Verzeichnis installieren.
- Ein IAM-Prinzipal mit Berechtigungen für das `create` und `describe` eines Amazon-EKS-Clusters. Weitere Informationen erhalten Sie unter the section called “Kubernetes-Cluster auf einem Outpost erstellen” und the section called “Auflisten oder Beschreiben aller Cluster”.

Schritt 1: Cluster-IAM-Rolle erstellen

1. Wenn Sie bereits eine Cluster-IAM-Rolle haben oder Ihren Cluster mit `eksctl` erstellen, können Sie diesen Schritt überspringen. Standardmäßig erstellt `eksctl` eine Rolle für Sie.
2. Führen Sie den folgenden Befehl aus, um eine JSON-Datei für eine IAM-Vertrauensrichtlinie zu erstellen.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

3. Erstellen Sie die Amazon-EKS-Cluster-IAM-Rolle. Falls erforderlich, stellen Sie *`eks-cluster-role-trust-policy.json`* den Pfad auf Ihrem Computer voran, in den Sie im vorherigen

Schritt die Datei geschrieben haben. Der Befehl verknüpft die im vorherigen Schritt erstellte Vertrauensrichtlinie mit der Rolle. Um eine IAM-Rolle zu erstellen, muss dem IAM-Prinzipal, der die Rolle erstellt, die `iam:CreateRole`-Aktion (Berechtigung) zugewiesen werden.

```
aws iam create-role --role-name myAmazonEKSClusterRole --assume-role-policy-document
 file://"eks-cluster-role-trust-policy.json"
```

4. Sie können entweder die von Amazon EKS verwaltete Richtlinie zuweisen oder Ihre eigene benutzerdefinierte Richtlinie erstellen. Informationen zu den Mindestberechtigungen, die Sie in Ihrer benutzerdefinierten Richtlinie verwenden müssen, finden Sie unter the section called “Cluster-IAM-Rolle”.

   Fügen Sie der Rolle die von Amazon EKS verwaltete Richtlinie mit dem Namen Amazon EKSCluster Policy hinzu. Um eine IAM-Richtlinie an einen IAM-Prinzipal anzuhängen, muss der Prinzipal, der die Richtlinie anhängt, eine der folgenden IAM-Aktionen (Berechtigungen) zugewiesen werden: `iam:AttachUserPolicy` oder `iam:AttachRolePolicy`.

   ```
   aws iam attach-role-policy --policy-arn arn:aws: iam::aws:policy/
   AmazonEKSClusterPolicy --role-name myAmazonEKSClusterRole
   ```

Serviceverknüpfte Rolle

Amazon EKS erstellt automatisch eine mit dem Service verknüpfte Rolle mit dem Namen `AWSServiceRoleForAmazonEKS`.

Diese ergänzt die Cluster-IAM-Rolle. Eine servicegebundene Rolle ist eine einzigartige Art von IAM-Rolle, die direkt mit Amazon EKS verknüpft ist. Diese Rolle ermöglicht es Amazon EKS, Cluster in Ihrem Konto zu verwalten. Weitere Informationen finden Sie unter the section called “Cluster--Rolle”.

Die IAM-Identität, die Sie zum Erstellen des EKS-Clusters verwenden, muss über die Berechtigung zum Erstellen der serviceverknüpften Rolle verfügen. Dies beinhaltet die `iam:CreateServiceLinkedRole`-Berechtigung.

Wenn die serviceverknüpfte Rolle noch nicht vorhanden ist und Ihre aktuelle IAM-Rolle nicht über ausreichende Berechtigungen zum Erstellen verfügt, schlägt der Cluster-Erstellungsvorgang fehl.

Schritt 2: Cluster erstellen

Sie können einen Cluster erstellen, indem Sie Folgendes verwenden:

- `eksctl`
- die AWS-Managementkonsole
- die AWS CLI

Cluster erstellen – eksctl

1. Sie benötigen eine Version `0.215.0` oder eine neuere Version des `eksctl` Befehlszeilentools, das auf Ihrem Gerät installiert ist oder AWS CloudShell. Informationen zum Installieren und Aktualisieren von `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.
2. Erstellen Sie einen Amazon `IPv4` EKS-Cluster mit der Amazon EKS-Standardversion von Kubernetes in Ihrer AWS Standardregion. Nehmen Sie vor der Ausführung des Befehls die folgenden Ersetzungen vor:
3. *`region-code`*Ersetzen Sie es durch die AWS Region, in der Sie Ihren Cluster erstellen möchten.
4. Ersetzen Sie *`my-cluster`* durch Ihren Cluster-Namen. Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein.
5. Ersetzen Sie *`1.33`* durch eine von Amazon EKS unterstützte Version.
6. Ändern Sie die Werte für `vpc-private-subnets`, um Ihre Anforderungen zu erfüllen. Sie können auch weitere hinzufügen IDs. Sie müssen mindestens zwei Subnetze IDs angeben. Wenn Sie lieber öffentliche Subnetze angeben möchten, können Sie `--vpc-private-subnets` zu `--vpc-public-subnets` ändern. Öffentliche Subnetze verfügen über eine zugeordnete Routing-Tabelle mit einer Route zu einem Internet-Gateway, private Subnetze verfügen nicht über eine zugeordnete Routing-Tabelle. Wir empfehlen, nach Möglichkeit private Subnetze zu verwenden.

   Die von Ihnen gewählten Subnetze müssen die Anforderungen für Amazon-EKS-Subnetze erfüllen. Bevor Sie Subnetze auswählen, empfehlen wir, dass Sie sich mit allen Anforderungen und Überlegungen für Amazon-EKS-VPCs und -Subnetze vertraut machen.
7. Führen Sie den folgenden Befehl aus:

```
eksctl create cluster --name my-cluster --region region-code --version 1.33 --vpc-
private-subnets subnet-ExampleID1,subnet-ExampleID2 --without-nodegroup
```

Die Clusterbereitstellung dauert mehrere Minuten. Während der Cluster erstellt wird, werden mehrere Ausgabezeilen angezeigt. Die letzte Ausgabezeile ähnelt der folgenden Beispielzeile.

```
[#]  EKS cluster "my-cluster" in "region-code" region is ready
```

8. Mit the section called “Schritt 3: kubeconfig aktualisieren” fortfahren

Optionale Einstellungen

Sie können die meisten Optionen, die beim Erstellen eines Clusters mit `eksctl` angegeben werden können, über den Befehl `eksctl create cluster --help` anzeigen. Verwenden Sie eine `config`-Datei, um alle verfügbaren Optionen anzuzeigen. Weitere Informationen finden Sie unter Verwenden von Config-Dateien und im Config-Datei-Schema in der `eksctl`-Dokumentation. Auf GitHub finden Sie Beispiele für Config-Dateien.

Im Folgenden finden Sie optionale Einstellungen, die bei Bedarf dem vorherigen Befehl hinzugefügt werden müssen. Sie können diese Optionen nur aktivieren, wenn Sie den Cluster erstellen, nicht danach. Wenn Sie diese Optionen angeben müssen, müssen Sie den Cluster mit einer eksctl-Konfigurationsdatei erstellen und die Einstellungen angeben, anstatt den vorherigen Befehl zu verwenden.

- Wenn Sie eine oder mehrere Sicherheitsgruppen angeben möchten, die Amazon EKS den erstellten Netzwerkschnittstellen zuweist, geben Sie die Option securityGroup an.

  Unabhängig davon, ob Sie Sicherheitsgruppen wählen oder nicht, erstellt Amazon EKS eine Sicherheitsgruppe, die die Kommunikation zwischen Ihrem Cluster und Ihrer VPC ermöglicht. Amazon EKS verknüpft diese Sicherheitsgruppe und alle, die Sie wählen, mit den erstellten Netzwerkschnittstellen. Weitere Informationen zu der Cluster-Sicherheitsgruppe, die Amazon EKS erstellt, finden Sie unter the section called “Anforderungen an Sicherheitsgruppen”. Sie können die Regeln in der von Amazon EKS erstellten Cluster-Sicherheitsgruppe ändern.
- Wenn Sie angeben möchten, welchem CIDR-Block (`IPv4`Classless Inter-Domain Routing) Kubernetes Dienst-IP-Adressen zuweist, geben Sie die Service-CIDR-Option an. IPv4

  Durch die Angabe Ihres eigenen Bereichs können Sie Konflikte zwischen Kubernetes-Services und anderen Netzwerken vermeiden, die mit Ihrer VPC verbunden sind. Geben Sie einen Bereich in CIDR-Notation ein. Beispiel: `10.2.0.0/16`.

  Der CIDR-Block muss die folgenden Anforderungen erfüllen:

- Verwendet einen der folgenden Bereiche: `10.0.0.0/8`, `172.16.0.0/12`, oder `192.168.0.0/16`.
- Hat eine Mindestgröße von `/24` und eine maximale Größe von `/12`.
- Überschneidet sich nicht mit dem Bereich der VPC für Ihre Amazon-EKS-Ressourcen.

  Sie können diese Option nur angeben, wenn Sie die `IPv4`-Adressfamilie verwenden, und nur bei Erstellen des Clusters. Wenn Sie dies nicht angeben, weist Kubernetes Service-IP-Adressen aus den CIDR-Blöcken `10.100.0.0/16` oder `172.20.0.0/16` zu.

- Wenn Sie einen Cluster erstellen und möchten, dass der Cluster Pods und Services `IPv6`-Adressen statt `IPv4`-Adressen zuweist, geben Sie die Option ipFamily an.

  Kubernetes weist Pods und Services standardmäßig `IPv4`-Adressen zu. Bevor Sie sich entscheiden, die `IPv6`-Familie zu verwenden, stellen Sie sicher, dass Sie mit allen Überlegungen und Anforderungen in den Themen the section called “VPC-Anforderungen und -Überlegungen”, the section called “Subnetz-Anforderungen und -Überlegungen”, the section called “Anforderungen an Sicherheitsgruppen”, und the section called “IPv6” vertraut sind. Wenn Sie die `IPv6`-Familie auswählen, können Sie keinen Adressbereich für Kubernetes festlegen, aus dem `IPv6`-Serviceadressen zugewiesen werden sollen, wie dies bei der `IPv4`-Familie möglich ist. Kubernetes weist Serviceadressen aus dem eindeutigen lokalen Adressbereich (`fc00::/7`) zu.

Cluster erstellen — Konsole AWS

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie Cluster hinzufügen und dann Erstellen aus.
3. Wählen Sie unter Konfigurationsoptionen die Option Benutzerdefinierte Konfiguration
   - Informationen zur schnellen Erstellung eines Clusters mit EKS Auto Mode finden Sie unter the section called “Management-Konsole”.
4. Deaktivieren Sie unter EKS Auto Mode die Option EKS Auto Mode verwenden.
   - Informationen zum Erstellen eines EKS-Auto-Mode-Clusters mit benutzerdefinierter Konfiguration finden Sie unter the section called “Automatik-Cluster erstellen”.
5. Füllen Sie auf der Seite Configure cluster (Cluster konfigurieren) die folgenden Felder aus:
   - Name – Ein Name für Ihren Cluster. Der Name darf nur alphanumerische Zeichen (Groß-/Kleinschreibung beachten), Bindestriche und Unterstriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name

muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein.

- Cluster-IAM-Rolle — Wählen Sie die Amazon EKS-Cluster-IAM-Rolle aus, die Sie erstellt haben, damit die Kubernetes-Steuerebene AWS Ressourcen in Ihrem Namen verwalten kann.
- Kubernetes-Version – Die Kubernetes-Version, die für den Cluster verwendet wird. Wir empfehlen, die frühere Version auszuwählen, es sei denn, Sie benötigen eine ältere Version.
- Supporttyp – Die Kubernetes-Versionsrichtlinie, die Sie für Ihren Cluster festlegen möchten. Wenn Ihr Cluster nur mit einer Standard-Supportversion ausgeführt werden soll, können Sie Standard-Support auswählen. Wenn Sie möchten, dass Ihr Cluster nach Ablauf des Standard-Supports für eine Version in den erweiterten Support übergeht, können Sie Erweiterter Support auswählen. Wenn Sie eine Kubernetes-Version auswählen, für die derzeit der erweiterte Support gilt, können Sie den Standard-Support nicht als Option auswählen.
- Secrets-Verschlüsselung – (Optional) Aktivieren Sie die Secrets-Verschlüsselung von Kubernetes-Secrets mithilfe eines KMS-Schlüssels. Sie können diese auch aktivieren, nachdem Sie Ihren Cluster erstellt haben. Stellen Sie vor Aktivierung dieser Funktion sicher, dass Sie mit den Informationen unter the section called “Geheime Verschlüsselung aktivieren” vertraut sind.
- Tags – (Optional) Fügen Sie Ihrem Cluster beliebige Tags hinzu. Weitere Informationen finden Sie unter the section called “Markieren Ihrer -Ressourcen”.
- ARC-Zonenverschiebung – (Optional) Sie können den Route53 Application Recovery Controller einsetzen, um beeinträchtigte Availability Zones zu mindern. Weitere Informationen finden Sie unter the section called “Weitere Informationen zur Zonenverschiebung”.

6. Geben Sie im Abschnitt Cluster-Zugriff der Seite „Cluster konfigurieren“ die folgenden Felder ein:
   - Administrator-Zugriff für Bootstrap-Cluster – Der Cluster-Ersteller ist automatisch ein Kubernetes-Administrator. Wenn Sie dies deaktivieren möchten, wählen Sie Cluster-Administratorzugriff verbieten aus.
   - Cluster-Authentifizierungsmodus — Legen Sie fest, wie Sie IAM-Benutzern und -Rollen Zugriff auf Kubernetes gewähren möchten. APIs Weitere Informationen finden Sie unter the section called “Einstellen des Cluster-Authentifizierungsmodus”.

   Wenn Sie mit dieser Seite fertig sind, wählen Sie Weiter aus.
7. Wählen Sie auf der Seite Specify networking (Netzwerk angeben) die Werte für die folgenden Felder aus:
   - VPC – Wählen Sie eine vorhandene VPC aus, die die Anforderungen für Amazon EKS VPC erfüllt, um Ihren Cluster zu erstellen. Bevor Sie sich für eine VPC entscheiden, empfehlen wir, sich mit allen Anforderungen und Überlegungen in the section called “VPC- und Subnetz-

Anforderungen" vertraut zu machen. Sie können nach der Cluster-Erstellung nicht mehr ändern, welche VPC Sie verwenden möchten. Wenn keine aufgeführt VPCs sind, müssen Sie zuerst eine erstellen. Weitere Informationen finden Sie unter the section called "Erstellen einer VPC".

- Subnetze – Standardmäßig sind alle im vorherigen Feld angegebenen Subnetze in der VPC vorausgewählt. Sie müssen mindestens zwei auswählen.

  Die von Ihnen gewählten Subnetze müssen die Anforderungen für Amazon-EKS-Subnetze erfüllen. Bevor Sie Subnetze auswählen, empfehlen wir, dass Sie sich mit allen Anforderungen und Überlegungen für Amazon-EKS-VPCs und -Subnetze vertraut machen.

  Security groups (Sicherheitsgruppen) – (Optional) Geben Sie eine oder mehrere Sicherheitsgruppen an, die Amazon EKS den erstellten Netzwerkschnittstellen zuordnen soll.

  Unabhängig davon, ob Sie Sicherheitsgruppen wählen oder nicht, erstellt Amazon EKS eine Sicherheitsgruppe, die die Kommunikation zwischen Ihrem Cluster und Ihrer VPC ermöglicht. Amazon EKS verknüpft diese Sicherheitsgruppe und alle, die Sie wählen, mit den erstellten Netzwerkschnittstellen. Weitere Informationen zu der Cluster-Sicherheitsgruppe, die Amazon EKS erstellt, finden Sie unter the section called "Anforderungen an Sicherheitsgruppen". Sie können die Regeln in der von Amazon EKS erstellten Cluster-Sicherheitsgruppe ändern.
- Cluster-IP-Adressfamilie wählen — Sie können entweder IPv4und wählen IPv6.

  Kubernetes weist Pods und Services standardmäßig `IPv4`-Adressen zu. Bevor Sie sich entscheiden, die `IPv6`-Familie zu verwenden, stellen Sie sicher, dass Sie mit allen Überlegungen und Anforderungen in den VPC-Anforderungen und -Überlegungen, the section called "Subnetz-Anforderungen und -Überlegungen"-, the section called "Anforderungen an Sicherheitsgruppen"- und the section called "IPv6"-Themen vertraut sind. Wenn Sie die `IPv6`-Familie auswählen, können Sie keinen Adressbereich für Kubernetes festlegen, aus dem `IPv6`-Serviceadressen zugewiesen werden sollen, wie dies bei der `IPv4`-Familie möglich ist. Kubernetes weist Serviceadressen aus dem eindeutigen lokalen Adressbereich (`fc00::/7`) zu.
- (Optional) Wählen Sie Configure Kubernetes Service IP address range (IP-Adressbereich des Kubernetes-Diensts konfigurieren) und geben Sie einen **`IPv4`**-Bereich für den Service an.

  Durch die Angabe Ihres eigenen Bereichs können Sie Konflikte zwischen Kubernetes-Services und anderen Netzwerken vermeiden, die mit Ihrer VPC verbunden sind. Geben Sie einen Bereich in CIDR-Notation ein. Beispiel: `10.2.0.0/16`.

  Der CIDR-Block muss die folgenden Anforderungen erfüllen:

- Verwendet einen der folgenden Bereiche: `10.0.0.0/8`, `172.16.0.0/12`, oder `192.168.0.0/16`.
- Hat eine Mindestgröße von `/24` und eine maximale Größe von `/12`.
- Überschneidet sich nicht mit dem Bereich der VPC für Ihre Amazon-EKS-Ressourcen.

Sie können diese Option nur angeben, wenn Sie die `IPv4`-Adressfamilie verwenden, und nur bei Erstellen des Clusters. Wenn Sie dies nicht angeben, weist Kubernetes Service-IP-Adressen aus den CIDR-Blöcken `10.100.0.0/16` oder `172.20.0.0/16` zu.

- Wählen Sie eine Option für den Cluster-Endpunktzugriff aus. Nachdem Ihr Cluster erstellt wurde, können Sie diese Option ändern. Bevor Sie eine nicht standardmäßige Option auswählen, sollten Sie sich mit den Optionen und deren Auswirkungen vertraut machen. Weitere Informationen finden Sie unter the section called “Cluster-Endpunktzugriff”.

  Wenn Sie mit dieser Seite fertig sind, wählen Sie Weiter aus.

8. (Optional) Auf der Seite Beobachtbarkeit konfigurieren können Sie Metriken und Optionen zur Steuerebenen-Protokollierung aktivieren. Standardmäßig sind alle Protokollierungstypen deaktiviert.
   - Weitere Informationen zur Prometheus-Metrik-Option finden Sie unter the section called “Schritt 1: Prometheus-Metriken aktivieren”.
   - Weitere Informationen zu den Optionen für die Steuerebenen-Protokollierung finden Sie unter the section called “Protokolle der Steuerebene”.

   Wenn Sie mit dieser Seite fertig sind, wählen Sie Weiter aus.
9. Wählen Sie auf der Seite Select add-ons (Add-Ons auswählen) die Add-Ons aus, die Sie Ihrem Cluster hinzufügen möchten. Bestimmte Add-Ons sind bereits vorausgewählt. Sie können so viele Amazon EKS-Add-Ons und AWS Marketplace-Add-Ons auswählen, wie Sie benötigen. Wenn die AWS Marketplace-Add-Ons, die Sie installieren möchten, nicht aufgeführt sind, können Sie auf die Seitennummerierung klicken, um zusätzliche Seitenergebnisse anzuzeigen, oder nach verfügbaren AWS Marketplace-Add-Ons suchen, indem Sie Text in das Suchfeld eingeben. Sie können auch nach Kategorie, Anbieter oder Preismodell filtern und dann die Add-Ons aus den Suchergebnissen auswählen. Beim Erstellen eines Clusters können Sie alle Add-Ons, die EKS Pod Identities unterstützen, wie in the section called “Pod Identity” beschrieben, anzeigen, auswählen und installieren.

   Wenn Sie mit dieser Seite fertig sind, wählen Sie Weiter aus.

Einige Add-Ons, wie Amazon VPC CNI, CoreDNS und kube-proxy, sind standardmäßig installiert. Wenn Sie eines der Standard-Add-Ons deaktivieren, kann dies Ihre Fähigkeit beeinträchtigen, Kubernetes-Anwendungen auszuführen.

10. Wählen Sie auf der Seite Einstellungen für ausgewählte Add-ons konfigurieren die Version aus, die Sie installieren möchten. Sie können nach der Clustererstellung jederzeit auf eine neuere Version aktualisieren.

    Bei Add-Ons, die EKS Pod Identities unterstützen, können Sie die Konsole verwenden, um die Rolle automatisch zu generieren, wobei der Name, die AWS verwaltete Richtlinie und die Vertrauensrichtlinie speziell für das Add-on vorausgefüllt sind. Sie können vorhandene Rollen wiederverwenden oder neue Rollen für unterstützte Add-Ons erstellen. Schritte zur Verwendung der Konsole zum Erstellen von Rollen für Add-Ons, die EKS Pod Identities unterstützen, finden Sie unter the section called “Add-on erstellen (Konsole)AWS”. Wenn ein Add-On EKS Pod Identity nicht unterstützt, wird nach der Erstellung des Clusters eine Meldung mit Anweisungen zur Verwendung des Assistenten zum Erstellen der IAM-Rollen für Servicekonten (IRSA) angezeigt.

    Sie können die Konfiguration jedes Add-Ons nach der Cluster-Erstellung aktualisieren. Weitere Informationen zum Konfigurieren eines Add-Ons finden Sie unter the section called “Add-On aktualisieren”. Wenn Sie mit dieser Seite fertig sind, wählen Sie Weiter aus.

11. Überprüfen Sie auf der Seite Review and create (Überprüfen und erstellen) die Informationen, die Sie auf den vorherigen Seiten eingegeben oder ausgewählt haben. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Wenn Sie zufrieden sind, klicken Sie auf Erstellen. Das Feld Status zeigt CREATING (WIRD ERSTELLT) an, während der Cluster bereitgestellt wird.

    > **Note**
    >
    > Sie erhalten möglicherweise eine Fehlermeldung, dass eine der Availability Zones in Ihrer Anfrage nicht über genügend Kapazität zum Erstellen eines Amazon-EKS-Clusters verfügt. Wenn dies der Fall ist, enthält die Fehlerausgabe die Availability Zones, die einen neuen Cluster unterstützen können. Versuchen Sie, Ihren Cluster mit mindestens zwei Subnetzen erneut zu erstellen, die sich in den unterstützten Availability Zones für Ihr Konto befinden. Weitere Informationen finden Sie unter the section called “Unzureichende Kapazität”.

    Die Clusterbereitstellung dauert mehrere Minuten.

12Mit the section called “Schritt 3: kubeconfig aktualisieren” fortfahren

Cluster erstellen — AWS CLI

1. Erstellen Sie den Cluster mit dem folgenden Befehl. Nehmen Sie vor der Ausführung des Befehls die folgenden Ersetzungen vor:
   - *region-code*Ersetzen Sie es durch die AWS Region, in der Sie Ihren Cluster erstellen möchten.
   - Ersetzen Sie *my-cluster* durch Ihren Cluster-Namen. Der Name darf nur alphanumerische Zeichen (Groß-/Kleinschreibung beachten), Bindestriche und Unterstriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein.
   - Ersetzen Sie *1.33* durch eine von Amazon EKS unterstützte Version.
   - Ersetzen Sie *111122223333* durch Ihre Konto-ID und *myAmazonEKSClusterRole* durch den Namen Ihres IAM-Rollen-Clusters.
   - Ersetzen Sie `subnetIds` durch Ihre eigenen Werte. Sie können auch weitere hinzufügen IDs. Sie müssen mindestens zwei Subnetze IDs angeben.

     Die von Ihnen gewählten Subnetze müssen die Anforderungen für Amazon-EKS-Subnetze erfüllen. Bevor Sie Subnetze auswählen, empfehlen wir, dass Sie sich mit allen Anforderungen und Überlegungen für Amazon-EKS-VPCs und -Subnetze vertraut machen.
   - Wenn Sie keine Sicherheitsgruppen-ID angeben möchten, entfernen Sie `,securityGroupIds=sg-<ExampleID1>` aus dem Befehl. Wenn Sie eine oder mehrere Sicherheitsgruppen angeben möchten IDs, ersetzen Sie die Werte für `securityGroupIds` durch Ihre eigenen. Sie können auch weitere hinzufügen IDs.

     Unabhängig davon, ob Sie Sicherheitsgruppen wählen oder nicht, erstellt Amazon EKS eine Sicherheitsgruppe, die die Kommunikation zwischen Ihrem Cluster und Ihrer VPC ermöglicht. Amazon EKS verknüpft diese Sicherheitsgruppe und alle, die Sie wählen, mit den erstellten Netzwerkschnittstellen. Weitere Informationen zu der Cluster-Sicherheitsgruppe, die Amazon EKS erstellt, finden Sie unter the section called “Anforderungen an Sicherheitsgruppen”. Sie können die Regeln in der von Amazon EKS erstellten Cluster-Sicherheitsgruppe ändern.

```
aws eks create-cluster --region region-code --name my-cluster --kubernetes-version
 1.33 \
```

```
   --role-arn arn:aws: iam::111122223333:role/myAmazonEKSClusterRole \
   --resources-vpc-config subnetIds=subnet-ExampleID1,subnet-
ExampleID2,securityGroupIds=sg-ExampleID1
```

> **Note**
>
> Sie erhalten möglicherweise eine Fehlermeldung, dass eine der Availability Zones in Ihrer Anfrage nicht über genügend Kapazität zum Erstellen eines Amazon-EKS-Clusters verfügt. Wenn dies der Fall ist, enthält die Fehlerausgabe die Availability Zones, die einen neuen Cluster unterstützen können. Versuchen Sie, Ihren Cluster mit mindestens zwei Subnetzen erneut zu erstellen, die sich in den unterstützten Availability Zones für Ihr Konto befinden. Weitere Informationen finden Sie unter the section called “Unzureichende Kapazität”.

Im Folgenden finden Sie optionale Einstellungen, die bei Bedarf dem vorherigen Befehl hinzugefügt werden müssen. Sie können diese Optionen nur aktivieren, wenn Sie den Cluster erstellen, nicht danach.

- Standardmäßig installiert EKS während der Cluster-Erstellung mehrere Netzwerk-Add-Ons. Dazu gehören Amazon VPC CNI, CoreDNS und kube-proxy.

  Wenn Sie die Installation dieser Standard-Netzwerk-Add-Ons deaktivieren möchten, verwenden Sie den folgenden Parameter. Dies kann als Alternative verwendet werden CNIs, z. B. für Cilium. Weitere Informationen finden Sie in der EKS-API-Referenz.

  ```
  aws eks create-cluster --bootstrapSelfManagedAddons false
  ```

- Wenn Sie angeben möchten welchem `IPv4` CIDR-Block (Classless Inter-Domain Routing) Kubernetes Service-IP-Adressen zuweist, geben Sie die Option `--kubernetes-network-config serviceIpv4Cidr=<cidr-block>` für den folgenden Befehl an.

  Durch die Angabe Ihres eigenen Bereichs können Sie Konflikte zwischen Kubernetes-Services und anderen Netzwerken vermeiden, die mit Ihrer VPC verbunden sind. Geben Sie einen Bereich in CIDR-Notation ein. Beispiel: `10.2.0.0/16`.

  Der CIDR-Block muss die folgenden Anforderungen erfüllen:

  - Verwendet einen der folgenden Bereiche: `10.0.0.0/8`, `172.16.0.0/12`, oder `192.168.0.0/16`.

- Hat eine Mindestgröße von `/24` und eine maximale Größe von `/12`.
- Überschneidet sich nicht mit dem Bereich der VPC für Ihre Amazon-EKS-Ressourcen.

Sie können diese Option nur angeben, wenn Sie die `IPv4`-Adressfamilie verwenden, und nur bei Erstellen des Clusters. Wenn Sie dies nicht angeben, weist Kubernetes Service-IP-Adressen aus den CIDR-Blöcken `10.100.0.0/16` oder `172.20.0.0/16` zu.

- Wenn Sie einen Cluster erstellen und möchten, dass der Cluster Pods und Services `IPv6`-Adressen statt `IPv4`-Adressen zuweist, fügen Sie dem folgenden Befehl `--kubernetes-network-config ipFamily=ipv6` hinzu.

  Kubernetes weist Pods und Services standardmäßig `IPv4`-Adressen zu. Bevor Sie sich entscheiden, die `IPv6`-Familie zu verwenden, stellen Sie sicher, dass Sie mit allen Überlegungen und Anforderungen in den VPC-Anforderungen und -Überlegungen, the section called “Subnetz-Anforderungen und -Überlegungen”-, the section called “Anforderungen an Sicherheitsgruppen”- und the section called “IPv6”-Themen vertraut sind. Wenn Sie die `IPv6`-Familie auswählen, können Sie keinen Adressbereich für Kubernetes festlegen, aus dem `IPv6`-Serviceadressen zugewiesen werden sollen, wie dies bei der `IPv4`-Familie möglich ist. Kubernetes weist Serviceadressen aus dem eindeutigen lokalen Adressbereich (`fc00::/7`) zu.

2. Die Bereitstellung des Clusters dauert mehrere Minuten. Sie können den Status Ihres Clusters mit dem folgenden Befehl überprüfen.

```
aws eks describe-cluster --region region-code --name my-cluster --query
 "cluster.status"
```

   Fahren Sie nicht mit dem nächsten Schritt fort, bis die zurückgegebene Ausgabe `ACTIVE` ist.

3. Mit the section called “Schritt 3: kubeconfig aktualisieren” fortfahren

Schritt 3: kubeconfig aktualisieren

1. Wenn Sie Ihren Cluster mit `eksctl` erstellt haben, können Sie diesen Schritt überspringen. Das liegt daran, dass `eksctl` diesen Schritt bereits für Sie durchgeführt hat. Aktivieren Sie `kubectl`, um mit Ihrem Cluster zu kommunizieren, indem Sie einen neuen Kontext zur Datei `kubectl config` hinzufügen. Weitere Informationen zum Erstellen und Aktualisieren der Datei finden Sie unter the section called “Zugriff auf den Cluster mit kubectl”.

```
aws eks update-kubeconfig --region region-code --name my-cluster
```

Eine Beispielausgabe sieht wie folgt aus.

```
Added new context arn:aws: eks:region-code:111122223333:cluster/my-cluster to /home/
username/.kube/config
```

2. Bestätigen Sie die Kommunikation mit Ihrem Cluster, indem Sie den folgenden Befehl ausführen.

```
kubectl get svc
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   28h
```

Schritt 4: Cluster-Einrichtung

1. (Empfohlen) Um einige Amazon EKS-Add-Ons zu verwenden oder einzelnen Kubernetes-Workloads bestimmte AWS Identity and Access Management (IAM) -Berechtigungen zuzuweisen, erstellen Sie einen IAM OpenID Connect (OIDC) -Anbieter für Ihren Cluster. Sie müssen einen IAM-OIDC-Anbieter für Ihren Cluster nur einmal erstellen. Weitere Informationen zu Amazon EKS-Add-ons finden Sie unter the section called “Amazon-EKS-Add-ons”. Weitere Informationen zum Zuweisen bestimmter IAM-Berechtigungen zu Ihren Workloads finden Sie unter the section called “Anmeldeinformationen mit IRSA”.
2. (Empfohlen) Konfigurieren Sie Ihren Cluster für das Amazon VPC CNI-Plug-In für Kubernetes, bevor Sie EC2 Amazon-Knoten in Ihrem Cluster bereitstellen. Standardmäßig wurde das Plugin mit Ihrem Cluster installiert. Wenn Sie EC2 Amazon-Knoten zu Ihrem Cluster hinzufügen, wird das Plugin automatisch auf jedem EC2 Amazon-Knoten bereitgestellt, den Sie hinzufügen. Für das Plugin müssen Sie eine der folgenden IAM-Richtlinien an eine IAM-Rolle anfügen: Wenn Ihr Cluster die `IPv4`-Familie verwendet, verwenden Sie die verwaltete IAM-Richtlinie AmazonEKS_CNI_Policy. Wenn Ihr Cluster die `IPv6`-Familie verwendet, verwenden Sie eine von Ihnen erstellte IAM-Richtlinie.

   Die IAM-Rolle, an die Sie die Richtlinie anhängen, kann die IAM-Rolle des Knotens oder eine dedizierte Rolle sein, die nur für das Plugin verwendet wird. Wir empfehlen, die Richtlinie an diese Rolle anzuhängen. Weitere Informationen zum Erstellen der Rolle finden Sie unter the section called “Konfiguration für IRSA” oder Amazon-EKS-Knoten-IAM-Rolle.

3. Wenn Sie Ihren Cluster mit dem bereitgestellt haben AWS-Managementkonsole, können Sie diesen Schritt überspringen. Der AWS-Managementkonsole stellt standardmäßig das Amazon VPC CNI-Plug-In für Kubernetes-, CoreDNS- und `kube-proxy` Amazon EKS-Add-Ons bereit.

   Wenn Sie Ihren Cluster entweder mit der CLI `eksctl` oder mit der AWS CLI bereitstellen, werden das Amazon VPC CNI-Plug-In für Kubernetes, CoreDNS und selbstverwaltete Add-Ons bereitgestellt. `kube-proxy` Sie können das Amazon-VPC-CNI-Plugin für Kubernetes, CoreDNS und selbstverwaltete `kube-proxy`-Add-Ons, die mit Ihrem Cluster bereitgestellt werden, auf Amazon-EKS-Add-Ons migrieren. Weitere Informationen finden Sie unter the section called “Amazon-EKS-Add-ons”.
4. (Optional) Falls noch nicht geschehen, können Sie Prometheus-Metriken für Ihren Cluster aktivieren. Weitere Informationen finden Sie unter Erstellen eine Scrapers im Benutzerhandbuch von Amazon Managed Service für Prometheus.
5. Wenn Sie beabsichtigen, Workloads auf Ihrem Cluster bereitzustellen, die Amazon-EBS-Volumes verwenden, müssen Sie vor der Bereitstellung der Workloads Amazon EBS CSI auf Ihrem Cluster installieren.

Nächste Schritte

- Der IAM-Prinzipal, der den Cluster erstellt hat, ist der einzige Prinzipal, der Zugriff auf den Cluster hat. Gewähren Sie Berechtigungen für andere IAM-Prinzipalen, damit sie auf Ihren Cluster zugreifen können.
- Wenn der IAM-Prinzipal, der den Cluster erstellt hat, nur über die Mindest-IAM-Berechtigungen verfügt, auf die in den Voraussetzungen verwiesen wird, dann möchten Sie vielleicht weitere Amazon-EKS-Berechtigungen für diesen Prinzipal hinzufügen. Weitere Informationen über das Erteilen von Amazon-EKS-Berechtigungen für IAM-Prinzipals finden Sie unter the section called “IAM-Referenz”.
- Wenn Sie möchten, dass der IAM-Prinzipal, der den Cluster erstellt hat, oder andere Prinzipalen Kubernetes-Ressourcen in der Amazon-EKS-Konsole sehen können, gewähren Sie den Entitäten die erforderlichen Berechtigungen.
- Wenn Sie möchten, dass Knoten und IAM-Prinzipalen von Ihrer VPC aus auf Ihren Cluster zugreifen können, aktivieren Sie den privaten Endpunkt für Ihren Cluster. Standardmäßig ist der öffentliche Endpunkt aktiviert. Falls gewünscht, können Sie den öffentlichen Endpunkt deaktivieren, nachdem Sie den privaten Endpunkt aktiviert haben. Weitere Informationen finden Sie unter the section called “Cluster-Endpunktzugriff”.

- Aktivieren Sie die Secrets-Verschlüsselung für Ihren Cluster.
- Konfigurieren Sie die Protokollierung für Ihren Cluster.
- Fügen Sie Ihrem Cluster Knoten hinzu.

Von Amazon EKS bereitgestellte Kontrollebene

-Übersicht

Amazon EKS Provisioned Control Plane ist eine Funktion, mit der Cluster-Administratoren aus einer Reihe von Skalierungsstufen auswählen und die von ihnen gewählte Stufe für eine sehr hohe, vorhersehbare Leistung von der Steuerungsebene des Clusters aus festlegen können. Auf diese Weise können Clusteradministratoren sicherstellen, dass die Kontrollebene immer mit der angegebenen Kapazität bereitgestellt wird.

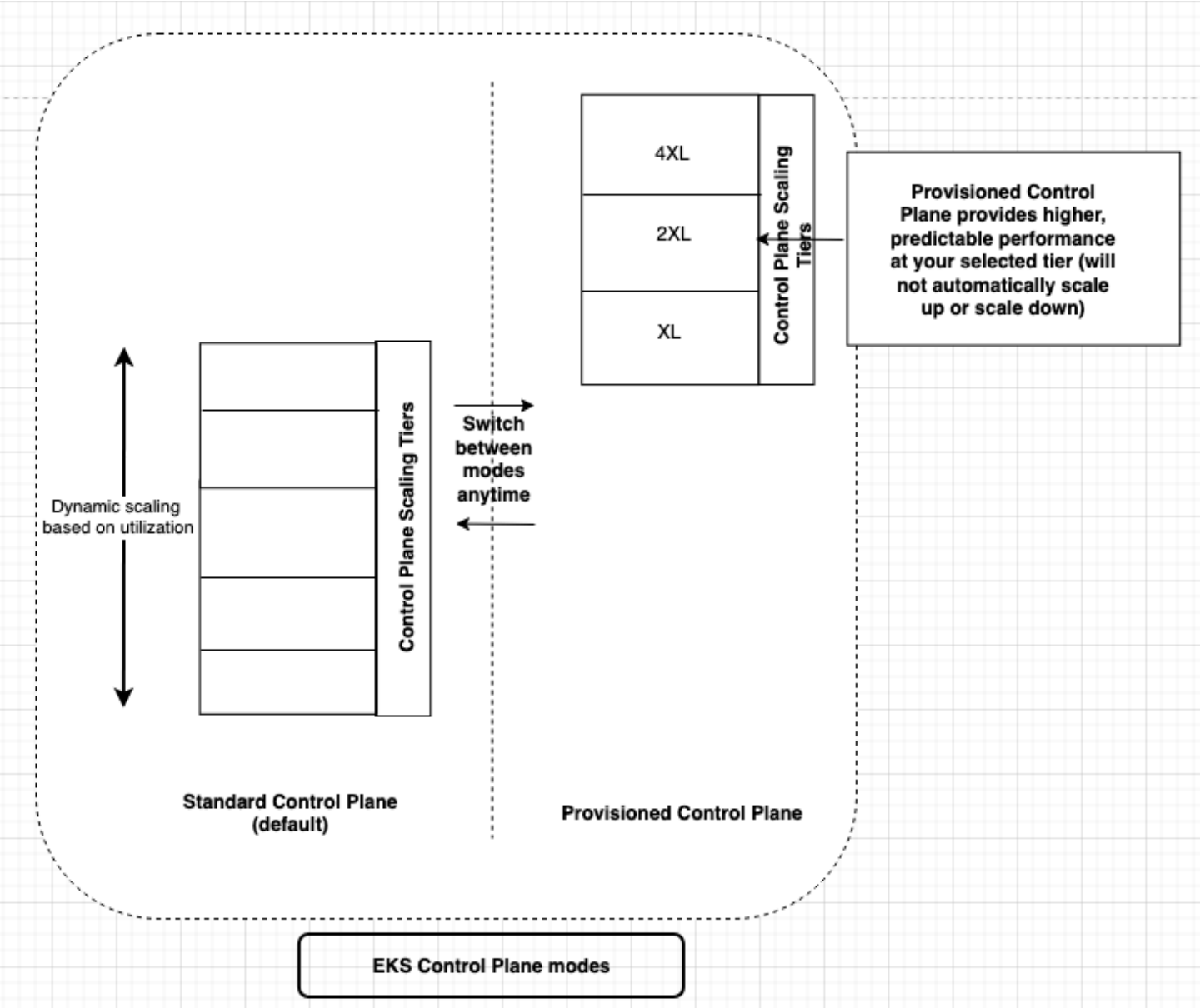
Amazon EKS bietet zwei Betriebsmodi für die Steuerungsebene Ihres Clusters. Standardmäßig verwenden Amazon EKS-Cluster den Standardmodus, in dem die Steuerungsebene je nach Ihren Workload-Anforderungen automatisch nach oben und unten skaliert wird. Der Standardmodus weist dynamisch ausreichend Kapazität auf der Steuerungsebene zu, um Ihren Workload-Anforderungen gerecht zu werden. Dies ist die empfohlene Lösung für die meisten Anwendungsfälle. Für spezielle Workloads, die aufgrund der Skalierung der Steuerungsebene keine Leistungsschwankungen tolerieren können, oder solche, die eine sehr hohe Kapazität der Steuerungsebene erfordern, können Sie jedoch optional den Bereitstellungsmodus verwenden. Im Bereitstellungsmodus können Sie die Kapazität der Steuerungsebene vorab zuweisen, sodass diese stets bereit ist, anspruchsvolle Workload-Anforderungen zu bewältigen.

> **Note**
>
> Der Bereitstellungsmodus ist neben dem standardmäßigen Standardmodus ein zusätzlicher Betriebsmodus auf der Steuerungsebene. Durch die Einführung des Bereitstellungsmodus wird das Verhalten im Standardmodus nicht geändert.

Mit EKS Provisioned Control Plane können Clusteradministratoren die gewünschte Kapazität der Steuerungsebene im Voraus bereitstellen und so eine vorhersehbare und hohe Leistung von der Steuerungsebene des Clusters aus gewährleisten, die immer verfügbar ist. EKS Provisioned Control Plane ermöglicht es Clusteradministratoren außerdem, dieselbe Kapazität der Steuerungsebene in

allen Umgebungen bereitzustellen, vom Staging über die Produktion bis hin zu den Standorten für die Notfallwiederherstellung. Dies ist wichtig, um sicherzustellen, dass die Leistung der Kontrollebene in allen Umgebungen konsistent und vorhersehbar ist. Schließlich bietet Ihnen EKS Provisioned Control Plane Zugriff auf ein sehr hohes Maß an Leistung auf der Kontrollebene und ermöglicht so die Ausführung von massiv skalierbaren KI-Workloads, Hochleistungscomputing und umfangreichen Datenverarbeitungs-Workloads auf Kubernetes.

Alle vorhandenen und neuen Amazon EKS-Cluster arbeiten standardmäßig im Standardmodus. Für Cluster, die eine hohe, vorhersehbare Leistung auf der Kontrollebene erfordern, können Sie sich für die Verwendung der EKS Provisioned Control Plane-Funktion entscheiden. Ihnen wird der Stundensatz für die jeweilige Skalierungsstufe der Steuerungsebene zusätzlich zu den EKS-Stundengebühren für den standardmäßigen oder erweiterten Support in Rechnung gestellt. Weitere Informationen zur Preisgestaltung finden Sie unter Amazon EKS-Preise.



Anwendungsfälle

EKS Provisioned Control Plane wurde für spezifische Szenarien entwickelt, in denen eine hohe und vorhersehbare Leistung der Kontrollebene für Ihren Betrieb entscheidend ist. Wenn Sie diese Anwendungsfälle verstehen, können Sie feststellen, ob EKS Provisioned Control Plane die richtige Lösung für Ihre Workloads ist.

Leistungskritische Workloads — Für Workloads, die minimale Latenz und maximale Leistung von der Kubernetes-Steuerebene erfordern, bietet EKS Provisioned Control Plane Kapazität, die Leistungsschwankungen durch Skalierung der Kontrollebene ausschließt.

Extrem skalierbare Workloads — Wenn Sie hoch skalierbare Workloads wie KI-Training und Inferenz, Hochleistungscomputing oder umfangreiche Datenverarbeitung ausführen, für die eine große Anzahl von Knoten im Cluster erforderlich ist, bietet Provisioned Control Plane die erforderliche Kapazität der Steuerungsebene, um diese anspruchsvollen Workloads zu unterstützen.

Erwartete Ereignisse mit hoher Nachfrage — Wenn Sie aufgrund eines bevorstehenden Ereignisses wie E-Commerce-Verkäufen oder Werbeaktionen, Produkteinführungen, Weihnachtseinkäufen oder großen Sport- oder Unterhaltungsereignissen mit einem plötzlichen Anstieg der Anfragen auf Kontrollebenen rechnen, können Sie mit Provisioned Control Plane die Kapazität Ihrer Kontrollebene im Voraus skalieren. Dieser proaktive Ansatz stellt sicher, dass Ihre Steuerungsebene bereit ist, die erhöhte Last zu bewältigen, ohne auf die automatische Skalierung warten zu müssen, um auf die Nachfrage zu reagieren.

Konsistenz der Umgebung — Mit der bereitgestellten Kontrollebene können Sie die Kapazität und Leistung der Steuerungsebene in allen Staging- und Produktionsumgebungen aufeinander abstimmen, sodass Sie potenzielle Probleme frühzeitig erkennen können, bevor sie in der Produktion eingesetzt werden. Durch die Beibehaltung derselben Kontrollebenenebene in allen Umgebungen können Sie sicherstellen, dass die Testergebnisse das Produktionsverhalten genau widerspiegeln, wodurch das Risiko leistungsbedingter Überraschungen bei der Einführung verringert wird.

Notfallwiederherstellung und Geschäftskontinuität — Für Notfallwiederherstellungsszenarien ermöglicht Ihnen Provisioned Control Plane die Bereitstellung von Failover-Umgebungen mit derselben Kapazität wie Ihre primäre Umgebung. Dies gewährleistet minimale Unterbrechungen und eine schnelle Wiederherstellung bei Failover-Ereignissen, da Ihr Disaster Recovery-Cluster ab dem Moment der Aktivierung über identische Leistungsmerkmale auf der Steuerungsebene wie Ihr Produktionscluster verfügt.

Skalierungsstufen auf der Steuerungsebene

EKS Provisioned Control Plane bietet Skalierungsstufen, die nach T-Shirt-Größen (XL, 2XL, 4XL) benannt sind. Jede Stufe definiert ihre Kapazität anhand von drei wichtigen Kubernetes-Attributen, die die Leistungsmerkmale der Kontrollebene Ihres Clusters bestimmen. Wenn Sie diese Attribute verstehen, können Sie die passende Stufe für Ihre Workload-Anforderungen auswählen.

Bei der Parallelität von API-Anfragen wird die Anzahl der Anfragen gemessen, die der API-Server der Kubernetes-Steuerebene gleichzeitig verarbeiten kann. Dies ist für Workloads mit hohem Durchsatz von entscheidender Bedeutung.

Die Pod-Scheduling-Rate gibt an, wie schnell der standardmäßige Kubernetes-Scheduler Pods auf Knoten einplanen kann, gemessen in Pods pro Sekunde.

Die Größe der Cluster-Datenbank gibt den Speicherplatz an, der etcd zugewiesen ist, der Datenbank, die den Clusterstatus/die Metadaten enthält.

Wenn Sie die Steuerungsebene Ihres Clusters mithilfe von Provisioned Control Plane auf einer bestimmten Skalierungsebene bereitstellen, stellt EKS sicher, dass die Steuerungsebene Ihres Clusters die dieser Ebene entsprechenden Grenzwerte einhält. Die Grenzen der Skalierungsebenen auf der Kontrollebene variieren je nach Kubernetes-Version, wie in den folgenden Tabellen dargestellt.

EKS v1.28 und v1.29

Skalierungsstufe der bereitgestellten Steuerungsebene	Parallelität von API-Anfragen (Sitze)	Pod-Planungsrate (Pods/Sekunde)	Größe der Cluster-Datenbank (GB)
XL	1700	100	16
2 XL	3400	100	16
4 XL	6800	100	16

EKS v1.30 und höher

Skalierungsstufe der bereitgestellten Steuerungsebene	Parallelität von API-Anfragen (Sitze)	Pod-Planungsrate (Pods/Sekunde)	Größe der Cluster-Datenbank (GB)
XL	1700	167	16
2 XL	3400	283	16
4 XL	6800	400	16

Überwachung der Steuerungsebene, Skalierung der Tier-Auslastung

Amazon EKS bietet mehrere Metriken, mit denen Sie die Tier-Auslastung Ihrer Kontrollebene überwachen können. Diese Metriken werden als CloudWatch Amazon-Metriken veröffentlicht und sind über die CloudWatch EKS-Konsole zugänglich. Darüber hinaus können diese Metriken vom Prometheus-Endpunkt Ihres EKS-Clusters abgerufen werden (siehe hier).

	Prometheus-Metrik	CloudWatch Metrik
Parallelität von API-Anfragen	apiserver_flowcontrol_current_executing_seats	apiserver_flowcontrol_current_executing_seats
Tarif für die Planung von Pods	scheduler_schedule_attempts_total	scheduler_schedule_attempts_total, Scheduler_Schedule_Attempts_Scheduled, Scheduler_Schedule_Attempts_Unschedulable
Größe der Cluster-Datenbank	apiserver_storage_size_bytes	apiserver_speichergröße_bytes

Erläuterung der Tier-Kapazität im Vergleich zur tatsächlichen Leistung

Wenn Sie eine Skalierungsstufe für die bereitgestellte Kontrollebene auswählen, stellen die Tierattribute die zugrunde liegenden Konfigurationen dar, die Amazon EKS auf Ihre Kontrollebene anwendet. Die tatsächliche Leistung, die Sie erzielen, hängt jedoch von Ihren spezifischen Workload-Mustern und Konfigurationen und der Einhaltung der Best Practices von Kubernetes ab. Während bei einer Stufe 4XL beispielsweise API Priority and Fairness (APF) mit 6.800 gleichzeitigen Anforderungsplätzen konfiguriert wird, hängt der tatsächliche Anforderungsdurchsatz, den Sie von der Steuerungsebene erhalten, von der Art der ausgeführten Operationen ab. Kubernetes bestraft beispielsweise Listenanfragen stärker als Get-Anfragen, sodass die effektive Anzahl von Listenanfragen, die gleichzeitig von der Steuerebene verarbeitet werden, geringer ist als die von Get-Anfragen (siehe hier). Auch wenn der Standard-Scheduler-QPS für ein 4XL-Tier auf 400 gesetzt ist, hängt Ihre tatsächliche Pod-Scheduling-Rate von Faktoren wie der Bereitschaft der Knoten und dem Zustand der Knoten für die Planung ab. Um eine optimale Leistung zu erzielen, sollten Sie sicherstellen, dass Ihre Anwendungen den Best Practices von Kubernetes folgen (siehe hier) und dass sie für Ihre Workload-Merkmale richtig konfiguriert sind.

Überlegungen

- Kapazität der Standard-Steuerungsebene — Der Standard-Steuerungsebenenmodus von EKS bietet das beste Preis-Leistungs-Verhältnis und ist die empfohlene Option für die meisten Anwendungsfälle. Für spezielle Workloads, die keine Leistungsschwankungen aufgrund der Skalierung der Steuerungsebene tolerieren können, oder solche, die eine sehr hohe Kapazität der Steuerungsebene erfordern, können Sie jedoch optional den Bereitstellungsmodus in Betracht ziehen.
- Opt-In erforderlich — Bestehende Cluster werden nicht automatisch von der Standard-Steuerebene auf eine teurere EKS Provisioned Control Plane Stufe hochskaliert. Sie müssen sich ausdrücklich für eine der neuen Skalierungsstufen von EKS Provisioned Control Plane entscheiden.
- Ausgangsbeschränkung — Der Standardmodus der Steuerungsebene unterstützt eine Cluster-Datenbankgröße (etcd) von bis zu 8 GB. Wenn die Datenbankgröße Ihres Clusters im Bereitstellungsmodus 8 GB überschreitet, können Sie erst wieder in den Standardmodus wechseln, wenn Sie die Datenbankgröße auf unter 8 GB reduziert haben. Wenn Sie beispielsweise 14 GB Datenbankspeicher im Bereitstellungsmodus verwenden, müssen Sie zunächst die Datenbankauslastung auf weniger als 8 GB reduzieren, bevor Sie zum Standardmodus zurückkehren.
- Keine automatische Stufenskalierung — EKS Provisioned Control Plane skaliert nicht automatisch zwischen den Stufen. Sobald Sie eine Skalierungsstufe ausgewählt haben, bleibt die Steuerungsebene Ihres Clusters an diese Stufe gebunden, sodass eine konsistente und vorhersehbare Leistung gewährleistet ist. Sie haben jedoch die Flexibilität, Ihre eigene Autoscaling-Lösung zu implementieren, indem Sie die Nutzungskennzahlen der Stufen überwachen und mithilfe der EKS Provisioned Control Plane APIs nach oben oder unten skalieren, wenn diese Metriken die von Ihnen definierten Schwellenwerte überschreiten, sodass Sie die volle Kontrolle über Ihre Skalierungsstrategie und Kostenoptimierung haben.
- Aktuelle Stufe anzeigen — Sie können die Amazon EKS-Konsole, Amazon Web Services CLI oder API verwenden, um die aktuelle Skalierungsstufe der Steuerungsebene anzuzeigen. In der CLI können Sie den `describe-cluster` folgenden Befehl ausführen: `aws eks describe-cluster --name cluster-name`
- Zeit für den Tier-Übergang — Sie können die Amazon EKS-Konsole, Amazon EKS oder CLI verwenden APIs, um die Skalierungsstufen zu beenden oder zwischen ihnen zu wechseln. Amazon EKS hat einen neuen Cluster-Aktualisierungstyp namens eingeführt`ScalingTierConfigUpdate`, den Sie überprüfen können, um den Fortschritt der Umstellung zu überwachen. Nachdem Sie einen Befehl zur Änderung der Stufe ausgeführt haben, können Sie die Updates auf dem

Cluster auflisten, um ein neues Update des Typs `ScalingTierConfigUpdate` mit Status zu sehen`Updating`. Der Status `Successful` wird nach Abschluss des Updates oder bei Auftreten eines `Failed` Fehlers auf geändert. Das Fehlerfeld im Update gibt den Grund für den Fehler an. Es gibt keine Einschränkungen, wie oft Sie zwischen den Stufen wechseln können. Das Ändern der Stufe der Steuerungsebene dauert mehrere Minuten.

- Auswahl der optimalen Stufe — Um die optimale Skalierungsstufe der Provisioned Control Plane für Ihren Cluster zu ermitteln, können Sie Lasttests durchführen, indem Sie Ihren Cluster auf der höchsten Stufe (4XL) bereitstellen. Führen Sie anschließend einen Lasttest durch, um Spitzenlasten auf der Steuerungsebene Ihres Clusters zu simulieren. Beobachten Sie die Nutzungsmetriken der Kontrollebene bei Spitzenlast und verwenden Sie diese Beobachtungen als Richtschnur für die Auswahl der geeigneten Stufe für den Bereitstellungsmodus.
- Preise für die bereitgestellte Kontrollebene — Ihnen wird der Stundensatz für die Skalierungsstufe „Provisioned Control Plane“ in Rechnung gestellt, auf der sich Ihr Cluster befindet. Dies gilt zusätzlich zu den stündlichen Gebühren für Standard- oder erweiterten Support. Weitere Informationen finden Sie auf der Amazon EKS-Preisseite.
- Größere Skalierungsstufe — Wenn Sie beabsichtigen, Ihren Cluster auf einer Skalierungsstufe von mehr als 4XL auszuführen, wenden Sie sich an Ihr Amazon Web Services Services-Kundenbetreuungsteam, um weitere Preisinformationen zu erhalten.
- Unterstützung für Kubernetes-Versionen und Regionen — EKS Provisioned Control Plane wird in allen kommerziellen und China Regionen von Amazon Web Services unterstützt. GovCloud Provisioned Control Plane funktioniert auf EKS v1.28 und höher.

Erste Schritte mit der Amazon EKS Provisioned Control Plane

Dieses Handbuch führt Sie durch die Schritte zur Einrichtung und Verwendung der EKS Provisioned Control Plane mithilfe von AWS CLI und AWS-Managementkonsole.

Voraussetzungen

Stellen Sie vor Beginn sicher, dass Sie über Folgendes verfügen:

- AWS CLI — Ein Befehlszeilentool für die Arbeit mit AWS Services, einschließlich Amazon EKS. Weitere Informationen finden Sie unter Installation im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Wir empfehlen, dass Sie die AWS CLI nach der Installation ebenfalls konfigurieren. Weitere Informationen finden Sie unter Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Beachten Sie, dass AWS CLI v2 erforderlich ist, um die auf dieser Seite gezeigte Option update-kubeconfig zu verwenden.

- Erforderliche IAM-Berechtigungen — Der von Ihnen verwendete IAM-Sicherheitsprinzipal muss über Berechtigungen verfügen, um mit Amazon EKS-IAM-Rollen, serviceverknüpften Rollen AWS CloudFormation, einer VPC und verwandten Ressourcen zu arbeiten. Weitere Informationen finden Sie unter Aktionen und Verwenden von serviceverknüpften Rollen im IAM-Benutzerhandbuch. Sie müssen alle Schritte in diesem Handbuch als derselbe Benutzer ausführen. Führen Sie den folgenden Befehl aus, um den aktuellen Benutzer zu überprüfen:

```
aws sts get-caller-identity
```

> **Note**
>
> Wir empfehlen, dass Sie die Schritte in diesem Thema in einer Bash-Shell ausführen. Wenn Sie keine Bash-Shell verwenden, erfordern einige Skript-Befehle wie Zeilenfortsetzungszeichen und die Art und Weise, wie Variablen gesetzt und verwendet werden, eine Anpassung für Ihre Shell. Darüber hinaus können die Zitier- und Escape-Regeln für Ihre Shell unterschiedlich sein. Weitere Informationen finden Sie unter Verwenden von Anführungszeichen mit Zeichenfolgen in der AWS CLI im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle.

Von EKS bereitgestellte Steuerungsebene — CLI AWS

Erstellen Sie einen Cluster mit EKS Provisioned Control Plane Scaling Tier

```
aws eks create-cluster --name prod-cluster \
--role-arn arn:aws:iam::012345678910:role/eks-service-role-AWSServiceRoleForAmazonEKS-
J7ONKE3BQ4PI \
--resources-vpc-config subnetIds=subnet-6782e71e,subnet-
e7e761ac,securityGroupIds=sg-6979fe18 \
--control-plane-scaling-config tier=tier-xl
```

Antwort:

```
{
    "cluster": {
        "name": "my-eks-cluster",
        "arn": "arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster",
        "createdAt": "2024-03-14T11:31:44.348000-04:00",
```

```
        "version": "1.26",
        "endpoint": "https://JSA79429HJDASKJDJ8223829MNDNASW.yl4.us-
east-2.eks.amazonaws.com",
        "roleArn": "arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-cluster-
ServiceRole-zMF6CBakwwbW",
        "resourcesVpcConfig": {
            "subnetIds": [
                "subnet-0fb75d2d8401716e7",
                "subnet-02184492f67a3d0f9",
                "subnet-04098063527aab776",
                "subnet-0e2907431c9988b72",
                "subnet-04ad87f71c6e5ab4d",
                "subnet-09d912bb63ef21b9a"
            ],
            "securityGroupIds": [
                "sg-0c1327f6270afbb36"
            ],
            "clusterSecurityGroupId": "sg-01c84d09d70f39a7f",
            "vpcId": "vpc-0012b8e1cc0abb17d",
            "endpointPublicAccess": true,
            "endpointPrivateAccess": true,
            "publicAccessCidrs": [
                "22.19.18.2/32"
            ]
        },
        "controlPlaneScalingConfig": {
            "tier": "tier-xl"
        },
        "kubernetesNetworkConfig": {
            "serviceIpv4Cidr": "10.100.0.0/16",
            "ipFamily": "ipv4"
        },
        "logging": {
            "clusterLogging": [
                {
                    "types": [
                        "api",
                        "audit",
                        "authenticator",
                        "controllerManager",
                        "scheduler"
                    ],
                    "enabled": true
                }
```

```
            ]
        },
        "identity": {
            "oidc": {
                "issuer": "https://oidc.eks.us-east-2.amazonaws.com/id/
JSA79429HJDASKJDJ8223829MNDNASW"
            }
        },
        "status": "CREATING",
        "certificateAuthority": {
            "data": "CA_DATA_STRING..."
        },
        "platformVersion": "eks.14",
        "tags": {
            "aws:cloudformation:stack-name": "eksctl-my-eks-cluster-cluster",
            "alpha.eksctl.io/cluster-name": "my-eks-cluster",
            "karpenter.sh/discovery": "my-eks-cluster",
            "aws:cloudformation:stack-id": "arn:aws:cloudformation:us-
east-2:111122223333:stack/eksctl-my-eks-cluster-cluster/e752ea00-e217-11ee-
beae-0a9599c8c7ed",
            "auto-delete": "no",
            "eksctl.cluster.k8s.io/v1alpha1/cluster-name": "my-eks-cluster",
            "EKS-Cluster-Name": "my-eks-cluster",
            "alpha.eksctl.io/cluster-oidc-enabled": "true",
            "aws:cloudformation:logical-id": "ControlPlane",
            "alpha.eksctl.io/eksctl-version": "0.173.0-dev
+a7ee89342.2024-03-01T03:40:57Z",
            "Name": "eksctl-my-eks-cluster-cluster/ControlPlane"
        },
        "health": {
            "issues": []
        },
        "accessConfig": {
            "authenticationMode": "API_AND_CONFIG_MAP"
        }
    }
}
```

Sehen Sie sich die Skalierungsstufe der Steuerungsebene des Clusters an

```
aws eks describe-cluster --name prod-cluster
```

Antwort:

```
{
    "cluster": {
        "name": "my-eks-cluster",
        "arn": "arn:aws:eks:us-east-2:111122223333:cluster/my-eks-cluster",
        "createdAt": "2024-03-14T11:31:44.348000-04:00",
        "version": "1.26",
        "endpoint": "https://JSA79429HJDASKJDJ8223829MNDNASW.yl4.us-
east-2.eks.amazonaws.com",
        "roleArn": "arn:aws:iam::111122223333:role/eksctl-my-eks-cluster-cluster-
ServiceRole-zMF6CBakwwbW",
        "resourcesVpcConfig": {
            "subnetIds": [
                "subnet-0fb75d2d8401716e7",
                "subnet-02184492f67a3d0f9",
                "subnet-04098063527aab776",
                "subnet-0e2907431c9988b72",
                "subnet-04ad87f71c6e5ab4d",
                "subnet-09d912bb63ef21b9a"
            ],
            "securityGroupIds": [
                "sg-0c1327f6270afbb36"
            ],
            "clusterSecurityGroupId": "sg-01c84d09d70f39a7f",
            "vpcId": "vpc-0012b8e1cc0abb17d",
            "endpointPublicAccess": true,
            "endpointPrivateAccess": true,
            "publicAccessCidrs": [
                "22.19.18.2/32"
            ]
        },
        "controlPlaneScalingConfig": {
            "tier": "tier-xl"
        },
        "kubernetesNetworkConfig": {
            "serviceIpv4Cidr": "10.100.0.0/16",
            "ipFamily": "ipv4"
        },
        "logging": {
            "clusterLogging": [
                {
                    "types": [
                        "api",
                        "audit",
```

```
                        "authenticator",
                        "controllerManager",
                        "scheduler"
                    ],
                    "enabled": true
                }
            ]
        },
        "identity": {
            "oidc": {
                "issuer": "https://oidc.eks.us-east-2.amazonaws.com/id/
JSA79429HJDASKJDJ8223829MNDNASW"
            }
        },
        "status": "ACTIVE",
        "certificateAuthority": {
            "data": "CA_DATA_STRING..."
        },
        "platformVersion": "eks.14",
        "tags": {
            "aws:cloudformation:stack-name": "eksctl-my-eks-cluster-cluster",
            "alpha.eksctl.io/cluster-name": "my-eks-cluster",
            "karpenter.sh/discovery": "my-eks-cluster",
            "aws:cloudformation:stack-id": "arn:aws:cloudformation:us-
east-2:111122223333:stack/eksctl-my-eks-cluster-cluster/e752ea00-e217-11ee-
beae-0a9599c8c7ed",
            "auto-delete": "no",
            "eksctl.cluster.k8s.io/v1alpha1/cluster-name": "my-eks-cluster",
            "EKS-Cluster-Name": "my-eks-cluster",
            "alpha.eksctl.io/cluster-oidc-enabled": "true",
            "aws:cloudformation:logical-id": "ControlPlane",
            "alpha.eksctl.io/eksctl-version": "0.173.0-dev
+a7ee89342.2024-03-01T03:40:57Z",
            "Name": "eksctl-my-eks-cluster-cluster/ControlPlane"
        },
        "health": {
            "issues": []
        },
        "accessConfig": {
            "authenticationMode": "API_AND_CONFIG_MAP"
        }
    }
}
```

Aktualisieren Sie den Cluster so, dass er die EKS Provisioned Control Plane verwendet

```
aws eks update-cluster-config --name prod-cluster \
--control-plane-scaling-config tier=tier-2xl
```

Antwort:

```
{
    "update": {
        "id": "7551c64b-1d27-4b1e-9f8e-c45f056eb6fd",
        "status": "InProgress",
        "type": "ScalingTierConfigUpdate",
        "params": [
            {
                "type": "UpdatedTier",
                "value": "tier-2xl"
            },
            {
                "type": "PreviousTier",
                "value": "tier-xl"
            }
        ],
        "createdAt": 1565807210.37,
        "errors": []
    }
}
```

Update zur Skalierung der Steuerungsebene anzeigen

```
aws eks list-updates --name example
```

Antwort:

```
{
    "updateIds": [
        "7551c64b-1d27-4b1e-9f8e-c45f056eb6fd1"
    ]
}
```

Verlassen Sie die bereitgestellte Steuerungsebene auf die Standard-Steuerungsebene

```
aws eks update-cluster-config --name prod-cluster \
```

```
--control-plane-scaling-config tier=standard
```

Antwort:

```
{
    "update": {
        "id": "7551c64b-1d27-4b1e-9f8e-c45f056eb6fd",
        "status": "InProgress",
        "type": "ScalingTierConfigUpdate",
        "params": [
            {
                "type": "UpdatedTier",
                "value": "standard"
            },
            {
                "type": "PreviousTier",
                "value": "tier-2xl"
            }
        ],
        "createdAt": 1565807210.37,
        "errors": []
    }
}
```

Von EKS bereitgestellte Steuerungsebene - AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie Cluster erstellen.
3. Wählen Sie unter Konfigurationsoptionen die Option Benutzerdefinierte Konfiguration aus.
4. Scrollen Sie nach unten zur Skalierungsstufe der Kontrollebene. Wählen Sie Skalierungsebene verwenden aus, um Provisioned Control Plane zu aktivieren.
5. Wählen Sie aus verschiedenen Skalierungsstufenoptionen wie XL, 2XL und 4XL die Skalierungsstufe der Steuerungsebene aus, die Sie für den Cluster bereitstellen möchten.
6. Wählen Sie nach Bedarf weitere Optionen für die Clusterkonfiguration aus. Wählen Sie im letzten Schritt Cluster erstellen aus. Beachten Sie, dass es mehrere Minuten dauern kann, bis die Clustererstellung abgeschlossen ist.

Vorbereitung auf Kubernetes-Versionsupgrades und Beheben von Fehlkonfigurationen mit Cluster-Einblicken

Amazon-EKS-Cluster-Erkenntnisse bietet die Erkennung von Problemen und Empfehlungen zu deren Behebung, um Sie bei der Verwaltung Ihres Clusters zu unterstützen. Für jeden Amazon-EKS-Cluster werden automatisch wiederkehrende Überprüfungen anhand einer von Amazon EKS zusammengestellten Liste von Erkenntnissen durchgeführt. Diese Erkenntnisprüfungen werden vollständig von Amazon EKS verwaltet und bieten Empfehlungen zum Umgang mit etwaigen Ergebnissen.

Cluster-Erkenntnis-Typen

- Konfigurationserkenntnisse: Identifizieren Fehlkonfigurationen in Ihrer Einrichtung von EKS-Hybridknoten, welche die Funktionalität Ihres Clusters oder Ihrer Workloads beeinträchtigen könnten.
- Upgrade-Erkenntnisse: Identifizieren Probleme, die sich auf Ihre Fähigkeit auswirken könnten, auf neue Versionen von Kubernetes zu aktualisieren.

Überlegungen

- Häufigkeit: Amazon EKS aktualisiert die Cluster-Erkenntnisse alle 24 Stunden. Alternativ können Sie sie manuell aktualisieren, um den neuesten Status anzuzeigen. Sie können beispielsweise Cluster-Erkenntnisse nach der Behebung eines Problems manuell aktualisieren, um zu sehen, ob das Problem behoben wurde.
- Berechtigungen: Amazon EKS erstellt automatisch einen Cluster-Zugriffseintrag für Cluster-Erkenntnisse in jedem EKS-Cluster. Dieser Eintrag gewährt EKS die Berechtigung, Informationen zu Ihrem Cluster anzuzeigen Amazon EKS verwendet diese Informationen, um die Erkenntnisse zu generieren. Weitere Informationen finden Sie unter the section called “Amazon EKSCluster InsightsPolicy”.

Anwendungsfälle

Cluster-Erkenntnisse in Amazon EKS bietet automatisierte Überprüfungen, um die Integrität, Zuverlässigkeit und optimale Konfiguration Ihrer Kubernetes-Cluster zu gewährleisten. Nachfolgend

finden Sie wichtige Anwendungsfälle für Cluster-Erkenntnisse, darunter die Upgrade-Bereitschaft und die Fehlerbehebung bei der Konfiguration.

Upgrade-Einblicke

Upgrade-Erkenntnisse sind eine bestimmte Art von Erkenntnisüberprüfungen innerhalb von Cluster-Erkenntnissen. Diese Überprüfungen liefern Erkenntnisse zur Bereitschaft für ein Kubernetes-Versions-Upgrade. Amazon EKS führt Upgrade-Erkenntnisüberprüfungen in jedem EKS-Cluster durch.

> **Important**
>
> Amazon EKS hat vorübergehend das Feature zurückgesetzt, das bei bestimmten Problemen mit Cluster-Erkenntnissen die Verwendung eines `--force`-Flags für das Upgrade Ihres Clusters erforderlich gemacht hätte. Weitere Informationen finden Sie unter Vorübergehende Rücknahme der Durchsetzung von Upgrade-Erkenntnisse bei der Aktualisierung der Cluster-Version auf GitHub.
> Weitere Informationen zum Aktualisieren Ihres Clusters finden Sie unter the section called “Schritt 3: Cluster-Steuerebene aktualisieren”.

Bevor Sie die Kubernetes-Version Ihres Clusters aktualisieren, können Sie die Registerkarte Upgrade-Erkenntnisse des Beobachtbarkeits-Dashboards in der Amazon-EKS-Konsole verwenden. Wenn in Ihrem Cluster Probleme festgestellt wurden, überprüfen Sie diese und nehmen Sie entsprechende Korrekturen vor. Die Probleme enthalten Links zur Amazon-EKS- und Kubernetes-Dokumentation. Aktualisieren Sie nach der Behebung des Problems die Cluster-Erkenntnisse bei Bedarf, um die neuesten Erkenntnisse abzurufen Wenn alle Probleme behoben wurden, aktualisieren Sie Ihren Cluster.

Amazon EKS liefert Erkenntnisse zur Bereitschaft für ein Kubernetes-Versions-Upgrade. Upgrade-Erkenntnisse identifizieren mögliche Probleme, die sich ggf. auf Kubernetes-Cluster-Upgrades auswirken. Dadurch müssen Administratoren weniger Zeit für die Vorbereitung von Upgrades aufwenden und die Zuverlässigkeit von Anwendungen unter neueren Kubernetes-Versionen erhöht sich. Cluster werden von Amazon EKS automatisch anhand einer Liste möglicher Probleme gescannt, die sich auf das Upgrade der Kubernetes-Version auswirken. Amazon EKS aktualisiert die Liste der Erkenntnisüberprüfungen regelmäßig auf der Grundlage der Änderungen, die in den einzelnen veröffentlichten Kubernetes-Versionen vorgenommen wurden.

Upgrade-Erkenntnisse von Amazon EKS beschleunigen den Test- und Überprüfungsprozess für neue Versionen. Sie ermöglichen es Cluster-Administratoren und Anwendungsentwicklern außerdem, die neuesten Kubernetes-Funktionen zu nutzen, indem sie auf mögliche Probleme hinweisen und Tipps zur Problembehandlung geben.

Konfigurationseinblicke

EKS-Cluster-Erkenntnisse führen automatische Scans von Amazon-EKS-Clustern mit Hybridknoten durch, um Konfigurationsprobleme zu identifizieren, welche die Kommunikation zwischen der Kubernetes-Steuerebene und Webhooks, kubectl-Befehle wie exec und Protokolle und vieles mehr beeinträchtigen. Konfigurationserkenntnisse decken Probleme auf und liefern Empfehlungen zur Behebung, wodurch die Zeit bis zur vollständigen Funktionsfähigkeit der Hybridknoten-Konfiguration verkürzt wird.

Erste Schritte

Um die Liste der durchgeführten Einblickprüfungen und alle relevanten Probleme anzuzeigen, die Amazon EKS identifiziert hat, können Sie die AWS-Managementkonsole, die AWS-CLI, AWS-SDKs und die Amazon-EKS-`ListInsights`-API-Operation verwenden. Um zu beginnen, sehen Sie sich the section called “Cluster-Erkenntnisse anzeigen” an.

Cluster-Erkenntnisse anzeigen

Amazon EKS bietet zwei Arten von Einblicken: Konfigurationseinblicke und Upgrade-Einblicke. Konfigurationseinblicke identifizieren Fehlkonfigurationen in Ihrer Einrichtung von EKS Hybrid Nodes, welche die Funktionalität Ihres Clusters oder Ihrer Workloads beeinträchtigen könnten. Upgrade-Einblicke identifizieren Probleme, die sich auf Ihre Fähigkeit auswirken könnten, auf neue Versionen von Kubernetes zu aktualisieren.

Um die Liste der durchgeführten Erkenntnisprüfungen und alle relevanten Probleme anzuzeigen, die Amazon EKS identifiziert hat, können Sie die Suche in der AWS-Managementkonsole-, der AWS-CLI-, AWS-SDKs- und Amazon-EKS-`ListInsights`-API-Operation aufrufen.

Konfigurationserkenntnisse anzeigen (Konsole)

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie in der Clusterliste den Namen des Amazon-EKS-Clusters aus, für den Sie die Erkenntnisse anzeigen möchten.
3. Wählen Sie Cluster überwachen aus.

4. Wählen Sie die Registerkarte Cluster-Zustand aus.
5. In der Tabelle Konfigurationsinformationen werden die folgenden Spalten angezeigt:
   - Name: Die Überprüfung, die von Amazon EKS für den Cluster durchgeführt wurde.
   - Erkenntnisstatus – Eine Erkenntnis mit dem Status `Error` weist auf eine Fehlkonfiguration hin, die wahrscheinlich die Cluster-Funktionalität beeinträchtigt. Eine Erkenntnis mit dem Status `Warning` bedeutet, dass die Konfiguration nicht dem dokumentierten Ansatz entspricht, die Cluster-Funktionalität jedoch möglicherweise funktioniert, wenn Sie sie absichtlich konfiguriert haben. Eine Erkenntnis mit dem Status `Passing` bedeutet, dass Amazon EKS keine Probleme im Zusammenhang mit dieser Erkenntnisüberprüfung in Ihrem Cluster festgestellt hat.
   - Version – Die anwendbare Version.
   - Zeit der letzten Aktualisierung – Der Zeitpunkt, zu dem der Status der Erkenntnis für diesen Cluster zuletzt aktualisiert wurde.
   - Beschreibung: Informationen der letzten Erkenntnisüberprüfung (einschließlich der Warnung und der empfohlenen Behandlungsmaßnahmen).

Upgrade-Einblicke anzeigen (Konsole)

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie in der Clusterliste den Namen des Amazon-EKS-Clusters aus, für den Sie die Erkenntnisse anzeigen möchten.
3. Wählen Sie Cluster überwachen aus.
4. Wählen Sie die Registerkarte Upgrade-Erkenntnisse aus.
5. Um die neuesten Daten anzuzeigen, klicken Sie auf die Schaltfläche Erkenntnisse aktualisieren und warten Sie, bis der Aktualisierungsvorgang abgeschlossen ist.
6. In der Tabelle Upgrade-Erkenntnisse werden die folgenden Spalten angezeigt:
   - Name: Die Überprüfung, die von Amazon EKS für den Cluster durchgeführt wurde.
   - Erkenntnisstatus: – Eine Erkenntnis mit dem Status „Fehler“ bedeutet normalerweise, dass die betroffene Kubernetes-Version N+1 der aktuellen Cluster-Version ist, während der Status „Warnung“ bedeutet, dass die Erkenntnis für eine zukünftige Kubernetes-Version N+2 oder höher gilt. Eine Erkenntnis mit dem Status „Bestanden“ bedeutet, dass Amazon EKS keine Probleme im Zusammenhang mit dieser Erkenntnisüberprüfung in Ihrem Cluster festgestellt hat. Bei einer Erkenntnis mit dem Status „Unbekannt“ kann Amazon EKS nicht feststellen, ob Ihr Cluster von dieser Erkenntnisüberprüfung betroffen ist.

- Version – Die Kubernetes-Version, die das Erkenntnis auf mögliche Probleme überprüft hat.
- Zeit der letzten Aktualisierung – Der Zeitpunkt, zu dem der Status der Erkenntnis für diesen Cluster zuletzt aktualisiert wurde.
- Letzte Übergangszeit – Der Zeitpunkt, zu dem sich der Status dieser Erkenntnis zuletzt geändert hat.
- Beschreibung: Informationen der letzten Erkenntnisüberprüfung (einschließlich der Warnung und der empfohlenen Behandlungsmaßnahmen).

Cluster-Erkenntnisse anzeigen (AWS-CLI)

1. Um die neuesten Daten anzuzeigen, aktualisieren Sie die Erkenntnisse für einen angegebenen Cluster. Nehmen Sie nach Bedarf die folgenden Änderungen am Befehl vor und führen Sie anschließend den geänderten Befehl aus:
   - Ersetzen Sie *region-code* durch den Code für Ihre AWS-Region.
   - Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters.

   ```
   aws eks start-insight-refresh --region region-code --cluster-name my-cluster
   ```

2. Um den Status einer Erkenntnisaktualisierung zu verfolgen, führen Sie den folgenden Befehl aus. Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters.

   ```
   aws eks describe-insights-refresh --cluster-name my-cluster
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   {
       "message": "Insights refresh is in progress",
       "status": "IN_PROGRESS",
       "startedAt": "2025-07-30T13:36:09-07:00"
   }
   ```

3. Listen Sie die Erkenntnisse für einen angegebenen Cluster auf. Nehmen Sie nach Bedarf die folgenden Änderungen am Befehl vor und führen Sie anschließend den geänderten Befehl aus:
   - Ersetzen Sie *region-code* durch den Code für Ihre AWS-Region.
   - Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters.

```
aws eks list-insights --region region-code --cluster-name my-cluster
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
"insights":
    [
        {
            "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "name": "Deprecated APIs removed in Kubernetes vX.XX",
            "category": "UPGRADE_READINESS",
            "kubernetesVersion": "X.XX",
            "lastRefreshTime": 1734557315.000,
            "lastTransitionTime": 1734557309.000,
            "description": "Checks for usage of deprecated APIs that are scheduled
 for removal in Kubernetes vX.XX. Upgrading your cluster before migrating to the
 updated APIs supported by vX.XX could cause application impact.",
            "insightStatus":
            {
                "status": "PASSING",
                "reason": "No deprecated API usage detected within the last 30
 days.",
            },
        },
        {
            "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "name": "Kubelet version skew",
            "category": "UPGRADE_READINESS",
            "kubernetesVersion": "X.XX",
            "lastRefreshTime": 1734557309.000,
            "lastTransitionTime": 1734557309.000,
            "description": "Checks for kubelet versions of worker nodes in the
 cluster to see if upgrade would cause non compliance with supported Kubernetes
 kubelet version skew policy.",
            "insightStatus":
            {
                "status": "UNKNOWN",
                "reason": "Unable to determine status of node kubelet versions.",
            },
        },
        {
```

```
            "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
            "name": "Deprecated APIs removed in Kubernetes vX.XX",
            "category": "UPGRADE_READINESS",
            "kubernetesVersion": "X.XX",
            "lastRefreshTime": 1734557315.000,
            "lastTransitionTime": 1734557309.000,
            "description": "Checks for usage of deprecated APIs that are scheduled
for removal in Kubernetes vX.XX. Upgrading your cluster before migrating to the
updated APIs supported by vX.XX could cause application impact.",
            "insightStatus":
            {
                "status": "PASSING",
                "reason": "No deprecated API usage detected within the last 30
days.",
            },
        },
        {
            "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
            "name": "Cluster health issues",
            "category": "UPGRADE_READINESS",
            "kubernetesVersion": "X.XX",
            "lastRefreshTime": 1734557314.000,
            "lastTransitionTime": 1734557309.000,
            "description": "Checks for any cluster health issues that prevent
successful upgrade to the next Kubernetes version on EKS.",
            "insightStatus":
            {
                "status": "PASSING",
                "reason": "No cluster health issues detected.",
            },
        },
        {
            "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLEbbbbb",
            "name": "EKS add-on version compatibility",
            "category": "UPGRADE_READINESS",
            "kubernetesVersion": "X.XX",
            "lastRefreshTime": 1734557314.000,
            "lastTransitionTime": 1734557309.000,
            "description": "Checks version of installed EKS add-ons to ensure they
are compatible with the next version of Kubernetes. ",
            "insightStatus": { "status": "PASSING", "reason": "All installed EKS
add-on versions are compatible with next Kubernetes version."},
        },
        {
```

```
            "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLEccccc",
            "name": "kube-proxy version skew",
            "category": "UPGRADE_READINESS",
            "kubernetesVersion": "X.XX",
            "lastRefreshTime": 1734557314.000,
            "lastTransitionTime": 1734557309.000,
            "description": "Checks version of kube-proxy in cluster to see if
 upgrade would cause non compliance with supported Kubernetes kube-proxy version
 skew policy.",
            "insightStatus":
            {
                "status": "PASSING",
                "reason": "kube-proxy versions match the cluster control plane
 version.",
            },
        },
        {
            "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLEddddd",
            "name": "Deprecated APIs removed in Kubernetes vX.XX",
            "category": "UPGRADE_READINESS",
            "kubernetesVersion": "X.XX",
            "lastRefreshTime": 1734557315.000,
            "lastTransitionTime": 1734557309.000,
            "description": "Checks for usage of deprecated APIs that are scheduled
 for removal in Kubernetes vX.XX. Upgrading your cluster before migrating to the
 updated APIs supported by vX.XX could cause application impact.",
            "insightStatus":
            {
                "status": "PASSING",
                "reason": "No deprecated API usage detected within the last 30
 days.",
            },
        },
    ],
"nextToken": null,
}
```


4. Um beschreibende Informationen zu einer Erkenntnis zu erhalten, führen Sie den folgenden Befehl aus. Nehmen Sie nach Bedarf die folgenden Änderungen am Befehl vor und führen Sie anschließend den geänderten Befehl aus:
   - Ersetzen Sie *region-code* durch den Code für Ihre AWS-Region.

- Ersetzen Sie *a1b2c3d4-5678-90ab-cdef-EXAMPLE22222* durch eine Erkenntnis-ID, die Sie aus der Auflistung der Cluster-Erkenntnisse abrufen.
- Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters.

```
aws eks describe-insight --region region-code --id a1b2c3d4-5678-90ab-cdef-
EXAMPLE22222 --cluster-name my-cluster
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
  "insight":
    {
      "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
      "name": "Kubelet version skew",
      "category": "UPGRADE_READINESS",
      "kubernetesVersion": "1.27",
      "lastRefreshTime": 1734557309.000,
      "lastTransitionTime": 1734557309.000,
      "description": "Checks for kubelet versions of worker nodes in the cluster
 to see if upgrade would cause non compliance with supported Kubernetes kubelet
 version skew policy.",
      "insightStatus":
        {
          "status": "UNKNOWN",
          "reason": "Unable to determine status of node kubelet versions.",
        },
      "recommendation": "Upgrade your worker nodes to match the Kubernetes version
 of your cluster control plane.",
      "additionalInfo":
        {
          "Kubelet version skew policy": "https://kubernetes.io/releases/version-
skew-policy/#kubelet",
          "Updating a managed node group": "https://docs.aws.amazon.com/eks/latest/
userguide/update-managed-node-group.html",
        },
      "resources": [],
      "categorySpecificSummary":
        { "deprecationDetails": [], "addonCompatibilityDetails": [] },
    },
}
```

Aktualisierung des vorhandenen Clusters auf die neue Kubernetes-Version

> **Tip**
>
> Melden Sie sich für bevorstehende Amazon EKS-Workshops an.

Wenn eine neue Kubernetes-Version in Amazon EKS verfügbar ist, können Sie Ihren Amazon-EKS-Cluster auf die neueste Version aktualisieren.

> **Important**
>
> Nach der Aktualisierung eines Clusters ist ein Downgrade auf eine frühere Version nicht mehr möglich. Bevor Sie auf eine neue Kubernetes-Version aktualisieren, empfehlen wir Ihnen, die Informationen unter Kubernetes-Versionslebenszyklus in EKS verstehen und die Aktualisierungsschritte in diesem Thema zu lesen.

Neue Kubernetes-Versionen führen oft bedeutende Änderungen ein. Daher empfehlen wir Ihnen, das Verhalten Ihrer Anwendungen mit einer neuen Kubernetes-Version zu testen, bevor Sie Ihre Produktionscluster aktualisieren. Hierzu erstellen Sie einen kontinuierlichen Integrations-Workflow, um das Verhalten Ihrer Anwendungen zu testen, bevor Sie auf eine neue Kubernetes-Version aktualisieren.

Der Aktualisierungsprozess besteht darin, dass Amazon EKS neue API-Serverknoten mit der aktualisierten Kubernetes-Version startet, um die vorhandenen zu ersetzen. Amazon EKS führt auf diesen neuen Knoten standardmäßige Infrastruktur- und Bereitschafts-Zustandsprüfungen für den Netzwerkverkehr durch, um sicherzustellen, dass sie wie erwartet funktionieren. Sobald Sie das Cluster-Upgrade gestartet haben, können Sie es jedoch weder anhalten noch beenden. Wenn eine dieser Prüfungen fehlschlägt, macht Amazon EKS die Infrastruktur-Bereitstellung rückgängig und der Cluster verbleibt in der vorherigen Kubernetes-Version. Laufende Anwendungen sind davon nicht betroffen und Ihr Cluster befindet sich nie in einem nicht deterministischen oder nicht wiederherstellbaren Zustand. Amazon EKS sichert regelmäßig alle verwalteten Cluster, und es gibt Mechanismen, um Cluster bei Bedarf wiederherzustellen. Wir evaluieren und verbessern unsere Verwaltungsprozesse für die Kubernetes-Infrastruktur fortlaufend.

Um den Cluster zu aktualisieren, benötigt Amazon EKS bis zu fünf verfügbare IP-Adressen aus den Subnetzen, die beim Erstellen des Clusters bereitgestellt wurden. Amazon EKS erstellt in jedem der von Ihnen angegebenen Subnetze neue Elastic-Network-Schnittstellen für den Cluster (Netzwerkschnittstellen). Die Netzwerkschnittstellen können in anderen Subnetzen erstellt werden als Ihre vorhandenen Netzwerkschnittstellen. Stellen Sie daher sicher, dass Ihre Sicherheitsgruppenregeln die erforderliche Cluster-Kommunikation für jedes der Subnetze zulassen, die Sie bei der Erstellung Ihres Clusters angegeben haben. Wenn eines der Subnetze, die Sie bei der Erstellung des Clusters angegeben haben, nicht vorhanden ist, nicht genügend verfügbare IP-Adressen hat oder nicht über Sicherheitsgruppenregeln verfügt, welche die notwendige Cluster-Kommunikation zulassen, kann die Aktualisierung fehlschlagen.

Um sicherzustellen, dass der API-Server-Endpunkt für Ihren Cluster immer zugänglich ist, bietet Amazon EKS eine hochverfügbare Kubernetes-Steuerebene und führt während Aktualisierungsvorgängen fortlaufende Aktualisierungen der API-Server-Instances durch. Um wechselnde IP-Adressen von API-Server-Instances zu berücksichtigen, die Ihren Kubernetes-API-Server-Endpunkt unterstützen, müssen Sie sicherstellen, dass Ihre API-Server-Clients die Wiederverbindung effektiv verwalten. Aktuelle Versionen von `kubectl` und der Kubernetes-Client-Bibliotheken, die offiziell unterstützt werden, führen diesen Vorgang zur Wiederherstellung der Verbindung transparent durch.

> **Note**
>
> Weitere Informationen zu den Inhalten einer Cluster-Aktualisierung finden Sie unter Bewährte Methoden für Cluster-Upgrades im EKS-Leitfaden für bewährte Methoden. Diese Ressource unterstützt Sie bei der Planung eines Upgrades und vermittelt Ihnen die Strategie für das Upgraden eines Clusters.

Überlegungen zu Amazon EKS Auto Mode

- Die Rechenleistung von Amazon EKS Auto Mode steuert die Kubernetes-Version der Knoten. Nach dem Upgrade der Steuerebene beginnt EKS Auto Mode mit der schrittweisen Aktualisierung der verwalteten Knoten. EKS Auto Mode berücksichtigt Budgets für Pod-Unterbrechung.
- Sie müssen die Funktionen von Amazon EKS Auto Mode, einschließlich der automatischen Skalierung der Rechenleistung, des Blockspeichers und der Load-Balancing-Funktionen, nicht manuell aktualisieren.

Zusammenfassung

Die allgemeine Zusammenfassung des Upgrade-Prozesses für Amazon-EKS-Cluster lautet wie folgt:

1. Stellen Sie sicher, dass sich Ihr Cluster in einem Zustand befindet, der ein Upgrade unterstützt. Dazu gehört die Überprüfung der Kubernetes-Ressourcen, die von den im Cluster bereitgestellten Ressourcen APIs verwendet werden, um sicherzustellen, dass der Cluster frei von Gesundheitsproblemen ist. Nutzen Sie die Upgrade-Informationen von Amazon EKS, um die Upgrade-Bereitschaft Ihres Clusters zu bewerten.
2. Aktualisieren Sie die Steuerebene auf die nächste Nebenversion (z. B. von 1.32 auf 1.33).
3. Aktualisieren Sie die Knoten in der Datenebene, sodass sie mit denen der Steuerebene übereinstimmen.
4. Aktualisieren Sie alle weiteren Anwendungen, die in dem Cluster ausgeführt werden (z. B. `cluster-autoscaler`).
5. Aktualisieren Sie die von Amazon EKS bereitgestellten Add-Ons, beispielsweise die standardmäßig enthaltenen:
   - Empfohlene Amazon-VPC-CNI-Version
   - Empfohlene CoreDNS-Version
   - Von `kube-proxy` empfohlene Version
6. Aktualisieren Sie alle Clients, die mit dem Cluster kommunizieren (z. B. `kubectl`).

Schritt 1: Upgrade vorbereiten

Vergleichen Sie die Kubernetes-Version Ihrer Cluster-Steuerebene mit der Kubernetes-Version Ihrer Knoten.

- Rufen Sie die Kubernetes-Version Ihrer Cluster-Steuerebene ab.

```
kubectl version
```

- Rufen Sie die Kubernetes-Version Ihrer Knoten ab. Dieser Befehl gibt alle selbstverwalteten und verwalteten Amazon- EC2, Fargate- und Hybrid-Knoten zurück. Jeder Fargate-Pod wird als eigener Knoten aufgeführt.

```
kubectl get nodes
```

Bevor Sie Ihre Steuerebene auf eine neue Kubernetes-Version aktualisieren, stellen Sie sicher, dass die Kubernetes-Minor-Version sowohl der verwalteten Knoten als auch der Fargate-Knoten in Ihrem Cluster mit der Version Ihrer Steuerebene übereinstimmt. Wenn auf Ihrer Steuerungsebene beispielsweise Version ausgeführt wird `1.29` und auf einem Ihrer Knoten Version ausgeführt wird`1.28`, müssen Sie Ihre Knoten auf Version aktualisieren, `1.29` bevor Sie Ihre Kontrollebene auf Version 1.30 aktualisieren. Wir empfehlen Ihnen außerdem, Ihre selbstverwalteten Knoten und Hybridknoten auf dieselbe Version wie Ihre Steuerebene zu aktualisieren, bevor Sie die Steuerebene aktualisieren. Weitere Informationen finden Sie unter the section called “Aktualisierung”, the section called “Update-Methoden” und the section called “Hybridknoten aktualisieren”. Wenn Sie Fargate-Knoten mit einer Nebenversion haben, die niedriger ist als die Version der Steuerebene, löschen Sie zunächst den Pod, der durch den Knoten dargestellt wird. Aktualisieren Sie dann Ihre Steuerebene. Alle verbleibenden Pods werden auf die neue Version aktualisiert, nachdem Sie sie neu bereitgestellt haben.

Schritt 2: Überlegungen zum Upgrade prüfen

Amazon-EKS-Cluster-Erkenntnisse überprüfen Cluster automatisch anhand einer Liste potenzieller Probleme, die sich auf Upgrades der Kubernetes-Version auswirken können, wie beispielsweise die Verwendung veralteter Kubernetes-APIs Amazon EKS aktualisiert regelmäßig die Liste der durchzuführenden Erkenntnisprüfungen basierend auf Bewertungen von Änderungen im Kubernetes-Projekt. Amazon EKS aktualisiert die Liste der Erkenntnisprüfungen auch, wenn Änderungen im Amazon-EKS-Service zusammen mit neuen Versionen eingeführt werden. Weitere Informationen finden Sie unter the section called “Cluster-Einblicke”.

Lesen Sie den Migrationsleitfaden für veraltete APIs in den Kubernetes-Dokumenten.

Upgrade-Erkenntnisse überprüfen

Nutzen Sie die Upgrade-Erkenntnisse von Amazon EKS, um Probleme zu identifizieren. Weitere Informationen finden Sie unter the section called “Upgrade-Einblicke anzeigen (Konsole)”.

Detaillierte Überlegungen

- Da Amazon EKS eine hoch verfügbare Steuerebene ausführt, dürfen Sie jeweils nur um eine Unterversion aktualisieren. Weitere Informationen zu dieser Anforderung finden Sie unter Kubernetes-Version und Version-Skew-Supportrichtlinie. Angenommen, Ihre aktuelle Cluster-Version ist Version `1.28` und Sie möchten sie auf Version `1.30` aktualisieren. Sie müssen zuerst Ihren Version `1.28`-Cluster auf Version `1.29` und dann Ihren Version `1.29`-Cluster auf Version `1.30` aktualisieren.

- Überprüfen Sie den Versionsunterschied zwischen Kubernetes `kube-apiserver` und `kubelet` auf Ihren Knoten.
  - Ab der Kubernetes-Version `1.28` kann `kubelet` bis zu drei Nebenversionen älter sein als `kube-apiserver`. Weitere Informationen finden Sie in der Richtlinie zum Unterschied bei der Kubernetes-Upstream-Version.
  - Wenn das `kubelet` Ihrer verwalteten Knoten und Ihrer Fargate-Knoten mindestens über die `1.25`-Version oder neuer verfügt, können Sie Ihren Cluster um bis zu drei Versionen aktualisieren, ohne die `kubelet`-Version zu aktualisieren. Wenn das `kubelet` also beispielsweise über die Version `1.25` verfügt, können Sie die Version Ihres Amazon EKS-Clusters von `1.25` auf`1.26`, auf `1.27` und auf `1.28` aktualisieren und für `kubelet` die Version `1.25` beibehalten.
- Vergewissern Sie sich daher vor dem Start einer Aktualisierung, dass das `kubelet` auf Ihren Knoten die gleiche Kubernetes-Version hat wie Ihre Steuerebene.
- Wenn Ihr Cluster mit einer Version des Amazon-VPC-CNI-Plugins für Kubernetes konfiguriert ist, die älter als `1.8.0` ist, empfehlen wir Ihnen, das Plugin auf die neueste Version zu aktualisieren, bevor Sie Ihren Cluster aktualisieren. Informationen zum Aktualisieren des Plugins finden Sie unter the section called “Amazon VPC CNI”.
- Sie können eine Sicherungskopie Ihres Amazon EKS-Clusters erstellen, damit Sie Ihren Amazon EKS-Clusterstatus und den persistenten Speicher wiederherstellen können, falls während des Upgrade-Vorgangs Fehler auftreten. Siehe the section called “ AWS Backup”

Schritt 3: Cluster-Steuerebene aktualisieren

Important

Amazon EKS hat vorübergehend das Feature zurückgesetzt, das bei bestimmten Problemen mit Cluster-Erkenntnissen die Verwendung eines `--force`-Flags für das Upgrade Ihres Clusters erforderlich gemacht hätte. Weitere Informationen finden Sie unter Vorübergehendes Rollback der Durchsetzung von Upgrade-Erkenntnissen bei der Aktualisierung der Cluster-Version auf. GitHub
Amazon EKS aktualisiert die Cluster-Erkenntnisse 24 Stunden nach der „letzten Aktualisierungszeit”. Sie können den Zeitpunkt, zu dem Sie ein Problem behoben haben, mit der „letzten Aktualisierungszeit” der Cluster-Übersicht vergleichen.

Darüber hinaus kann es bis zu 30 Tage dauern, bis der Erkenntnis-Status nach der Behebung einer veralteten API-Nutzung aktualisiert wird. Upgrade-Erkenntnisse suchen stets nach veralteter API-Nutzung über einen fortlaufenden Zeitraum von 30 Tagen.

Sie können die Anfrage zum Upgrade Ihrer EKS-Steuerebenen-Version wie folgt senden:

- eksctl
- die Konsole AWS
- die AWS CLI

Cluster aktualisieren – eksctl

Für diesen Vorgang ist `eksctl` Version `0.215.0` oder höher erforderlich. Sie können Ihre -Version mit dem folgenden Befehl überprüfen:

```
eksctl version
```

Eine Installations- und Aktualisierungsanleitung für `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.

Aktualisieren Sie die Kubernetes-Version Ihrer Amazon-EKS-Steuerebene. Ersetzen Sie `<cluster-name>` mit Ihrem Clusternamen. Ersetzen Sie `<version-number>` durch die von Amazon EKS unterstützte Versionsnummer, auf die Sie Ihren Cluster aktualisieren möchten. Eine Liste der unterstützten Versionsnummern finden Sie unter Unterstützte Amazon-EKS-Versionen.

```
eksctl upgrade cluster --name <cluster-name> --version <version-number> --approve
```

Die Aktualisierung dauert einige Minuten.

Fahren Sie fort mit the section called “Schritt 4: Cluster-Komponenten aktualisieren”.

Cluster aktualisieren — AWS Konsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie Jetzt aktualisieren für den Cluster, den Sie aktualisieren möchten.
3. Wählen Sie die Version aus, auf die Ihr Cluster aktualisiert werden soll, und wählen Sie Upgrade.

4. Die Aktualisierung dauert einige Minuten. Fahren Sie fort mit the section called “Schritt 4: Cluster-Komponenten aktualisieren”.

Cluster aktualisieren — AWS CLI

1. Stellen Sie sicher, dass die AWS CLI installiert ist und ob Sie angemeldet sind. Weitere Informationen finden Sie unter Installation oder Aktualisierung auf die neueste Version der AWS CLI.
2. Aktualisieren Sie Ihren Amazon EKS-Cluster mit dem folgenden AWS CLI-Befehl. Ersetzen Sie `<cluster-name>` und `<region-code>` des Clusters, den Sie aktualisieren möchten. Ersetzen Sie `<version-number>` durch die von Amazon EKS unterstützte Versionsnummer, auf die Sie Ihren Cluster aktualisieren möchten. Eine Liste der unterstützten Versionsnummern finden Sie unter Unterstützte Amazon-EKS-Versionen.

```
aws eks update-cluster-version --name <cluster-name> \
  --kubernetes-version <verion-number> --region <region-code>
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "update": {
        "id": "<update-id>",
        "status": "InProgress",
        "type": "VersionUpdate",
        "params": [
            {
                "type": "Version",
                "value": "<version-number>"
            },
            {
                "type": "PlatformVersion",
                "value": "eks.1"
            }
        ],
[...]
        "errors": []
    }
```

3. Die Aktualisierung dauert einige Minuten. Überwachen Sie den Status Ihres Cluster-Updates mit dem folgenden Befehl. Verwenden Sie zusätzlich zur Verwendung desselben `<cluster-name>` und `<region-code>` das `<update-id>`, das der vorherige Befehl zurückgegeben hat.

```
aws eks describe-update --name <cluster-name> \
   --region <region-code> --update-id <update-id>
```

   Wenn der Status `Successful` angezeigt wird, ist das Update abgeschlossen.

4. Fahren Sie fort mit the section called “Schritt 4: Cluster-Komponenten aktualisieren”.

Schritt 4: Cluster-Komponenten aktualisieren

1. Nachdem Ihre Cluster-Aktualisierung abgeschlossen wurde, aktualisieren Sie Ihre Knoten auf dieselbe Kubernetes-Nebenversion wie Ihr aktualisierter Cluster. Weitere Informationen finden Sie unter the section called “Update-Methoden”, the section called “Aktualisierung” und the section called “Hybridknoten aktualisieren”. Alle neuen Pods, die in Fargate gestartet werden, verfügen über eine `kubelet`-Version, die Ihrer Cluster-Version entspricht. Vorhandene Fargate-Pods werden nicht geändert.
2. (Optional) Wenn Sie den Kubernetes Cluster Autoscaler in Ihrem Cluster bereitgestellt haben, bevor Sie den Cluster aktualisiert haben, aktualisieren Sie den Cluster Autoscaler auf die neueste Version, die der Kubernetes-Haupt- und Nebenversion entspricht, auf die Sie aktualisiert haben.
   a. Öffnen Sie die Seite Cluster-Autoscaler-Versionen in einem Web-Browser und suchen Sie die neuste Cluster-Autoscaler-Version, die der Haupt- und Nebenversion Ihres Kubernetes-Clusters entspricht. Wenn beispielsweise die Kubernetes-Version Ihres Clusters `1.30` lautet, suchen Sie die neueste Cluster-Autoscaler-Version, die mit `1.30` beginnt. Notieren Sie die semantische Versionsnummer (z. B. `1.30.n`) für diese Version, um sie im nächsten Schritt zu verwenden.
   b. Legen Sie das Cluster Autoscaler-Abbild-Tag mit dem folgenden Befehl auf die Version fest, die Sie im vorherigen Schritt notiert haben. Ersetzen Sie ggf. `X.XX.X` durch Ihren eigenen Wert.

```
kubectl -n kube-system set image deployment.apps/cluster-autoscaler cluster-
autoscaler=registry.k8s.io/autoscaling/cluster-autoscaler:vX.XX.X
```

3. (Nur Cluster mit GPU-Knoten) Wenn Ihr Cluster Knotengruppen mit GPU-Unterstützung hat (z. B.`p3.2xlarge`), müssen Sie das NVIDIA-Geräte-Plugin für Kubernetes DaemonSet auf Ihrem Cluster aktualisieren. Ersetzen Sie es `<vX.X.X>` durch die gewünschte s-device-pluginNVIDIA/K8-Version, bevor Sie den folgenden Befehl ausführen.

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/<vX.X.X>/
deployments/static/nvidia-device-plugin.yml
```

4. Aktualisieren Sie das Amazon-VPC-CNI-Plugin für Kubernetes, CoreDNS und `kube-proxy`-Add-Ons. Wir empfehlen, die Add-Ons auf die Mindestversionen zu aktualisieren, die unter Servicekonto-Tokens aufgeführt sind.
   - Wenn Sie Amazon-EKS-Add-Ons verwenden, wählen Sie Clusters (Cluster) in der Amazon-EKS-Konsole und dann im linken Navigationsbereich den Namen des Clusters aus, den Sie aktualisiert haben. In der Konsole werden Benachrichtigungen angezeigt. Diese informieren Sie darüber, dass für jedes Add-On, für das eine Aktualisierung verfügbar ist, eine neue Version verfügbar ist. Um ein Add-on zu aktualisieren, wählen Sie die Registerkarte Add-ons aus. Wählen Sie in einem der Felder für ein Add-on, für das eine Aktualisierung verfügbar ist, Jetzt aktualisieren aus, wählen Sie eine verfügbare Version aus, und wählen Sie dann Aktualisieren aus.
   - Alternativ können Sie die AWS CLI verwenden oder `eksctl` Add-Ons aktualisieren. Weitere Informationen finden Sie unter the section called “Add-On aktualisieren”.
5. Sofern erforderlich, aktualisieren Sie Ihre Version von `kubectl`. Sie müssen eine `kubectl`-Version verwenden, die nur in der Minor-Version von Ihrer Amazon–EKS-Cluster-Steuerebene abweicht.

Herabstufen der Kubernetes-Version für einen Amazon-EKS-Cluster

Sie können die Kubernetes-Version eines Amazon-EKS-Clusters nicht herabstufen. Erstellen Sie stattdessen einen neuen Cluster auf einer früheren Amazon-EKS-Version und migrieren Sie die Workloads.

Einen Cluster löschen

Wenn Sie einen Amazon-EKS-Cluster nicht mehr benötigen, sollten Sie die zugehörigen Ressourcen löschen, um unnötige Kosten zu vermeiden.

Sie können einen Cluster mit `eksctl` AWS-Managementkonsole, der oder der AWS CLI löschen.

Überlegungen

- Wenn Sie eine Fehlermeldung erhalten, weil der Cluster-Ersteller entfernt wurde, lesen Sie diesen Artikel.
- Amazon Managed Service für Prometheus-Ressourcen liegen außerhalb des Cluster-Lebenszyklus und müssen unabhängig vom Cluster verwaltet werden. Wenn Sie Ihren Cluster löschen, löschen Sie auch alle relevanten Scraper, um damit verbundenen Kosten zu vermeiden. Weitere Informationen finden Sie unter Suchen und Löschen eine Scrapers im Benutzerhandbuch von Amazon Managed Service für Prometheus.
- Informationen zum Entfernen eines verbundenen Clusters finden Sie unter the section called “Einen Cluster abmelden”.

Überlegungen zu EKS Auto Mode

- Alle EKS-Automodus-Knoten werden gelöscht, einschließlich der EC2 verwalteten Instanzen
- Alle Load Balancer werden gelöscht

Weitere Informationen finden Sie unter the section called “EKS Auto Mode deaktivieren”.

Cluster löschen (eksctl)

Für diesen Vorgang ist `eksctl` Version `0.215.0` oder höher erforderlich. Sie können Ihre -Version mit dem folgenden Befehl überprüfen:

```
eksctl version
```

Eine Installations- und Upgrade-Anleitung für `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.

1. Listen Sie alle in Ihrem Cluster ausgeführten Services auf.

   ```
   kubectl get svc --all-namespaces
   ```

   a. Löschen Sie alle Services, die dem Wert `EXTERNAL-IP` zugeordnet sind. Diese Services werden hinter einem Elastic-Load-Balancing-Load-Balancer ausgeführt und müssen in Kubernetes gelöscht werden, damit der Load Balancer und die zugeordneten Ressourcen

korrekt freigegeben werden können. *service-name*Ersetzen Sie es durch den Namen der einzelnen Dienste, die wie beschrieben aufgeführt sind.

```
kubectl delete svc service-name
```

2. Löschen Sie den Cluster und die zugehörigen Knoten mit dem folgenden Befehl. Ersetzen Sie dabei *prod* durch den Namen Ihres Clusters.

```
eksctl delete cluster --name prod
```

Ausgabe:

```
[#]  using region region-code
[#]  deleting EKS cluster "prod"
[#]  will delete stack "eksctl-prod-nodegroup-standard-nodes"
[#]  waiting for stack "eksctl-prod-nodegroup-standard-nodes" to get deleted
[#]  will delete stack "eksctl-prod-cluster"
[#]  the following EKS cluster resource(s) for "prod" will be deleted: cluster. If in
 doubt, check CloudFormation console
```

Cluster löschen (AWS Konsole)

1. Listen Sie alle in Ihrem Cluster ausgeführten Services auf.

```
kubectl get svc --all-namespaces
```

2. Löschen Sie alle Services, die dem Wert `EXTERNAL-IP` zugeordnet sind. Diese Services werden hinter einem Elastic-Load-Balancing-Load-Balancer ausgeführt und müssen in Kubernetes gelöscht werden, damit der Load Balancer und die zugeordneten Ressourcen korrekt freigegeben werden können. *service-name*Ersetzen Sie es durch den Namen der einzelnen Dienste, die wie beschrieben aufgeführt sind.

```
kubectl delete svc service-name
```

3. Alle Knotengruppen und Fargate-Profile löschen.
   a. Öffnen Sie die Amazon-EKS-Konsole.
   b. Wählen Sie im linken Navigationsbereich Clusters (Cluster) für Amazon EKS aus, und wählen Sie dann in der Registerkartenliste der Cluster den Namen des zu löschenden Clusters aus.

c. Wählen Sie auf der Registerkarte Compute (Datenverarbeitung) eine zu löschende Knotengruppe aus. Klicken Sie auf Delete (Löschen), geben Sie den Namen der Knotengruppe ein und wählen Sie anschließend Delete (Löschen) aus. Alle Knotengruppen im Cluster löschen.

> **Note**
>
> Die aufgelisteten Knotengruppen sind nur verwaltete Knotengruppen.

d. Wählen Sie ein zu löschendes Fargate-Profil aus, wählen Sie Delete (Löschen) aus, geben Sie den Namen des Profils ein und wählen Sie dann Delete (Löschen). Alle Fargate-Profile im Cluster löschen.

4. Löschen Sie alle selbstverwalteten AWS CloudFormation Knotenstapel.
   a. Öffnen Sie die AWS CloudFormation -Konsole.
   b. Wählen Sie den Knoten-Stack aus, den Sie löschen möchten, und wählen Sie Delete (Löschen) aus.
   c. Wählen Sie im Bestätigungsdialogfeld Stack löschen Stack löschen aus. Löschen Sie alle selbstverwalteten Knoten-Stacks im Cluster.
5. Löschen Sie den Cluster.
   a. Öffnen Sie die Amazon-EKS-Konsole.
   b. Wählen Sie den Cluster aus, der gelöscht werden soll, und klicken Sie auf Delete (Löschen).
   c. Klicken Sie auf dem Bestätigungsbildschirm zum Löschen des Clusters auf Delete (Löschen).
6. (Optional) Löschen Sie den AWS CloudFormation VPC-Stack.
   a. Öffnen Sie die AWS CloudFormation -Konsole.
   b. Wählen Sie den zu löschenden VPC-Stack und dann Löschen aus.
   c. Wählen Sie im Bestätigungsdialogfeld Stack löschen Stack löschen aus.

Cluster löschen (AWS CLI)

1. Listen Sie alle in Ihrem Cluster ausgeführten Services auf.

```
kubectl get svc --all-namespaces
```

2. Löschen Sie alle Services, die dem Wert `EXTERNAL-IP` zugeordnet sind. Diese Services werden hinter einem Elastic-Load-Balancing-Load-Balancer ausgeführt und müssen in Kubernetes

gelöscht werden, damit der Load Balancer und die zugeordneten Ressourcen korrekt freigegeben werden können. *service-name*Ersetzen Sie es durch den Namen der einzelnen Dienste, die wie beschrieben aufgeführt sind.

```
kubectl delete svc service-name
```

3. Alle Knotengruppen und Fargate-Profile löschen.
   a. Listen Sie die Knotengruppen in Ihrem Cluster mit dem folgenden Befehl auf.

   ```
   aws eks list-nodegroups --cluster-name my-cluster
   ```

   > **Note**
   >
   > Die aufgelisteten Knotengruppen sind nur verwaltete Knotengruppen.

   b. Löschen Sie jede Knotengruppe mit folgendem Befehl. Alle Knotengruppen im Cluster löschen.

   ```
   aws eks delete-nodegroup --nodegroup-name my-nodegroup --cluster-name my-cluster
   ```

   c. Listen Sie die Fargate-Profile in Ihrem Cluster mit dem folgenden Befehl auf.

   ```
   aws eks list-fargate-profiles --cluster-name my-cluster
   ```

   d. Löschen Sie jedes Fargate-Profil mit dem folgenden Befehl. Alle Fargate-Profile im Cluster löschen.

   ```
   aws eks delete-fargate-profile --fargate-profile-name my-fargate-profile --
   cluster-name my-cluster
   ```

4. Löschen Sie alle selbstverwalteten AWS CloudFormation Knotenstapel.
   a. Listen Sie Ihre verfügbaren AWS CloudFormation Stacks mit dem folgenden Befehl auf. Suchen Sie den Vorlagennamen des Knotens in der sich daraus ergebenden Ausgabe.

   ```
   aws cloudformation list-stacks --query "StackSummaries[].StackName"
   ```

   b. Löschen Sie jeden Knoten-Stack mit dem folgenden Befehl und ersetzen Sie dabei *node-stack* durch den Namen Ihres Knoten-Stacks. Löschen Sie alle selbstverwalteten Knoten-Stacks im Cluster.

```
aws cloudformation delete-stack --stack-name node-stack
```

5. Löschen Sie den Cluster mit dem folgenden Befehl und ersetzen Sie dabei *my-cluster* durch den Namen Ihres Clusters.

```
aws eks delete-cluster --name my-cluster
```

6. (Optional) Löschen Sie den AWS CloudFormation VPC-Stack.
   a. Listen Sie Ihre verfügbaren AWS CloudFormation Stacks mit dem folgenden Befehl auf. Suchen Sie die VPC-Vorlage in der sich daraus ergebenden Ausgabe.

```
aws cloudformation list-stacks --query "StackSummaries[].StackName"
```

   b. Löschen Sie den VPC-Stack mit dem folgenden Befehl und ersetzen Sie dabei *my-vpc-stack* durch den Namen Ihres VPC-Stacks.

```
aws cloudformation delete-stack --stack-name my-vpc-stack
```

Schützen Ihrer EKS-Cluster vor versehentlichem Löschen

Das versehentliche Löschen eines EKS-Clusters kann den Betrieb des Kubernetes-Clusters beeinträchtigen.

Sie können EKS-Cluster nun vor versehentlichem Löschen schützen. Wenn Sie den Löschschutz für einen Cluster aktivieren, müssen Sie diesen zunächst deaktivieren, bevor Sie den Cluster löschen können.

Der Löschschutz dient dazu, versehentliches Löschen zu verhindern. Sie sollten sorgfältig festlegen, wer zum Löschen von Clustern berechtigt ist.

Wenn Sie versuchen, einen aktiven Cluster zu löschen, für den der Löschschutz aktiviert ist, erhalten Sie eine `InvalidRequestException`.

> **Important**
>
> Wenn Sie den Löschschutz auf einem Cluster aktivieren, benötigen Sie sowohl die als auch die UpdateClusterConfig DeleteCluster IAM-Berechtigungen, um zuerst den Löschschutz zu entfernen und schließlich den Cluster zu löschen.

> **Note**
>
> Wenn der Cluster-Status „Erstellen“, „Fehler“ oder „Löschen“ lautet, können Sie den Cluster auch dann löschen, wenn der Löschschutz aktiviert ist.

Aktivierung des Löschschutzes für einen vorhandenen Cluster

Sie können dies nur in einem Cluster im aktiven Status ausführen.

```
aws eks update-cluster-config --name <cluster-name> --region <aws-region> --deletion-protection
```

So deaktivieren Sie den Löschschutz für einen vorhandenen Cluster

```
aws eks update-cluster-config --name <cluster-name> --region <aws-region> --no-deletion-protection
```

Cluster-API-Server-Endpunkt

Dieses Thema hilft Ihnen, den privaten Zugriff für den Kubernetes-API-Server-Endpunkt Ihres Amazon-EKS-Clusters zu aktivieren und den öffentlichen Zugriff über das Internet einzuschränken oder vollständig zu deaktivieren.

Wenn Sie einen neuen Cluster erstellen, erstellt Amazon EKS einen Endpunkt für den verwalteten Kubernetes-API-Server, über den Sie mit Ihrem Cluster kommunizieren (mit Kubernetes-Verwaltungswerkzeugen wie `kubectl`). Standardmäßig ist dieser API-Serverendpunkt im Internet öffentlich, und der Zugriff auf den API-Server wird mithilfe einer Kombination aus AWS Identity and Access Management (IAM) und nativer Kubernetes Role Based Access Control (RBAC) gesichert. Dieser Endpunkt wird als öffentlicher Cluster-Endpunkt bezeichnet. Es gibt auch einen privaten

Cluster-Endpunkt. Weitere Informationen zum privaten Cluster-Endpunkt finden Sie im folgenden Abschnitt the section called “Privater Cluster-Endpunkt”.

`IPv6`-Cluster-Endpunkt-Format

EKS erstellt für neue `IPv6`-Cluster, die nach Oktober 2024 erstellt werden, einen eindeutigen Dual-Stack-Endpunkt im folgenden Format: Ein IPv6 Cluster ist ein Cluster, den Sie `IPv6` in der IP-Familie (`ipFamily`) -Einstellung des Clusters auswählen.

Example

AWS

public/private EKS-Cluster-Endpunkt: `eks-cluster.region.api.aws`

AWS GovCloud (US)

public/private EKS-Cluster-Endpunkt: `eks-cluster.region.api.aws`

Amazon Web Services in China

public/private EKS-Cluster-Endpunkt: `eks-cluster.region.api.amazonwebservices.com.cn`

> **Note**
>
> Der Dual-Stack-Cluster-Endpunkt wurde im Oktober 2024 eingeführt. Weitere Informationen zu `IPv6`-Clustern finden Sie unter the section called “IPv6”. Cluster, die vor Oktober 2024 erstellt wurden, verwenden stattdessen das folgende Endpunkt-Format.

`IPv4`-Cluster-Endpunkt-Format

EKS erstellt für jeden Cluster, der in der IP-Familieneinstellung (ipFamily) des Clusters `IPv4` auswählt, einen eindeutigen Endpunkt im folgenden Format:

Example

AWS

public/private EKS-Cluster-Endpunkt `eks-cluster.region.eks.amazonaws.com`

AWS GovCloud (US)

public/private EKS-Cluster-Endpunkt `eks-cluster.region.eks.amazonaws.com`

Amazon Web Services in China

public/private EKS-Cluster-Endpunkt `eks-cluster.region.amazonwebservices.com.cn`

> **Note**
>
> Vor Oktober 2024 verwendeten `IPv6`-Cluster auch dieses Endpunkt-Format. Bei diesen Clustern werden sowohl beim öffentlichen als auch beim privaten Endpunkt nur `IPv4`-Adressen von diesem Endpunkt aufgelöst.

Privater Cluster-Endpunkt

Sie können den privaten Zugriff auf den Kubernetes-API-Server aktivieren, sodass die gesamte Kommunikation zwischen Ihren Knoten und dem API-Server innerhalb Ihrer VPC bleibt. Sie können die IP-Adressen einschränken, die über das Internet auf Ihren API-Server zugreifen können, oder den Internetzugriff auf den API-Server vollständig deaktivieren.

> **Note**
>
> Da dieser Endpunkt für den Kubernetes-API-Server und kein herkömmlicher AWS PrivateLink Endpunkt für die Kommunikation mit einer AWS API ist, wird er in der Amazon VPC-Konsole nicht als Endpunkt angezeigt.

Wenn Sie den privaten Endpunktzugriff für Ihren Cluster aktivieren, erstellt Amazon EKS in Ihrem Namen eine privat gehostete Route-53-Zone und ordnet diese der VPC Ihres Clusters zu. Diese private gehostete Zone wird von Amazon EKS verwaltet und wird nicht in den Route-53-Ressourcen Ihres Kontos angezeigt. Damit die private gehostete Zone Datenverkehr ordnungsgemäß an Ihren API-Server weiterleiten kann, müssen `enableDnsHostnames` und `enableDnsSupport` für Ihre VPC auf `true` gesetzt sein und die DHCP-Optionen für Ihre VPC müssen `AmazonProvidedDNS` in ihrer Domainnamen-Serverliste enthalten. Weitere Informationen finden Sie unter Aktualisieren der DNS-Unterstützung für Ihre VPC im Amazon-VPC-Benutzerhandbuch.

Sie können Ihre Anforderungen für den Zugriff auf Ihre API-Server-Endpunkte definieren, wenn Sie einen neuen Cluster erstellen. Sie können den Zugriff auf die API-Server-Endpunkte für einen Cluster jederzeit aktualisieren.

Ändern des Cluster-Endpunktzugriffs

Verwenden Sie die Verfahren in diesem Abschnitt, um den Endpunktzugriff für einen bestehenden Cluster zu ändern. Die folgende Tabelle zeigt die unterstützten Kombinationen von API-Server-Endpunktzugriffen und das damit verbundene Verhalten.

Endpunkt für öffentlichen Zugriff	Endpunkt für privaten Zugriff	Behavior
Enabled	Disabled	• Dies ist das Standardverhalten für neue Amazon-EKS-Cluster. • Kubernetes-API-Anfragen, die aus der VPC Ihres Clusters stammen (z. B. Knoten zur Steuerung der Steuerebenen-Kommunikation), verlassen zwar die VPC, aber nicht das Netzwerk von Amazon. • Ihr Cluster-API-Server ist über das Internet erreichbar. Sie können optional die Liste des öffentlichen Zugriffs verwenden, CIDRs um den Zugriff auf den öffentlichen Endpunkt anhand einer Liste von CIDR-Blöcken zu steuern. Wenn Sie den Zugriff auf bestimmte CIDR-Blöcke beschränken, empfehlen wir, dass Sie auch den privaten Endpunkt aktivieren oder sicherstellen, dass die von Ihnen angegebenen CIDR-Blöcke die Adressen enthalten, von denen Knoten und Fargate-Pods (falls Sie sie verwenden) auf den öffentlichen Endpunkt zugreifen.
Aktiviert	Aktiviert	• Kubernetes-API-Anfragen innerhalb der VPC Ihres Clusters (z. B. Knoten zur Steuerung der Steuerebe nen-Kommunikation) verwenden den privaten VPC-Endpunkt. Sie verwenden die Regeln für die Cluster-

Endpunkt für öffentlichen Zugriff	Endpunkt für privaten Zugriff	Behavior
		Sicherheitsgruppe, um den Zugriff auf den privaten Endpunkt zu steuern. • Ihr Cluster-API-Server ist über das Internet erreichbar. Sie können optional die Liste des öffentlichen Zugriffs verwenden, CIDRs um den Zugriff auf den öffentlichen Endpunkt anhand einer Liste von CIDR-Blöcken zu steuern. • Wenn Sie Hybridknoten mit Ihrem Amazon EKS-Cluster verwenden, wird nicht empfohlen, sowohl den öffentlichen als auch den privaten Endpunktzugriff zu aktivieren. Da Ihre Hybridknoten außerhalb Ihrer VPC ausgeführt werden, lösen sie den Cluster-Endpunkt in die öffentlichen IP-Adressen auf. Es wird empfohlen, für Cluster mit Hybridknoten entweder den öffentlichen oder den privaten Endpunktzugriff zu verwenden.

Endpunkt für öffentlichen Zugriff	Endpunkt für privaten Zugriff	Behavior
Disabled	Aktiviert	• Der gesamte Datenverkehr zu Ihrem Cluster-API-Server muss aus der VPC des Clusters oder einem verbundenen Netzwerk stammen. • Es gibt keinen öffentlichen Zugriff auf Ihren API-Server aus dem Internet. Alle `kubectl`-Befehle müssen aus der VPC oder einem verbundenen Netzwerk stammen. Die Verbindungsoptionen finden Sie unter the section called “Zugriff auf einen privaten API-Server”. • Sie verwenden die Regeln für die Cluster-Sicherheitsgruppe, um den Zugriff auf den privaten Endpunkt zu steuern. Beachten Sie, dass sich der öffentliche Zugriff CIDRs nicht auf den privaten Endpunkt auswirkt. • Der API-Server-Endpunkt des Clusters wird von öffentlichen DNS-Servern in eine private IP-Adresse von der VPC aufgelöst. In der Vergangenheit konnte der Endpunkt nur innerhalb der VPC aufgelöst werden. Wenn Ihr Endpunkt nicht in eine private IP-Adresse innerhalb der VPC für einen vorhandenen Cluster aufgelöst wird, können Sie: • Den öffentlichen Zugriff aktivieren und ihn dann erneut deaktivieren. Sie müssen dies nur einmal für einen Cluster tun und der Endpunkt wird von diesem Punkt an zu einer privaten IP-Adresse aufgelöst. • Aktualisieren Sie Ihren Cluster.

Zugangskontrollen für Endgeräte

Beachten Sie, dass sich jede der folgenden Methoden zur Steuerung des Endpunktzugriffs nur auf den jeweiligen Endpunkt auswirkt.

Cluster-Sicherheitsgruppe

Die Cluster-Sicherheitsgruppe steuert zwei Arten von Verbindungen: Verbindungen zur Kubelet-API und zum privaten Endpunkt. Die Verbindungen zur `kubelet` API werden in den Befehlen`kubectl attach`,, `kubectl cp` `kubectl exec`, `kubectl logs`, und `kubectl port-forward` verwendet. Die Cluster-Sicherheitsgruppe hat keine Auswirkungen auf den öffentlichen Endpunkt.

Öffentlicher Zugriff CIDRs

Der öffentliche Zugriff CIDRs steuert den Zugriff auf den öffentlichen Endpunkt anhand einer Liste von CIDR-Blöcken. Beachten Sie, dass der öffentliche Zugriff CIDRs keinen Einfluss auf den privaten Endpunkt hat. Der öffentliche Zugriff CIDRs verhält sich auf den `IPv6` Clustern und `IPv4` Clustern je nach dem Datum, an dem sie erstellt wurden, unterschiedlich. Dies wird im Folgenden beschrieben:

CIDR-Blöcke im öffentlichen Endpunkt (**`IPv6`**-Cluster)

Sie können `IPv6` und `IPv4`-CIDR-Blöcke zum öffentlichen Endpunkt eines `IPv6`-Clusters hinzufügen, da der öffentliche Endpunkt Dual-Stack-fähig ist. Dies gilt nur für neue Cluster mit dem Wert `ipFamily` auf `IPv6`, die Sie im Oktober 2024 oder später erstellt haben. Sie können diese Cluster anhand des neuen Endpunkt-Domain-Namens `api.aws` identifizieren.

CIDR-Blöcke im öffentlichen Endpunkt (**`IPv4`**-Cluster)

Sie können dem öffentlichen Endpunkt eines `IPv4`-Clusters `IPv4`-CIDR-Blöcke hinzufügen. Sie können keine `IPv6`-CIDR-Blöcke zum öffentlichen Endpunkt eines `IPv4`-Clusters hinzufügen. Wenn Sie dies versuchen, gibt EKS die folgende Fehlermeldung zurück: `The following CIDRs are invalid in publicAccessCidrs`

CIDR-Blöcke im öffentlichen Endpunkt (**`IPv6`**-Cluster vor Oktober 2024 erstellt)

Sie können `IPv4`-CIDR-Blöcke zum öffentlichen Endpunkt der alten `IPv6`-Cluster hinzufügen, die Sie vor Oktober 2024 erstellt haben. Sie können diese Cluster anhand des `eks.amazonaws.com`-Endpunkts identifizieren. Sie können dem öffentlichen Endpunkt dieser alten `IPv6`-Cluster, die Sie vor Oktober 2024 erstellt haben, keine `IPv6`-CIDR-Blöcke hinzufügen. Wenn Sie dies versuchen, gibt EKS die folgende Fehlermeldung zurück: `The following CIDRs are invalid in publicAccessCidrs`

Zugriff auf einen privaten API-Server

Wenn Sie den öffentlichen Zugriff für den Kubernetes-API-Server-Endpunkt Ihres Clusters deaktiviert haben, können Sie nur über Ihre VPC oder ein verbundenes Netzwerk auf den API-Server zugreifen. Hier sind einige Möglichkeiten, um auf den Kubernetes-API-Server-Endpunkt zuzugreifen:

Verbundenes Netzwerk

Verbinden Sie Ihr Netzwerk über ein AWS -Transit-Gateway oder eine andere Konnektivitätsoption mit der VPC und verwenden Sie dann einen Computer im verbundenen Netzwerk. Sie müssen sicherstellen, dass Ihre Amazon-EKS-Steuerebenen-Sicherheitsgruppe Regeln enthält, die den eingehenden Datenverkehr auf Port 443 von Ihrem verbundenen Netzwerk zulassen.

Gastgeber bei Amazon EC2 Bastion

Sie können eine EC2 Amazon-Instance in einem öffentlichen Subnetz in der VPC Ihres Clusters starten und sich dann über SSH bei dieser Instance anmelden, um Befehle auszuführen. `kubectl` Weitere Informationen finden Sie unter Linux-Bastion-Hosts in AWS. Sie müssen sicherstellen, dass Ihre Amazon-EKS-Steuerebenen-Sicherheitsgruppe Regeln enthält, die den eingehenden Datenverkehr auf Port 443 von Ihrem Bastion-Host zulassen. Weitere Informationen finden Sie unter the section called “Anforderungen an Sicherheitsgruppen”.

Achten Sie bei der Konfiguration `kubectl` für Ihren Bastion-Host darauf, AWS Anmeldeinformationen zu verwenden, die bereits der RBAC-Konfiguration Ihres Clusters zugeordnet sind, oder fügen Sie den IAM-Prinzipal, den Ihre Bastion verwenden wird, zur RBAC-Konfiguration hinzu, bevor Sie den öffentlichen Zugriff auf Endgeräte entfernen. Weitere Informationen erhalten Sie unter the section called “Kubernetes-API-Zugriff” und the section called “Nicht autorisiert oder Zugriff verweigert (`kubectl`)”.

AWS -Cloud9-IDE

AWS Cloud9 ist eine cloudbasierte integrierte Entwicklungsumgebung (IDE), mit der Sie Ihren Code mit nur einem Browser schreiben, ausführen und debuggen können. Sie können eine AWS Cloud9-IDE in der VPC Ihres Clusters erstellen und die IDE für die Kommunikation mit Ihrem Cluster verwenden. Weitere Informationen finden Sie unter Umgebung in AWS Cloud9 erstellen. Sie müssen sicherstellen, dass Ihre Amazon-EKS-Steuerebenen-Sicherheitsgruppe Regeln enthält, die den eingehenden Datenverkehr auf Port 443 von Ihrer IDE-Sicherheitsgruppe zulassen. Weitere Informationen finden Sie unter the section called “Anforderungen an Sicherheitsgruppen”.

Achten Sie bei der Konfiguration `kubectl` für Ihre AWS Cloud9-IDE darauf, AWS Anmeldeinformationen zu verwenden, die bereits der RBAC-Konfiguration Ihres Clusters zugeordnet sind, oder fügen Sie den IAM-Prinzipal, den Ihre IDE verwenden wird, zur RBAC-Konfiguration hinzu, bevor Sie den öffentlichen Endpunktzugriff entfernen. Weitere Informationen erhalten Sie unter the section called “Kubernetes-API-Zugriff” und the section called “Nicht autorisiert oder Zugriff verweigert (`kubectl`)”.

Bearbeiten Sie diese Seite auf GitHub

Konfiguration des Netzwerkzugriffs auf den Endpunkt des Cluster-API-Servers

Sie können den Endpunktzugriff Ihres Cluster-API-Servers mithilfe der AWS-Managementkonsole oder AWS CLI in den folgenden Abschnitten ändern.

Konfigurieren Sie den Endpunktzugriff — AWS Konsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie den Namen des Clusters aus, um Ihre Cluster-Informationen anzuzeigen.
3. Wählen Sie die Registerkarte Netzwerk und anschließend Endpunktzugriff verwalten aus.
4. Wählen Sie für den privaten Zugriff aus, ob Sie den privaten Zugriff für den Kubernetes-API-Server-Endpunkt Ihres Clusters aktivieren oder deaktivieren möchten. Wenn Sie den privaten Zugriff aktivieren, verwenden Kubernetes-API-Anfragen aus der VPC Ihres Clusters den privaten VPC-Endpunkt. Sie müssen den privaten Zugriff aktivieren, um den öffentlichen Zugriff zu deaktivieren.
5. Wählen Sie für Öffentlicher Zugriff aus, ob der öffentliche Zugriff für den Kubernetes-API-Server-Endpunkt Ihres Clusters aktiviert oder deaktiviert werden soll. Wenn Sie den öffentlichen Zugriff deaktivieren, kann der Kubernetes-API-Server Ihres Clusters nur Anfragen aus der VPC des Clusters empfangen.
6. (Optional) Wenn Sie Öffentlicher Zugriff aktiviert haben, können Sie angeben, welche Adressen aus dem Internet mit dem öffentlichen Endpunkt kommunizieren können. Wählen Sie Erweiterte Einstellungen aus. Geben Sie einen CIDR-Block ein, z. B. *`203.0.113.5/32`*. Der Block darf keine reservierten Adressen enthalten. Sie können zusätzliche Blöcke eingeben, indem Sie Quelle hinzufügen auswählen. Es gibt eine maximale Anzahl von CIDR-Blöcken, die Sie angeben können. Weitere Informationen finden Sie unter the section called “Servicekontingente”. Wenn Sie keine Blöcke angeben, empfängt der öffentliche API-Server-Endpunkt Anfragen von allen IP-Adressen

sowohl für IPv4 (0.0.0.0/0) als auch zusätzlich IPv6 (::/0) für Dual-Stack-Cluster IPv6. Wenn Sie den Zugriff auf Ihren öffentlichen Endpunkt mithilfe von CIDR-Blöcken einschränken, wird empfohlen, dass Sie auch den privaten Endpunktzugriff aktivieren, damit Knoten und Fargate-Pods (falls Sie diese verwenden) mit dem Cluster kommunizieren können. Wenn der private Endpunkt nicht aktiviert ist, müssen Ihre CIDR-Quellen für den öffentlichen Zugriffsendpunkt die Ausgangsquellen aus Ihrer VPC enthalten. Wenn Sie beispielsweise einen Knoten in einem privaten Subnetz haben, der über ein NAT-Gateway mit dem Internet kommuniziert, müssen Sie die ausgehende IP-Adresse des NAT-Gateways als Teil eines erlaubten CIDR-Blocks auf Ihrem öffentlichen Endpunkt hinzufügen.

7. Wählen Sie zum Abschluss Update (Aktualisieren) aus.

Endpunktzugriff konfigurieren — AWS CLI

Führen Sie die folgenden Schritte mit der AWS CLI-Version 1.27.160 oder höher aus. Sie können Ihre aktuelle Version mit aws --version überprüfen. Informationen zur Installation oder zum Upgrade der AWS CLI finden Sie unter AWS CLI installieren.

1. Aktualisieren Sie den Endpunktzugriff Ihres Cluster-API-Servers mit dem folgenden AWS CLI-Befehl. Verwenden Ihre eigenen Werte für den Clusternamen und den gewünschten Endpunkt. Wenn Sie endpointPublicAccess=true festlegen, können Sie (optional) einen einzelnen CIDR-Block oder eine kommagetrennte Liste von CIDR-Blöcken für publicAccessCidrs eingeben. Die Blöcke dürfen keine reservierten Adressen enthalten. Wenn Sie CIDR-Blöcke angeben, empfängt der öffentliche API-Server-Endpunkt nur Anforderungen von den aufgelisteten Blöcken. Es gibt eine maximale Anzahl von CIDR-Blöcken, die Sie angeben können. Weitere Informationen finden Sie unter the section called “Servicekontingente”. Wenn Sie den Zugriff auf Ihren öffentlichen Endpunkt mithilfe von CIDR-Blöcken einschränken, wird empfohlen, dass Sie auch den privaten Endpunktzugriff aktivieren, damit Knoten und Fargate-Pods (falls Sie diese verwenden) mit dem Cluster kommunizieren können. Wenn der private Endpunkt nicht aktiviert ist, müssen Ihre CIDR-Quellen für den öffentlichen Zugriffsendpunkt die Ausgangsquellen aus Ihrer VPC enthalten. Wenn Sie beispielsweise einen Knoten in einem privaten Subnetz haben, der über ein NAT-Gateway mit dem Internet kommuniziert, müssen Sie die ausgehende IP-Adresse des NAT-Gateways als Teil eines erlaubten CIDR-Blocks auf Ihrem öffentlichen Endpunkt hinzufügen. Wenn Sie keine CIDR-Blöcke angeben, empfängt der öffentliche API-Server-Endpunkt Anfragen von allen (0.0.0.0/0) IP-Adressen und zusätzlich IPv6 (::/0) für Dual-Stack-IPv6-Cluster.

> **Note**
>
> Der folgende Befehl ermöglicht den privaten Zugriff und den öffentlichen Zugriff von einer einzelnen IP-Adresse für den API-Server-Endpunkt. Ersetzen Sie *203.0.113.5/32* durch einen einzelnen CIDR-Block oder eine kommagetrennte Liste von CIDR-Blöcken, auf die Sie den Netzwerkzugriff beschränken möchten.

```
aws eks update-cluster-config \
    --region region-code \
    --name my-cluster \
    --resources-vpc-config
 endpointPublicAccess=true,publicAccessCidrs="203.0.113.5/32",endpointPrivateAccess=true
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "update": {
        "id": "e6f0905f-a5d4-4a2a-8c49-EXAMPLE00000",
        "status": "InProgress",
        "type": "EndpointAccessUpdate",
        "params": [
            {
                "type": "EndpointPublicAccess",
                "value": "true"
            },
            {
                "type": "EndpointPrivateAccess",
                "value": "true"
            },
            {
                "type": "publicAccessCidrs",
                "value": "[\"203.0.113.5/32\"]"
            }
        ],
        "createdAt": 1576874258.137,
        "errors": []
    }
}
```

2. Überwachen Sie den Status des aktualisierten Endpunktzugriffs mit dem folgenden Befehl unter Verwendung des Cluster-Namens und der Update-ID, die vom vorherigen Befehl zurückgegeben wurden. Ihre Aktualisierung ist abgeschlossen, wenn als Status `Successful` angezeigt wird.

```
aws eks describe-update \
    --region region-code \
    --name my-cluster \
    --update-id e6f0905f-a5d4-4a2a-8c49-EXAMPLE00000
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "update": {
        "id": "e6f0905f-a5d4-4a2a-8c49-EXAMPLE00000",
        "status": "Successful",
        "type": "EndpointAccessUpdate",
        "params": [
            {
                "type": "EndpointPublicAccess",
                "value": "true"
            },
            {
                "type": "EndpointPrivateAccess",
                "value": "true"
            },
            {
                "type": "publicAccessCidrs",
                "value": "[\"203.0.113.5/32\"]"
            }
        ],
        "createdAt": 1576874258.137,
        "errors": []
    }
}
```

Bearbeiten Sie diese Seite auf GitHub

Bereitstellung von Windows-Knoten in EKS-Clustern

Erfahren Sie, wie Sie den Windows-Support für Ihren Amazon-EKS-Cluster aktivieren und verwalten, um Windows-Container neben Linux-Containern auszuführen.

Überlegungen

Beachten Sie vor der Bereitstellung von Windows-Knoten die folgenden Überlegungen.

- EKS Auto Mode unterstützt keine Windows-Knoten
- Sie können Host-Netzwerke auf Windows-Knoten mithilfe von `HostProcess`-Pods verwenden. Weitere Informationen finden Sie HostProcessPod in der Kubernetes-Dokumentation unter Create a Windows.
- Amazon-EKS-Cluster müssen einen oder mehrere Linux- oder Fargate-Knoten enthalten, um Kernsystem-Pods auszuführen, die nur unter Linux, z. B. CoreDNS, ausgeführt werden.
- Die Ereignisprotokolle `kubelet` und `kube-proxy` werden in das `EKS Windows`-Ereignisprotokoll umgeleitet und auf 200 MB begrenzt.
- Sie können Sicherheitsgruppen nicht einzelnen Pods zuweisen, wenn Pods in Windows-Knoten ausgeführt werden.
- Sie können benutzerdefinierte Netzwerke nicht mit Windows-Knoten benutzen.
- Sie können `IPv6` nicht mit Windows-Knoten verwenden.
- Windows-Knoten unterstützen eine Elastic-Network-Schnittstelle pro Knoten. Die Anzahl der Pods, die Sie pro Windows-Knoten ausführen können, entspricht standardmäßig der Anzahl der IP-Adressen, die pro Elastic-Network-Schnittstelle für den Instance-Typ des Knotens verfügbar sind, minus eins. Weitere Informationen finden Sie unter IP-Adressen pro Netzwerkschnittstelle pro Instance-Typ im EC2 Amazon-Benutzerhandbuch.
- In einem Amazon-EKS-Cluster kann ein einzelner Service mit einem Load Balancer bis zu 1 024 Backend-Pods unterstützen. Jeder Pod hat seine eigene eindeutige IP-Adresse. Das bisherige Limit von 64 Pods trifft nicht mehr zu, nachdem ein Windows-Server-Update beginnend mit Betriebssystem-Build 17763.2746 durchgeführt wurde.
- Windows-Container werden in Fargate für Amazon-EKS-Pods nicht unterstützt.
- Es ist nicht möglich, Amazon EKS Hybrid Nodes mit Windows als Betriebssystem für den Host zu verwenden.
- Sie können keine Protokolle aus dem `vpc-resource-controller`-Pod abrufen. Dies war zuvor möglich, als Sie den Controller auf der Datenebene bereitgestellt haben.

- Es gibt eine Abkühlungsphase, bevor einem neuen Pod eine `IPv4`-Adresse zugewiesen wird. Dadurch wird verhindert, dass Datenverkehr aufgrund veralteter `IPv4`-Regeln an einen älteren Pod mit derselben `kube-proxy`-Adresse fließt.
- Die Quelle für den Controller wird auf verwaltet GitHub. Wenn Sie zum Controller beitragen oder Probleme gegen den Controller einreichen möchten, besuchen Sie das Projekt unter GitHub.
- Wenn Sie eine benutzerdefinierte AMI-ID für von Windows verwaltete Knotengruppen angeben, fügen Sie `eks:kube-proxy-windows` diese Ihrer AWS IAM Authenticator-Konfigurationsübersicht hinzu. Weitere Informationen finden Sie unter the section called “Grenzen und Bedingungen bei der Angabe einer AMI-ID”.
- Wenn die Beibehaltung Ihrer verfügbaren IPv4 Adressen für Ihr Subnetz von entscheidender Bedeutung ist, finden Sie weitere Informationen im EKS Best Practices Guide — Windows Networking IP Address Management.
- Überlegungen zu EKS-Zugriffseinträgen
  - Zugriffseinträge zur Verwendung mit Windows-Knoten erfordern den Typ `EC2_WINDOWS`. Weitere Informationen finden Sie unter the section called “Zugriffseinträge erstellen”.

    So erstellen Sie einen Zugriffseintrag für einen Windows-Knoten:

    ```
    aws eks create-access-entry --cluster-name my-cluster --principal-arn arn:aws:
     iam::111122223333:role/<role-name> --type EC2_Windows
    ```

Voraussetzungen

- Einen vorhandenen -Cluster.
- Ihr Cluster muss mindestens einen (wir empfehlen mindestens zwei) Linux-Knoten oder Fargate-Pod haben, um CoreDNS auszuführen. Wenn Sie den veralteten System-Windows-Support aktivieren, müssen Sie einen Linux-Knoten verwenden (Sie können keinen Fargate-Pod verwenden), um CoreDNS auszuführen.
- Ein vorhandener Amazon-EKS-Cluster-IAM-Rolle.

Windows-Support aktivieren

1. Wenn Sie keine Amazon-Linux-Knoten in Ihrem Cluster haben und Sicherheitsgruppen für Pods verwenden, fahren Sie mit dem nächsten Schritt fort. Ansonsten bestätigen Sie, dass die

verwaltete Richtlinie `AmazonEKSVPCResourceController` Ihrer Cluster-Rolle angefügt ist. Ersetzen Sie den *`eksClusterRole`* durch Ihren Clusterrollennamen.

```
aws iam list-attached-role-policies --role-name eksClusterRole
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "AmazonEKSClusterPolicy",
            "PolicyArn": "arn:aws: iam::aws:policy/AmazonEKSClusterPolicy"
        },
        {
            "PolicyName": "AmazonEKSVPCResourceController",
            "PolicyArn": "arn:aws: iam::aws:policy/AmazonEKSVPCResourceController"
        }
    ]
}
```

Wenn die Richtlinie wie in der vorherigen Ausgabe angehängt ist, überspringen Sie den nächsten Schritt.

2. Fügen Sie die von Amazon EKSVPCResource Controller verwaltete Richtlinie Ihrer Amazon EKS-Cluster-IAM-Rolle hinzu. Ersetzen Sie den *`eksClusterRole`* durch Ihren Clusterrollennamen.

```
aws iam attach-role-policy \
  --role-name eksClusterRole \
  --policy-arn arn:aws: iam::aws:policy/AmazonEKSVPCResourceController
```

3. Aktualisieren Sie das VPC-CNI, um Windows ConfigMap IPAM zu aktivieren. Hinweis: Wenn das VPC-CNI mithilfe eines Helm-Charts oder als Amazon EKS-Add-on auf Ihrem Cluster installiert ist, können Sie das möglicherweise nicht direkt ändern. ConfigMap Informationen zur Konfiguration von Amazon-EKS-Add-Ons finden Sie unter the section called “Felder, die angepasst werden können”.

   a. Erstellen Sie eine Datei mit dem Namen *`vpc-resource-controller-configmap.yaml`* und dem folgenden Inhalt.

```
apiVersion: v1
kind: ConfigMap
```

```
metadata:
  name: amazon-vpc-cni
  namespace: kube-system
data:
  enable-windows-ipam: "true"
```

b. Anwendung der `ConfigMap` auf Ihren Cluster.

```
kubectl apply -f vpc-resource-controller-configmap.yaml
```

4. Wenn in Ihrem Cluster der Authentifizierungsmodus so eingestellt ist, dass die `aws-auth`-Konfigurationszuordnung aktiviert wird:

   - Stellen Sie sicher, dass Ihre `aws-auth ConfigMap` eine Zuordnung für die Instance-Rolle des Windows-Knotens enthält, sodass sie die `eks:kube-proxy-windows`-RBAC-Berechtigungsgruppe einschließt. Prüfen Sie dies durch Ausführung des folgenden Befehls.

     ```
     kubectl get configmap aws-auth -n kube-system -o yaml
     ```

     Eine Beispielausgabe sieht wie folgt aus.

     ```
     apiVersion: v1
     kind: ConfigMap
     metadata:
       name: aws-auth
       namespace: kube-system
     data:
       mapRoles: |
         - groups:
           - system:bootstrappers
           - system:nodes
           - eks:kube-proxy-windows # This group is required for Windows DNS resolution
      to work
           rolearn: arn:aws: iam::111122223333:role/eksNodeRole
           username: system:node:{{EC2PrivateDNSName}}
     [...]
     ```

     `eks:kube-proxy-windows` sollte unter „Gruppen“ aufgelistet sein. Wenn die Gruppe nicht angegeben ist, müssen Sie Ihre `ConfigMap` aktualisieren oder sie so erstellen, dass sie die erforderliche Gruppe enthält. Weitere Informationen zur `aws-auth ConfigMap` finden Sie unter the section called “Anwenden von `aws-auth`ConfigMap auf Ihren Cluster”.

5. Wenn in Ihrem Cluster der Authentifizierungsmodus so eingestellt ist, dass die `aws-auth`-Konfigurationszuordnung deaktiviert wird, können Sie EKS-Zugriffseinträge verwenden. Erstellen Sie eine neue Knoten-Rolle für die Verwendung mit Windows-Instances, und EKS erstellt automatisch einen Zugriffseintrag vom Typ `EC2_WINDOWS`.

Windows-Pods bereitstellen

Wenn Sie Pods in Ihrem Cluster bereitstellen, müssen Sie das Betriebssystem angeben, das sie verwenden, wenn Sie eine Kombination aus Knotentypen ausführen.

Verwenden Sie für Linux-Pods den folgenden Knotenauswahltext in Ihren Manifesten.

```
nodeSelector:
        kubernetes.io/os: linux
        kubernetes.io/arch: amd64
```

Verwenden Sie für Windows-Pods den folgenden Knoten-Auswahltext in Ihren Manifesten.

```
nodeSelector:
        kubernetes.io/os: windows
        kubernetes.io/arch: amd64
```

Sie können eine Beispielanwendung bereitstellen, um die verwendeten Knotenselektoren zu sehen.

Unterstützung einer höheren Pod-Dichte in Windows-Knoten

In Amazon EKS wird jedem Pod eine `IPv4`-Adresse aus Ihrer VPC zugewiesen. Aus diesem Grund ist die Anzahl der Pods, die Sie in einem Knoten bereitstellen können, durch die verfügbaren IP-Adressen begrenzt, selbst wenn genügend Ressourcen vorhanden sind, um mehr Pods auf dem Knoten auszuführen. Da von einem Windows-Knoten nur eine elastische Netzwerkschnittstelle unterstützt wird, entspricht die maximale Anzahl verfügbarer IP-Adressen auf einem Windows-Knoten standardmäßig:

```
Number of private IPv4 addresses for each interface on the node - 1
```

Eine IP-Adresse wird als primäre IP-Adresse der Netzwerkschnittstelle verwendet und kann daher Pods nicht zugewiesen werden.

Sie können eine höhere Pod-Dichte in Windows-Knoten aktivieren, indem Sie die IP-Präfix-Delegierung aktivieren. Mit diesem Feature können Sie der primären Netzwerkschnittstelle ein `/28`-`IPv4`-Präfix zuweisen, anstatt sekundäre `IPv4`-Adressen zuzuweisen. Durch die Zuweisung eines IP-Präfixes wird die maximale Anzahl verfügbarer `IPv4`-Adressen auf dem Knoten auf Folgendes erhöht:

```
(Number of private IPv4 addresses assigned to the interface attached to the node - 1) *
 16
```

Angesichts dieser deutlich größeren Anzahl verfügbarer IP-Adressen sollten verfügbare IP-Adressen Ihre Fähigkeit, die Anzahl der Pods auf Ihren Knoten zu skalieren, nicht einschränken. Weitere Informationen finden Sie unter the section called “IP-Adressen erhöhen”.

Windows-Support deaktivieren

1. Wenn Ihr Cluster Amazon-Linux-Knoten enthält und Sie Sicherheitsgruppen für Pods mit ihnen verwenden, überspringen Sie dann diesen Schritt.

   Entfernen Sie die `AmazonVPCResourceController`-verwaltete IAM-Richtlinie von Ihrer Cluster-Rolle. Ersetzen Sie *`eksClusterRole`* durch den Namen Ihrer Cluster-Rolle.

   ```
   aws iam detach-role-policy \
       --role-name eksClusterRole \
       --policy-arn arn:aws:iam::aws:policy/AmazonEKSVPCResourceController
   ```

2. Deaktivieren Sie Windows IPAM im `amazon-vpc-cni`-ConfigMap.

   ```
   kubectl patch configmap/amazon-vpc-cni \
                       -n kube-system \
                       --type merge \
                       -p '{"data":{"enable-windows-ipam":"false"}}'
   ```

Bereitstellung privater Cluster mit eingeschränktem Internetzugang

In diesem Thema wird beschrieben, wie Sie einen Amazon EKS-Cluster bereitstellen, der in der AWS Cloud bereitgestellt wird, aber keinen ausgehenden Internetzugang hat. Wenn Sie einen lokalen Cluster auf AWS Outposts habenthe section called “Knoten”, finden Sie statt dieses Themas weitere Informationen unter.

Wenn Sie mit dem Amazon-EKS-Netzwerk nicht vertraut sind, finden Sie unter Entmystifizierung von Cluster-Netzwerken für Amazon-EKS-Worker-Knoten. Wenn Ihr Cluster nicht über einen ausgehenden Internetzugriff verfügt, muss es die folgenden Anforderungen erfüllen:

Anforderungen an die Cluster-Architektur

- Ihr Cluster muss Images von einer Container-Registry in Ihrer VPC abrufen. Sie können eine Amazon Elastic Container Registry in Ihrer VPC erstellen und Container-Images dorthin kopieren, damit Ihre Knoten daraus abrufen können. Weitere Informationen finden Sie unter the section called “Kopieren eines Images in ein Repository”.
- In Ihrem Cluster muss der private Endpunkt-Zugriff aktiviert sein. Dies ist erforderlich, damit sich Knoten beim Cluster-Endpunkt registrieren können. Der Endpunkt für öffentlichen Zugriff ist optional. Weitere Informationen finden Sie unter the section called “Cluster-Endpunktzugriff”.

Knoten-Anforderungen

- Selbstverwaltete Linux- und Windows-Knoten müssen die folgenden Bootstrap-Argumente enthalten, bevor sie gestartet werden. Diese Argumente umgehen die Amazon-EKS-Introspektion und erfordern keinen Zugriff auf die Amazon-EKS-API innerhalb der VPC.
  a. Bestimmen Sie den Wert des Endpunkts Ihres Clusters mit dem folgenden Befehl. Ersetzen Sie *my-cluster* mit dem Namen Ihres Clusters.

  ```
  aws eks describe-cluster --name my-cluster --query cluster.endpoint --output text
  ```

  Eine Beispielausgabe sieht wie folgt aus.

  ```
  https://EXAMPLE108C897D9B2F1B21D5EXAMPLE.sk1.region-code.eks.amazonaws.com
  ```

  b. Bestimmen Sie den Wert der Zertifizierungsstelle Ihres Clusters mit dem folgenden Befehl. Ersetzen Sie *my-cluster* mit dem Namen Ihres Clusters.

  ```
  aws eks describe-cluster --name my-cluster --query cluster.certificateAuthority --
  output text
  ```

  Die zurückgegebene Ausgabe ist eine lange Zeichenfolge.

  c. Ersetzen Sie die Werte von `apiServerEndpoint` und `certificateAuthority` im NodeConfig Objekt durch die Werte, die in der Ausgabe der vorherigen Befehle zurückgegeben

wurden. Weitere Informationen zur Angabe von Bootstrap-Argumenten beim Starten von selbstverwalteten Amazon Linux 2023-Knoten finden Sie unter the section called “Amazon Linux” und. the section called “Windows”

- Für Linux-Knoten:

```
---
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="BOUNDARY"

--BOUNDARY
Content-Type: application/node.eks.aws

---
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name: my-cluster
    apiServerEndpoint: [.replaceable]https://
EXAMPLE108C897D9B2F1B21D5EXAMPLE.sk1.region-code.eks.amazonaws.com
    certificateAuthority: [.replaceable]Y2VydGlmaWNhdGVBdXRob3JpdHk=
    ...
```

  Weitere Argumente finden Sie im Bootstrap-Skript unter. GitHub

- Für Windows-Knoten:

> **Note**
>
> Wenn Sie CIDR für den benutzerdefinierten Service verwenden, müssen Sie diesen mithilfe des `-ServiceCIDR`-Parameters angeben. Andernfalls schlägt die DNS-Auflösung für Pods im Cluster fehl.

```
-APIServerEndpoint cluster-endpoint -Base64ClusterCA certificate-authority
```

  Weitere Argumente finden Sie unter the section called “Bootstrap-Skript-Konfigurationsparameter”.

- Das `aws-auth ConfigMap` Ihres Clusters muss innerhalb Ihrer VPC erstellt werden. Weitere Informationen zum Erstellen und Hinzufügen von Einträgen zu `aws-auth ConfigMap` erhalten Sie durch Eingabe von `eksctl create iamidentitymapping --help` in Ihrem Terminal. Falls `ConfigMap` auf Ihrem Server nicht vorhanden ist, wird es von `eksctl` erstellt, wenn Sie den Befehl zum Hinzufügen einer Identitätszuordnung verwenden.

Pod-Anforderungen

- Pod Identity – Mit EKS Pod Identity konfigurierte Pods erhalten Anmeldeinformationen von der EKS-Auth-API. Wenn kein ausgehender Internetzugang vorhanden ist, müssen Sie einen VPC-Endpunkt für die EKS-Auth-API erstellen und verwenden: `com.amazonaws.region-code.eks-auth`. Weitere Informationen zu den EKS- und EKS-Auth-VPC-Endpunkten finden Sie unter the section called “ AWS PrivateLink”
- IRSA — Pods, die mit IAM-Rollen für Dienstkonten konfiguriert sind, beziehen Anmeldeinformationen aus einem API-Aufruf des AWS Security Token Service (AWS STS). Wenn es keinen ausgehenden Internetzugang gibt, müssen Sie einen AWS STS-VPC-Endpunkt in Ihrer VPC erstellen und verwenden. Die meisten AWS `v1` SDKs verwenden standardmäßig den globalen AWS STS-Endpunkt (`sts.amazonaws.com`), der den AWS STS-VPC-Endpunkt nicht verwendet. Um den AWS STS-VPC-Endpunkt zu verwenden, müssen Sie Ihr SDK möglicherweise so konfigurieren, dass es den regionalen AWS STS-Endpunkt (`sts.region-code.amazonaws.com`) verwendet. Weitere Informationen finden Sie unter the section called “STS-Endpunkte”.
- Die VPC-Subnetze Ihres Clusters müssen über einen VPC-Schnittstellenendpunkt für alle AWS Dienste verfügen, auf die Ihre Pods Zugriff benötigen. Weitere Informationen finden Sie unter Zugreifen auf einen AWS Dienst über einen Schnittstellen-VPC-Endpunkt. Einige häufig verwendete Services und Endpunkte sind in der folgenden Tabelle aufgeführt. eine vollständige Liste der Endpunkte finden Sie unter AWS -Services, die mit AWS integriert sindPrivateLink im AWS PrivateLink -Handbuch.

  Wir empfehlen Ihnen, private DNS-Namen für Ihre VPC-Endpunkte zu aktivieren, damit Workloads weiterhin ohne Probleme öffentliche AWS Dienstendpunkte verwenden können.

Service	Endpoint
Amazon EC2	com.amazonaws. *region-code* .ec2

Service	Endpoint
Amazon Elastic Container Registry (zum Abrufen von Container-Images)	com.amazonaws. *region-code* .ecr.api, com.amazonaws. *region-code* .ecr.dkr und com.amazonaws. *region-code* .s3
Amazon Application Load Balancers und Network Load Balancers	com.amazonaws. *region-code* .elastischer Lastenausgleich
(Optional) AWS X-Ray (erforderlich für die an AWS X-Ray gesendete Rückverfolgung)	com.amazonaws. *region-code* .xray
(Optional) Amazon SSM (erforderlich für den SSM-Agenten für Knoten-Verwaltungsaufgaben. Alternative zu SSH)	com.amazonaws. *region-code* .ssm
Amazon CloudWatch Logs (erforderlich für Knoten- und Pod-Logs, die an Amazon CloudWatch Logs gesendet werden)	com.amazonaws. *region-code* .protokolle
AWS Security Token Service (erforderlich, wenn IAM-Rollen für Dienstkonten verwendet werden)	com.amazonaws. *region-code* .sts
Amazon-EKS-Authentifizierung (erforderlich bei Verwendung von Pod-Identity-Zuordnungen)	com.amazonaws. *region-code* .eks-auth
Amazon EKS	com.amazonaws. *region-code* .eks

- Alle selbstverwalteten Knoten müssen in Subnetzen bereitgestellt werden, die über die von Ihnen benötigten VPC-Schnittstellenendpunkte verfügen. Wenn Sie eine verwaltete Knotengruppe erstellen, muss die Endpunktsicherheitsgruppe der VPC-Schnittstelle das CIDR für die Subnetze zulassen, oder Sie müssen die erstellte Knotensicherheitsgruppe zur Endpunktsicherheitsgruppe der VPC-Schnittstelle hinzufügen.
- EFS-Speicher — Wenn Ihre Pods Amazon EFS-Volumes verwenden, muss vor der Bereitstellung von Store an elastic file system with Amazon EFS die Datei kustomization.yaml des Treibers so geändert werden, dass die Container-Images dieselbe AWS Region wie der Amazon EKS-Cluster verwenden.

- Route53 unterstützt nicht. AWS PrivateLink Es ist nicht möglich, Route53-DNS-Einträge von einem privaten Amazon-EKS-Cluster aus zu verwalten. Dies hat Auswirkungen auf Kubernetes external-dns.
- Bei Verwendung der EKS-optimierten AMI sollten Sie den `ec2`-Endpunkt in der obigen Tabelle aktivieren. Alternativ können Sie den Knoten-DNS-Namen manuell festlegen. Das optimierte AMI verwendet EC2 APIs , um den DNS-Namen des Knotens automatisch festzulegen.
- Sie können den Load AWS Balancer Controller verwenden, um AWS Application Load Balancers (ALB) und Network Load Balancers in Ihrem privaten Cluster bereitzustellen. Beim Bereitstellen sollten Sie Befehlszeilen-Flags verwenden, um `enable-shield`, `enable-waf` und `enable-wafv2` auf falsch zu setzen. Die Zertifikatserkennung mit Hostnamen aus Eingangsobjekten wird nicht unterstützt. Dies liegt daran, dass der Controller AWS Certificate Manager erreichen muss, der keinen VPC-Schnittstellenendpunkt hat.

  Der Controller unterstützt Network Load Balancer mit IP-Zielen, die für die Verwendung mit Fargate erforderlich sind. Weitere Informationen erhalten Sie unter the section called “Application Load Balancing” und the section called “Erstellen eines Network Load Balancers”.
- Cluster Autoscaler wird unterstützt. Stellen Sie beim Bereitstellen von Cluster-Autoscaler-Pods sicher, dass die Befehlszeile `--aws-use-static-instance-list=true` enthält. Weitere Informationen finden Sie unter Statische Instanzliste verwenden auf GitHub. Die Worker-Node-VPC muss auch den AWS STS-VPC-Endpunkt und den VPC-Endpunkt für Autoscaling enthalten.
- Einige Container-Softwareprodukte verwenden API-Aufrufe, die auf den AWS Marketplace Metering Service zugreifen, um die Nutzung zu überwachen. Private Cluster lassen diese Aufrufe nicht zu, daher können Sie diese Containertypen nicht in privaten Clustern verwenden.

Skalierung von Cluster-Computing mit Karpenter und Cluster Autoscaler

Auto Scaling ist eine Funktion, die Ihre Ressourcen automatisch nach oben oder unten skaliert, um wechselnden Anforderungen gerecht zu werden. Dies ist eine wichtige Kubernetes-Funktion, die ansonsten umfangreiche Personalressourcen erfordern würde, um manuell zu arbeiten.

EKS Auto Mode

Amazon EKS Auto Mode skaliert die Rechenressourcen des Clusters automatisch. Wenn ein Pod nicht auf die vorhandenen Knoten passt, erstellt EKS Auto Mode einen neuen. EKS Auto Mode konsolidiert außerdem Workloads und löscht Knoten. EKS Auto Mode baut auf Karpenter auf.

Weitere Informationen finden Sie unter:

- EKS Auto Mode
- the section called “Knoten-Pool erstellen”
- the section called “Erhöhte Workload bereitstellen”

Weitere Lösungen

Amazon EKS unterstützt zwei zusätzliche Auto-Scaling-Produkte:

Karpenter

Karpenter ist ein flexibler, leistungsstarker Kubernetes-Cluster-Autoscaler, der zur Verbesserung der Anwendungsverfügbarkeit und Cluster-Effizienz beiträgt. Karpenter startet Rechenressourcen in der richtigen Größe (z. B. Amazon-EC2-Instances) als Reaktion auf sich ändernde Anwendungslasten in weniger als einer Minute. Durch die Integration von Kubernetes mit AWS kann Karpenter Just-in-Time-Rechenressourcen bereitstellen, die genau den Anforderungen Ihres Workloads entsprechen. Karpenter stellt automatisch neue Rechenressourcen basierend auf den spezifischen Anforderungen von Cluster-Workloads bereit. Dazu gehören Rechen-, Speicher-, Beschleunigungs- und Planungsanforderungen. Amazon EKS unterstützt Cluster, die Karpenter verwenden, obwohl Karpenter mit jedem konformierten Kubernetes-Cluster zusammenarbeitet. Weitere Informationen finden Sie in der Karpenter-Dokumentation.

> **Important**
>
> Karpenter ist eine Open-Source-Software, für deren Installation, Konfiguration und Verwaltung in ihren Kubernetes-Clustern die AWS-Kunden verantwortlich sind. AWS bietet technischen Support, wenn Karpenter unverändert mit einer kompatiblen Version in Amazon-EKS-Clustern ausgeführt wird. Es ist wichtig, dass Kunden die Verfügbarkeit und Sicherheit des Karpenter-Controllers sowie entsprechende Testverfahren aufrechterhalten, wenn sie ihn oder den Kubernetes-Cluster, in dem er ausgeführt wird, aktualisieren, genau wie bei jeder anderen vom Kunden verwalteten Software. Für Karpenter besteht kein AWS Service Level Agreement (SLA). Die Kunden sind dafür verantwortlich, sicherzustellen, dass die von Karpenter gestarteten EC2-Instances ihren geschäftlichen Anforderungen entsprechen.

Cluster Autoscaler

Der Kubernetes Cluster Autoscaler passt die Anzahl der Knoten in Ihrem Cluster automatisch an, wenn Pods ausfallen oder auf andere Knoten umgeplant werden. Der Cluster Autoscaler verwendet Auto-Scaling-Gruppen. Weitere Informationen finden Sie unter Cluster Autoscaler on AWS.

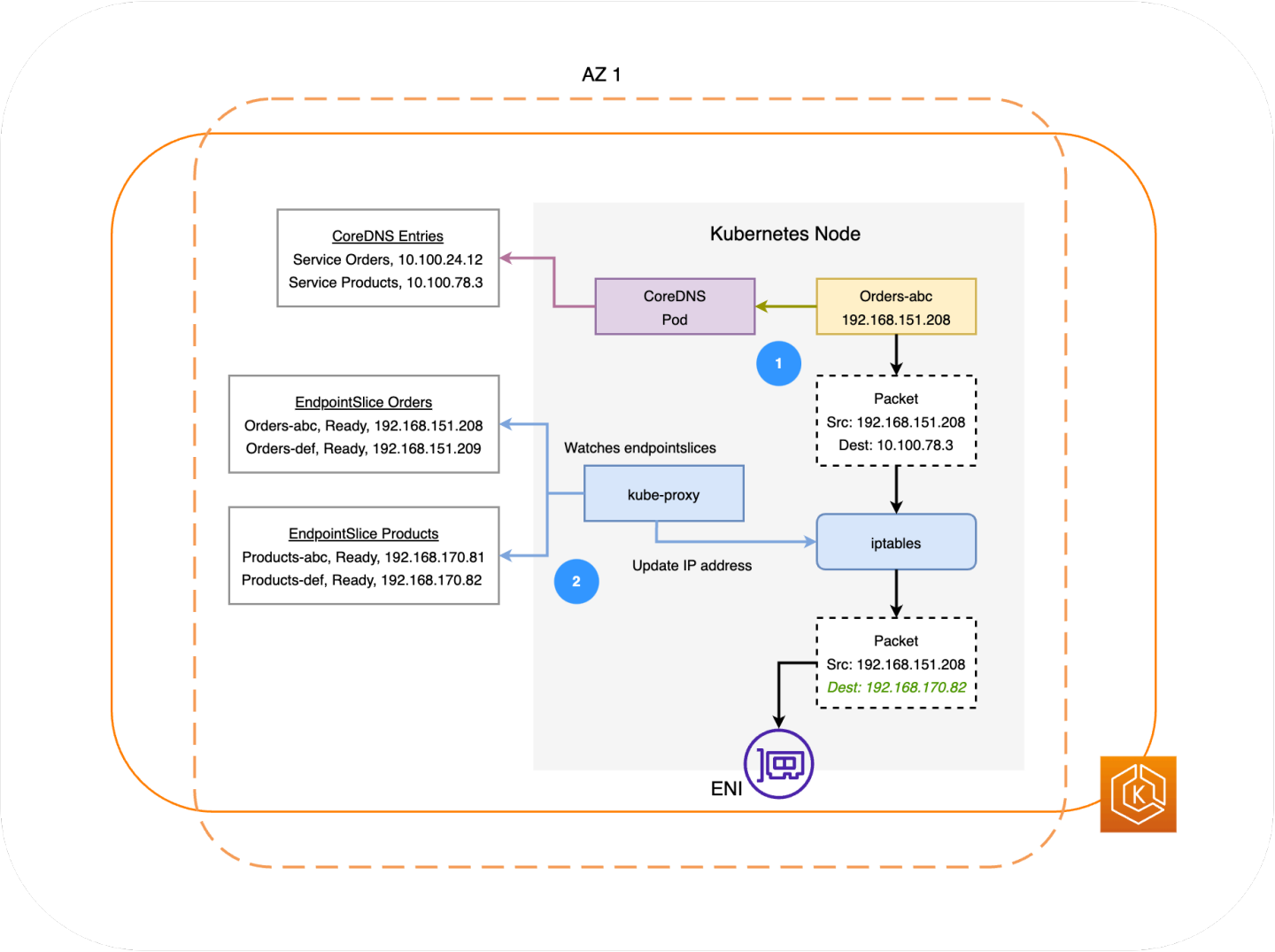
Weitere Informationen zur Zonenverschiebung von Amazon Application Recovery Controller (ARC) in Amazon EKS

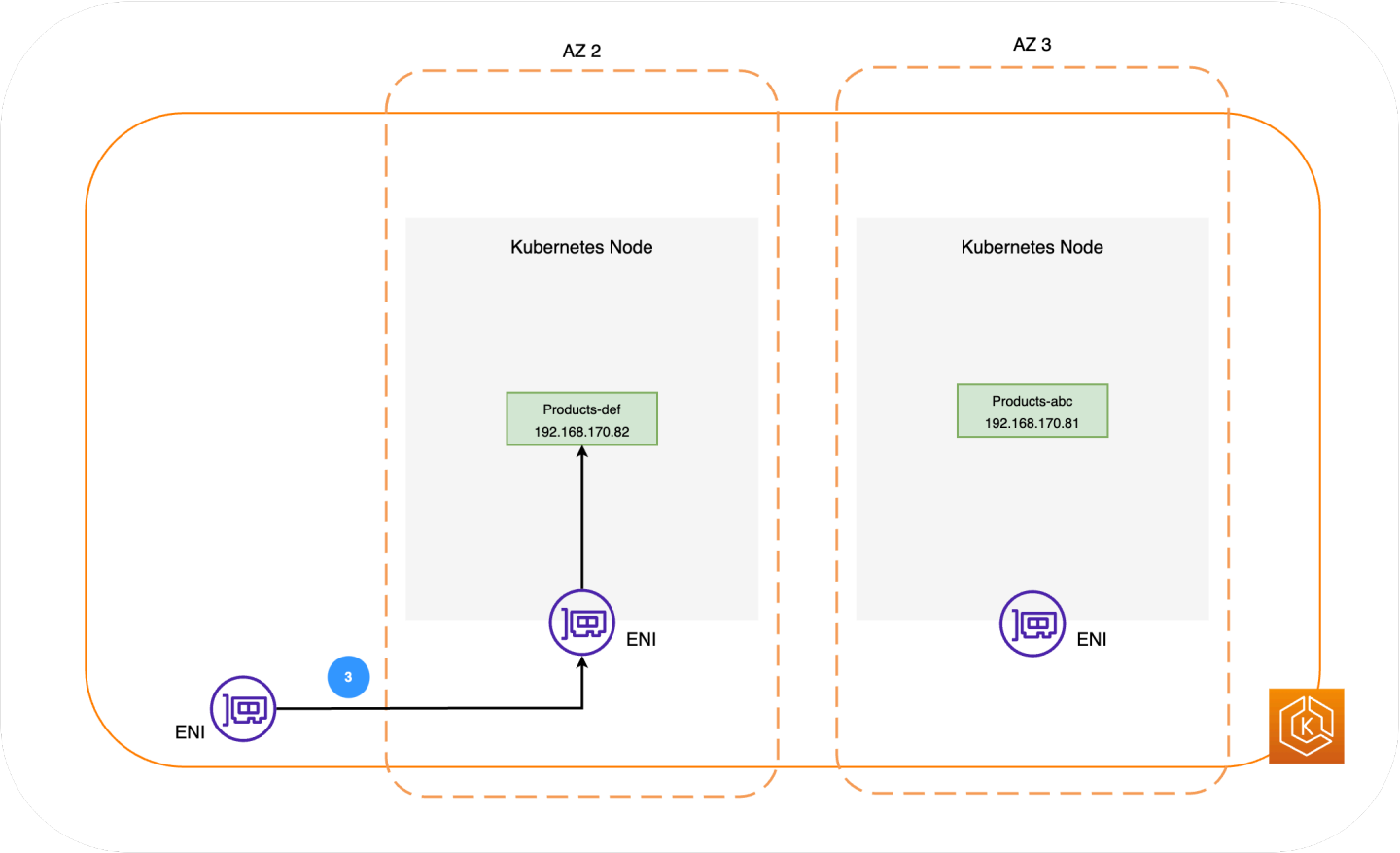
Kubernetes verfügt über native Features, mit denen Sie Ihre Anwendungen stabiler gegenüber Ereignissen wie einer verschlechterten Verfügbarkeit oder Beeinträchtigung einer Availability Zone (AZ) machen können. Wenn Sie Ihre Workloads in einem Amazon-EKS-Cluster ausführen, können Sie die Fehlertoleranz und Anwendungswiederherstellung Ihrer Anwendungsumgebung weiter verbessern, indem Sie Amazon Application Recovery Controller (ARC)-Zonenverschiebung oder automatische Zonenverschiebung einsetzen. ARC-Zonenverschiebung ist als vorübergehende Maßnahme konzipiert, mit der Sie den Datenverkehr für eine Ressource aus einer beeinträchtigten AZ verlagern können, bis die Zonenverschiebung abläuft oder Sie sie abbrechen. Bei Bedarf können Sie die Zonenverschiebung verlängern.

Sie können eine Zonenverschiebung für einen EKS-Cluster starten oder Sie können zulassen AWS , dass der Datenverkehr für Sie verschoben wird, indem Sie Zonal Autoshift aktivieren. Durch diese Verschiebung wird der east-to-west Netzwerkdatenverkehr in Ihrem Cluster so aktualisiert, dass nur Netzwerkendpunkte für Pods berücksichtigt werden, die auf Worker-Knoten ausgeführt werden, die fehlerfrei sind. AZs Darüber hinaus leitet jeder ALB oder NLB, der den eingehenden Datenverkehr für Anwendungen in Ihrem EKS-Cluster verarbeitet, den Datenverkehr automatisch an fehlerfreie Ziele weiter. AZs Für Kunden, die höchste Verfügbarkeit anstreben, kann es im Falle einer Beeinträchtigung einer AZ wichtig sein, den gesamten Datenverkehr von der beeinträchtigten AZ wegleiten zu können, bis diese wiederhergestellt ist. Hierzu können Sie auch einen ALB oder NLB mit ARC-Zonenverschiebung aktivieren.

Verständnis des Ost-West-Netzwerkdatenverkehrs zwischen Pods

Das folgende Diagramm veranschaulicht zwei Beispiel-Workloads: Bestellungen und Produkte. Der Zweck dieses Beispiels besteht darin, zu zeigen, wie Workloads und Pods in verschiedenen Umgebungen miteinander kommunizieren. AZs


AZ 1
CoreDNS Entries
Service Orders, 10.100.24.12
Service Products, 10.100.78.3
Kubernetes Node
CoreDNS
Pod
Orders-abc
192.168.151.208
1
Packet
Src: 192.168.151.208
Dest: 10.100.78.3
EndpointSlice Orders
Orders-abc, Ready, 192.168.151.208
Orders-def, Ready, 192.168.151.209
Watches endpointslices
kube-proxy
iptables
EndpointSlice Products
Products-abc, Ready, 192.168.170.81
Products-def, Ready, 192.168.170.82
2
Update IP address
Packet
Src: 192.168.151.208
Dest: 192.168.170.82
ENI



1. Damit Aufträge mit Produkten kommunizieren können, müssen sie zunächst den DNS-Namen des Ziel-Services auflösen. Aufträge kommunizieren mit CoreDNS, um die virtuelle IP-Adresse (Cluster-IP) für diesen Service abzurufen. Nachdem Aufträge den Produktnamen aufgelöst hat, sendet es Datenverkehr an diese Ziel-IP-Adresse.
2. Der Kube-Proxy läuft auf jedem Knoten im Cluster und sucht kontinuierlich nach Diensten. EndpointSlices Wenn ein Dienst erstellt wird, EndpointSlice wird ein Dienst im Hintergrund vom Controller erstellt und verwaltet. EndpointSlice Jeder EndpointSlice hat eine Liste oder Tabelle mit Endpunkten, die eine Teilmenge von Pod-Adressen sowie die Knoten enthält, auf denen sie ausgeführt werden. Der Kube-Proxy richtet mithilfe von `iptables` auf den Knoten Routing-Regeln für jeden dieser Pod-Endpunkte ein. Der Kube-Proxy ist auch für eine grundlegende Form des Lastausgleichs zuständig, indem er den für die Cluster-IP-Adresse eines Services bestimmten Datenverkehr umleitet, sodass dieser stattdessen direkt an die IP-Adresse eines Pods gesendet wird. Der kube-proxy erreicht dies, indem er die Ziel-IP-Adresse der ausgehenden Verbindung umschreibt.
3. Die Netzwerkpakete werden dann unter Verwendung von ENIs auf den jeweiligen Knoten an den Produkt-Pod in AZ 2 gesendet, wie im vorherigen Diagramm dargestellt.

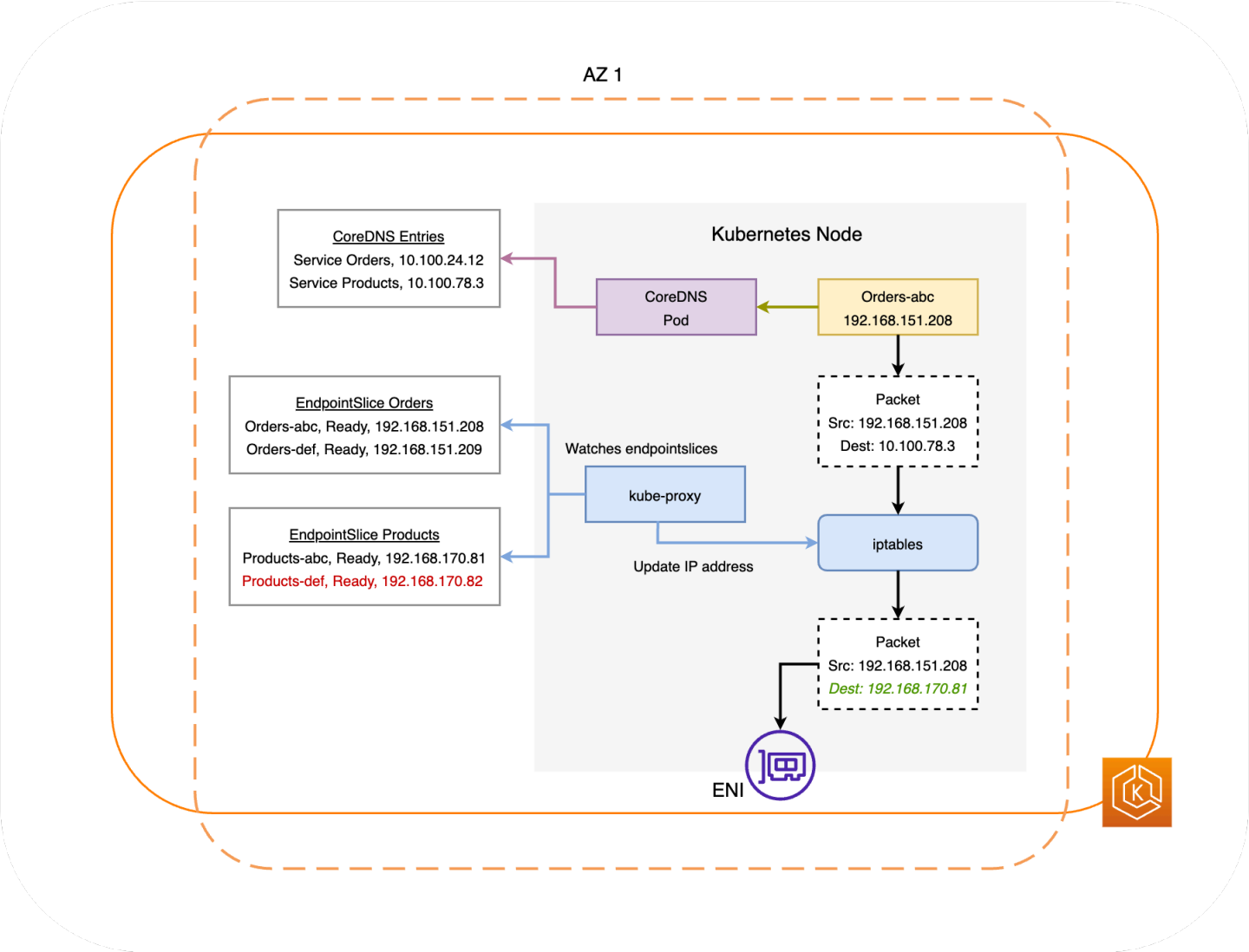
Verständnis der ARC-Zonenverschiebung in Amazon EKS

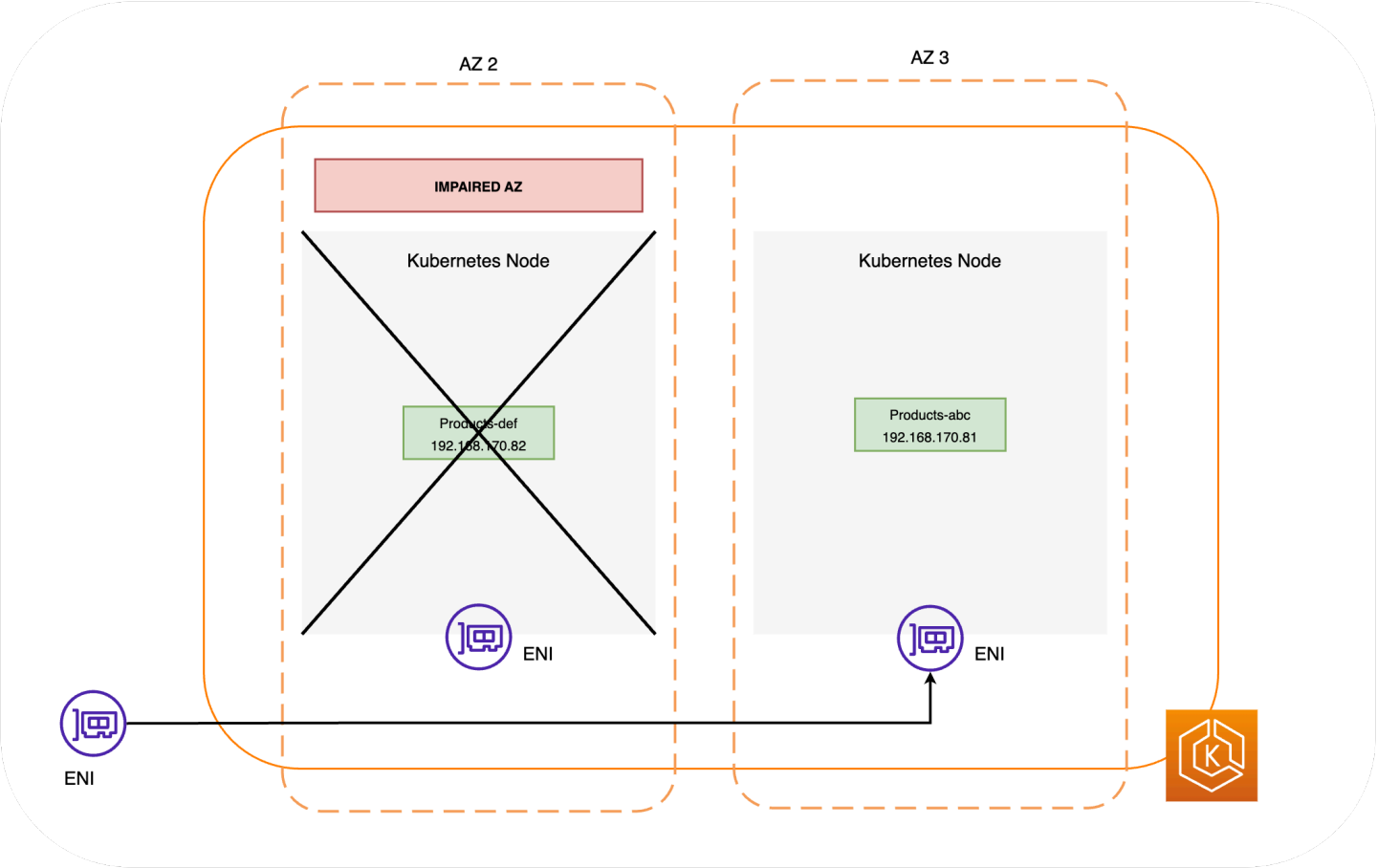
Sollte in Ihrer Umgebung eine AZ-Beeinträchtigung vorliegen, können Sie eine Zonenverschiebung für Ihre EKS-Cluster-Umgebung initiieren. Alternativ können AWS Sie den wechselnden Verkehr mit zonaler Autoshift für Sie verwalten lassen. Mit Zonal Autoshift wird der gesamte Zustand der AZ AWS überwacht und auf eine mögliche Beeinträchtigung der AZ reagiert, indem der Verkehr in Ihrer Cluster-Umgebung automatisch von der beeinträchtigten AZ weggeleitet wird.

Nachdem Ihr Amazon EKS-Cluster Zonal Shift mit ARC aktiviert hat, können Sie eine Zonal Shift starten oder Zonal Autoshift aktivieren, indem Sie die ARC-Konsole, die AWS CLI oder Zonal Shift und Zonal Autoshift verwenden. APIs Während einer EKS-Zonenverschiebung wird Folgendes automatisch durchgeführt:

- Alle Knoten in der betroffenen AZ werden gesperrt. Dadurch wird verhindert, dass der Kubernetes-Scheduler neue Pods auf Knoten in der fehlerhaften AZ plant.
- Wenn Sie Managed Node Groups verwenden, wird das Rebalancing der Availability Zone ausgesetzt und Ihre Auto Scaling Scaling-Gruppe wird aktualisiert, um sicherzustellen, dass neue EKS-Datenebenenknoten nur im AZs fehlerfreien Zustand gestartet werden.
- Die Knoten in der fehlerhaften AZ werden nicht beendet und Pods werden nicht aus den Knoten entfernt. Dadurch wird sichergestellt, dass Ihr Datenverkehr nach Ablauf oder Aufhebung einer Zonenverschiebung sicher in die AZ zurückgeführt werden kann, um die volle Kapazität wiederherzustellen.
- Der EndpointSlice Controller findet alle Pod-Endpunkte in der beeinträchtigten AZ und entfernt sie aus den entsprechenden. EndpointSlices Dadurch wird sichergestellt, dass nur Pod-Endpunkte, die sich in einem AZs fehlerfreien Zustand befinden, gezielt Netzwerkverkehr empfangen. Wenn eine Zonenverschiebung storniert wird oder abläuft, aktualisiert der EndpointSlice Controller sie so, EndpointSlices dass die Endpunkte in die wiederhergestellte AZ aufgenommen werden.

Die folgenden Diagramme bieten einen allgemeinen Überblick darüber, wie die EKS-Zonenverschiebung sicherstellt, dass nur fehlerfreie Pod-Endpunkte in Ihrer Cluster-Umgebung angesteuert werden.


AZ 1
CoreDNS Entries
Service Orders, 10.100.24.12
Service Products, 10.100.78.3
Kubernetes Node
CoreDNS
Pod
Orders-abc
192.168.151.208
Packet
Src: 192.168.151.208
Dest: 10.100.78.3
EndpointSlice Orders
Orders-abc, Ready, 192.168.151.208
Orders-def, Ready, 192.168.151.209
Watches endpointslices
kube-proxy
EndpointSlice Products
Products-abc, Ready, 192.168.170.81
Products-def, Ready, 192.168.170.82
iptables
Update IP address
Packet
Src: 192.168.151.208
Dest: 192.168.170.81
ENI



Anforderungen der EKS-Zonenverschiebung

Damit die Zonenverschiebung mit EKS erfolgreich funktioniert, müssen Sie Ihre Cluster-Umgebung im Voraus so einrichten, dass sie gegenüber einer AZ-Beeinträchtigung widerstandsfähig ist. Nachfolgend finden Sie eine Liste mit Konfigurationsoptionen, die zur Gewährleistung der Ausfallsicherheit beitragen.

- Stellen Sie die Worker-Knoten Ihres Clusters auf mehrere AZs
- Bereitstellung von ausreichend Rechenkapazität, um den Ausfall einer einzelnen AZ zu kompensieren
- Vorskalierung Ihrer Pods, einschließlich CoreDNS, in jeder AZ
- Verteilen Sie mehrere Pod-Repliken auf alle AZs, um sicherzustellen, dass Sie auch bei einem Wechsel von einer einzigen AZ immer noch über ausreichend Kapazität verfügen
- Co-Location von voneinander abhängigen oder verwandten Pods in derselben AZ
- Testen Sie, ob Ihre Cluster-Umgebung ohne eine AZ wie erwartet funktioniert, indem Sie manuell eine Zonenverschiebung weg von einer AZ starten. Alternativ können Sie die automatische

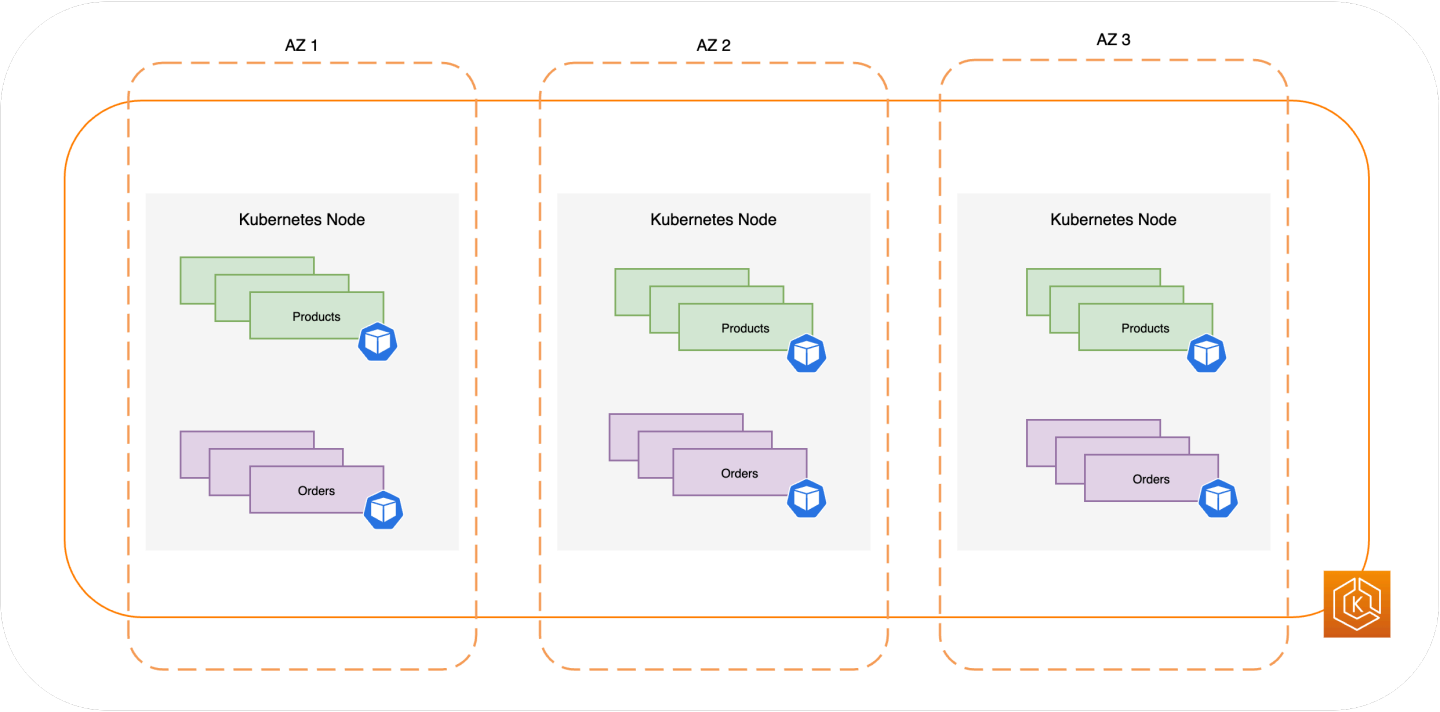
Zonenverschiebung aktivieren und sich auf Testläufe der automatischen Verschiebung verlassen. Tests mit manuellen oder praktischen Zonenverschiebungen sind für die Funktion der Zonenverschiebung in EKS nicht erforderlich, werden aber dringend empfohlen.

Bereitstellung Ihrer EKS-Worker-Knoten über mehrere Availability Zones hinweg

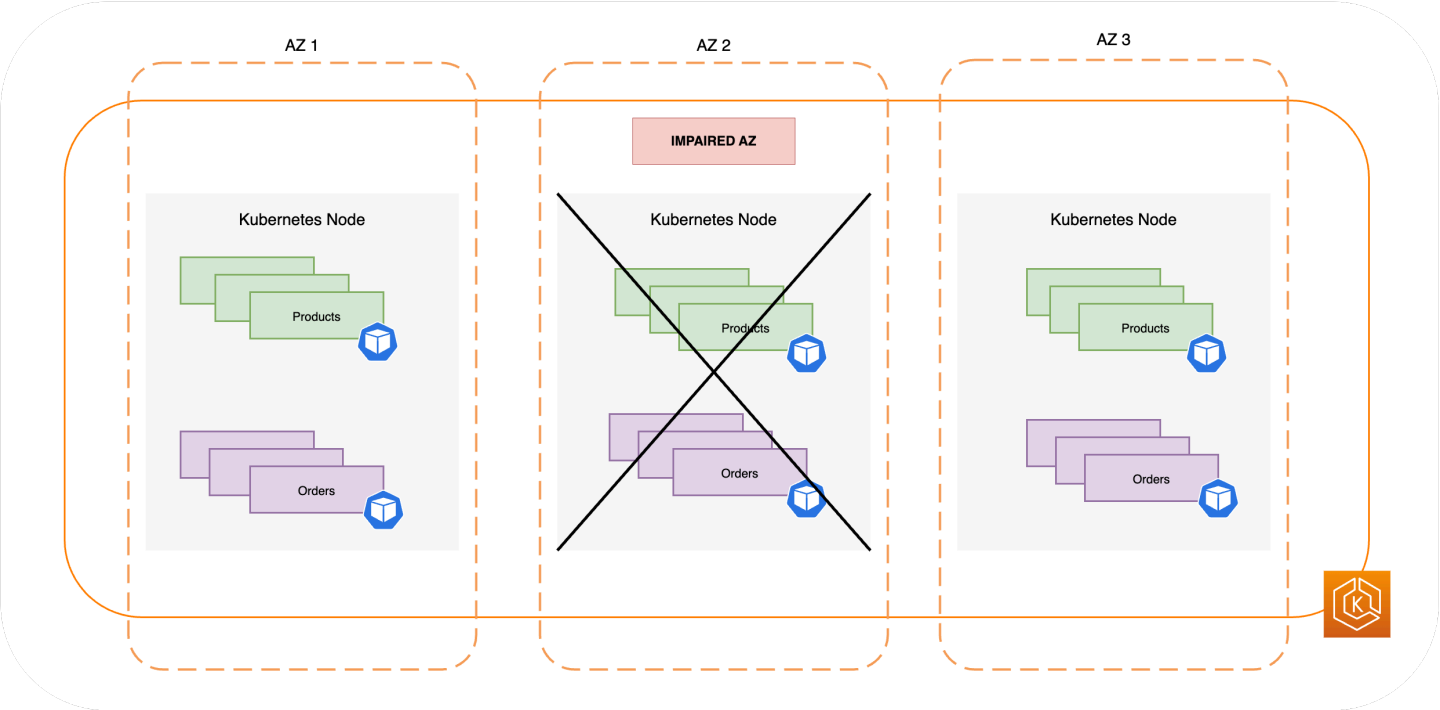
AWS Regionen haben mehrere, separate Standorte mit physischen Rechenzentren, die als Availability Zones (AZs) bezeichnet werden. AZs sind so konzipiert, dass sie physisch voneinander isoliert sind, um gleichzeitige Auswirkungen zu vermeiden, die sich auf eine gesamte Region auswirken könnten. Wenn Sie einen EKS-Cluster bereitstellen, empfehlen wir, dass Sie Ihre Worker-Knoten auf mehreren Knoten AZs in einer Region bereitstellen. Dies trägt dazu bei, dass Ihre Clusterumgebung widerstandsfähiger gegenüber Beeinträchtigungen durch eine einzelne AZ ist, und ermöglicht es Ihnen, die Hochverfügbarkeit Ihrer Anwendungen aufrechtzuerhalten, die in der anderen ausgeführt AZs werden. Wenn Sie mit einer Zonenverlagerung weg von der betroffenen AZ beginnen, wird das Cluster-interne Netzwerk Ihrer EKS-Umgebung automatisch aktualisiert, sodass es nur noch fehlerfrei verwendet wird AZs, um die Hochverfügbarkeit Ihres Clusters aufrechtzuerhalten.

Die Sicherstellung einer Multi-AZ-Konfiguration für Ihre EKS-Umgebung erhöht die allgemeine Zuverlässigkeit Ihres Systems. Multi-AZ-Umgebungen beeinflussen jedoch die Art und Weise, wie Anwendungsdaten übertragen und verarbeitet werden, was sich wiederum auf die Netzwerkgebühren Ihrer Umgebung auswirkt. Insbesondere häufiger ausgehender zonenübergreifender Verkehr (zwischen den einzelnen Zonen verteilter Verkehr AZs) kann erhebliche Auswirkungen auf Ihre Netzwerkkosten haben. Sie können verschiedene Strategien anwenden, um die Menge des zonenübergreifenden Datenverkehrs zwischen Pods in Ihrem EKS-Cluster zu steuern und die damit verbundenen Kosten zu senken. Weitere Informationen zur Optimierung der Netzwerkkosten beim Betrieb hochverfügbarer EKS-Umgebungen finden Sie in diesen Bewährten Methoden.

Das folgende Diagramm zeigt eine hochverfügbare EKS-Umgebung mit drei fehlerfreien Komponenten. AZs



Das folgende Diagramm zeigt, wie eine EKS-Umgebung mit drei AZs EKS-Umgebungen eine Beeinträchtigung der AZ aushält und hochverfügbar bleibt, da zwei davon noch intakt sind. AZs



Stellen Sie ausreichend Rechenkapazität bereit, um den Ausfall einer einzelnen Availability Zone zu kompensieren.

Um die Ressourcennutzung und die Kosten für Ihre Recheninfrastruktur in der EKS-Datenebene zu optimieren, empfiehlt es sich, die Rechenkapazität an Ihren Workload-Anforderungen auszurichten. Wenn jedoch alle Ihre Worker-Knoten voll ausgelastet sind, sind Sie darauf angewiesen, dass neue Worker-Knoten zur EKS-Datenebene hinzugefügt werden, bevor neue Pods geplant werden können. Wenn Sie wichtige Workloads ausführen, empfiehlt es sich im Allgemeinen, redundante Kapazitäten online zu betreiben, um Szenarien wie plötzliche Lastanstiege und Probleme mit der Knotenintegrität zu bewältigen. Wenn Sie eine Zonenverschiebung planen, beabsichtigen Sie, bei einer Wertminderung die gesamte Kapazität einer AZ zu entfernen. Das bedeutet, dass Sie Ihre redundante Rechenkapazität so anpassen müssen, dass sie ausreicht, um die Last auch dann zu bewältigen, wenn eine der Maschinen AZs offline ist.

Wenn Sie Ihre Rechenressourcen skalieren, dauert der Vorgang des Hinzufügens neuer Knoten zur EKS-Datenebene einige Zeit. Dies kann Auswirkungen auf die Echtzeitleistung und Verfügbarkeit Ihrer Anwendungen haben, insbesondere im Falle einer Beeinträchtigung der Zone. Ihre EKS-Umgebung sollte in der Lage sein, den Ausfall einer AZ zu kompensieren, ohne dass dies zu einer Beeinträchtigung der Benutzererfahrung für Ihre Endbenutzer oder Clients führt. Das heißt, die Verzögerung zwischen dem Zeitpunkt, zu dem ein neuer Pod benötigt wird, und dem Zeitpunkt, zu dem er tatsächlich auf einem Worker-Knoten geplant wird, sollte minimiert oder ganz vermieden werden.

Darüber hinaus sollten Sie bei einer Beeinträchtigung der Zonen darauf achten, das Risiko zu minimieren, dass Rechenkapazitätsengpässe auftreten, die verhindern würden, dass neu benötigte Knoten im fehlerfreien Zustand zu Ihrer EKS-Datenebene hinzugefügt werden. AZs

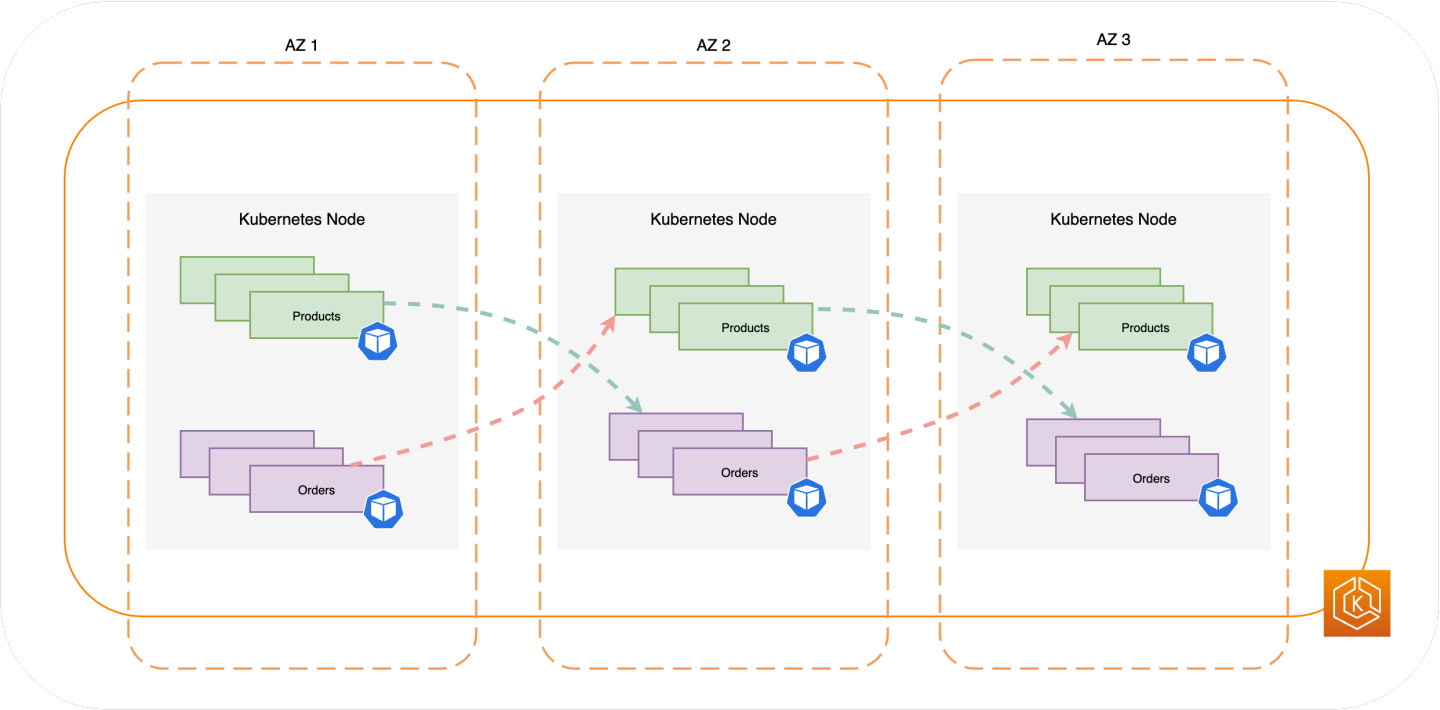
Um das Risiko dieser potenziellen negativen Auswirkungen zu verringern, empfehlen wir, dass Sie in einigen Worker-Knoten in jedem der Workerknoten zu viel Rechenkapazität bereitstellen. AZs Auf diese Weise verfügt der Kubernetes Scheduler über bereits vorhandene Kapazitäten für neue Pod-Platzierungen. Dies ist besonders wichtig, wenn Sie einen Pod in Ihrer Umgebung verlieren. AZs

Ausführung und Verteilung mehrerer Pod-Replikate über Availability Zones hinweg

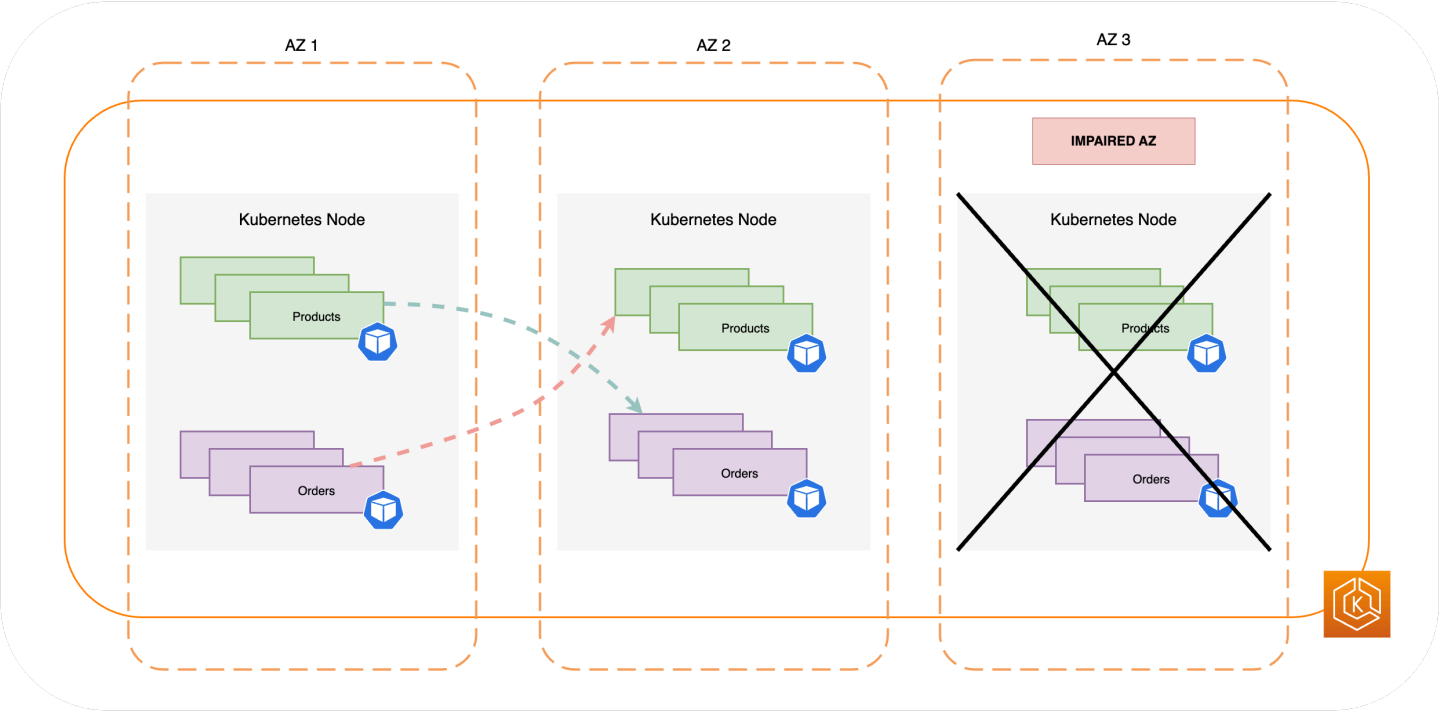
Mit Kubernetes können Sie Ihre Workloads vorab skalieren, indem Sie mehrere Instances (Pod-Replikate) einer einzelnen Anwendung ausführen. Durch die Ausführung mehrerer Pod-Replikate für eine Anwendung werden einzelne Fehlerquellen beseitigt und die Gesamtleistung gesteigert, da die Ressourcenbelastung eines einzelnen Replikats reduziert wird. Um jedoch sowohl eine

hohe Verfügbarkeit als auch eine bessere Fehlertoleranz für Ihre Anwendungen zu erreichen, empfehlen wir Ihnen, mehrere Replikate Ihrer Anwendung auszuführen und die Replikate auf verschiedene Ausfall-Domains (auch Topologie-Domains genannt) zu verteilen. Die Ausfall-Domains in diesem Szenario sind die Availability Zones. Durch die Verwendung von Topologie-Verteilungsbeschränkungen können Sie Ihre Anwendungen so einrichten, dass sie über eine bereits vorhandene, statische Stabilität verfügen. Wenn es dann zu einer Beeinträchtigung der Verfügbarkeit kommt, verfügt Ihre Umgebung über genügend Replikate im fehlerfreien Zustand, um etwaige Datenverkehrsspitzen oder AZs -spitzen sofort zu bewältigen.

Das folgende Diagramm zeigt eine EKS-Umgebung, in der der east-to-west Datenverkehr fließt, wenn alle AZs fehlerfrei sind.



Das folgende Diagramm zeigt eine EKS-Umgebung mit einem east-to-west Verkehrsfluss, bei dem eine einzelne AZ ausgefallen ist und Sie eine Zonenverschiebung gestartet haben.



Der folgende Code-Ausschnitt ist ein Beispiel dafür, wie Sie Ihre Workload mit mehreren Replikaten in Kubernetes einrichten können.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: orders
spec:
  replicas: 9
  selector:
    matchLabels:
      app:orders
  template:
    metadata:
      labels:
        app: orders
        tier: backend
    spec:
      topologySpreadConstraints:
      - maxSkew: 1
        topologyKey: "topology.kubernetes.io/zone"
        whenUnsatisfiable: ScheduleAnyway
        labelSelector:
          matchLabels:
```

```
            app: orders
```

Vor allem sollten Sie mehrere Replikate Ihrer DNS-Server-Software (CoreDNS/kube-dns) ausführen und ähnliche Topologie-Verteilungsbeschränkungen anwenden, sofern diese nicht standardmäßig konfiguriert sind. Auf diese Weise können Sie sicherstellen, dass bei einer einzelnen AZ-Beeinträchtigung genügend DNS-Pods fehlerfrei sind, AZs um weiterhin Service Discovery-Anfragen für andere kommunizierende Pods im Cluster bearbeiten zu können. Das CoreDNS EKS-Add-On verfügt über Standardeinstellungen für die CoreDNS-Pods, die sicherstellen, dass Knoten, die in mehreren AZs verfügbar sind, über die Availability Zones Ihres Clusters verteilt sind. Falls gewünscht, können Sie diese Standardeinstellungen durch Ihre eigenen benutzerdefinierten Konfigurationen ersetzen.

Wenn Sie CoreDNS mit Helm installieren, können Sie `replicaCount` in der Datei values.yaml anpassen, um sicherzustellen, dass Sie über ausreichend Replikate in jeder AZ verfügen. Um sicherzustellen, dass diese Replikate auf die verschiedenen Replikate AZs in Ihrer Clusterumgebung verteilt sind, stellen Sie außerdem sicher, dass Sie die `topologySpreadConstraints` Eigenschaft in derselben Datei aktualisieren. `values.yaml` Der folgende Code-Ausschnitt veranschaulicht, wie Sie CoreDNS entsprechend konfigurieren können.

CoreDNS Helm values.yaml

```
replicaCount: 6
topologySpreadConstraints:
  - maxSkew: 1
    topologyKey: topology.kubernetes.io/zone
    whenUnsatisfiable: ScheduleAnyway
    labelSelector:
      matchLabels:
        k8s-app: kube-dns
```

Bei einer AZ-Beeinträchtigung können Sie die erhöhte Belastung der CoreDNS-Pods durch den Einsatz eines Systems zur automatischen Skalierung für CoreDNS abfedern. Die Anzahl der erforderlichen DNS- Instances hängt von der Anzahl der in Ihrem Cluster ausgeführten Workloads ab. CoreDNS ist CPU-gebunden, wodurch es mithilfe des Horizontal Pod Autoscaler (HPA) basierend auf der CPU skaliert werden kann. Nachfolgend finden Sie ein Beispiel, das Sie an Ihre Anforderungen anpassen können.

```
apiVersion: autoscaling/v1
kind: HorizontalPodAutoscaler
metadata:
```

```
  name: coredns
  namespace: default
spec:
  maxReplicas: 20
  minReplicas: 2
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: coredns
  targetCPUUtilizationPercentage: 50
```

Alternativ kann EKS die automatische Skalierung der CoreDNS-Bereitstellung in der EKS-Add-On-Version von CoreDNS verwalten. Dieser CoreDNS-Autoscaler überwacht kontinuierlich den Status des Clusters, einschließlich der Anzahl der Knoten und CPU-Kerne. Basierend auf diesen Informationen passt der Controller die Anzahl der Replikate der CoreDNS-Bereitstellung in einem EKS-Cluster dynamisch an.

Um die Konfiguration für die automatische Skalierung im CoreDNS-EKS-Add-On zu aktivieren, verwenden Sie die folgende Konfigurationseinstellung:

```
{
  "autoScaling": {
    "enabled": true
  }
}
```

Sie können auch NodeLocal DNS oder den Cluster Proportional Autoscaler verwenden, um CoreDNS zu skalieren. Weitere Informationen finden Sie unter Horizontale Skalierung von CoreDNS.

Co-Location voneinander abhängiger Pods in derselben Availability Zone

In der Regel haben Anwendungen unterschiedliche Workloads, die miteinander kommunizieren müssen, um einen Prozess erfolgreich abzuschließen. end-to-end Wenn diese unterschiedlichen Anwendungen auf verschiedene Bereiche verteilt sind AZs und sich nicht in derselben AZ befinden, kann sich eine einzelne Beeinträchtigung der AZ auf den Prozess auswirken. end-to-end Wenn Anwendung A beispielsweise mehrere Replikate in AZ 1 und AZ 2 hat, Anwendung B jedoch alle Replikate in AZ 3 hat, wirkt sich der Verlust von AZ 3 auf end-to-end Prozesse zwischen den beiden Workloads, Anwendung A und Anwendung B, aus. Wenn Sie Beschränkungen der Topologieverteilung mit Pod-Affinität kombinieren, können Sie die Ausfallsicherheit Ihrer Anwendung verbessern, indem Sie Pods über alle Pods verteilen. AZs Darüber hinaus wird dadurch eine

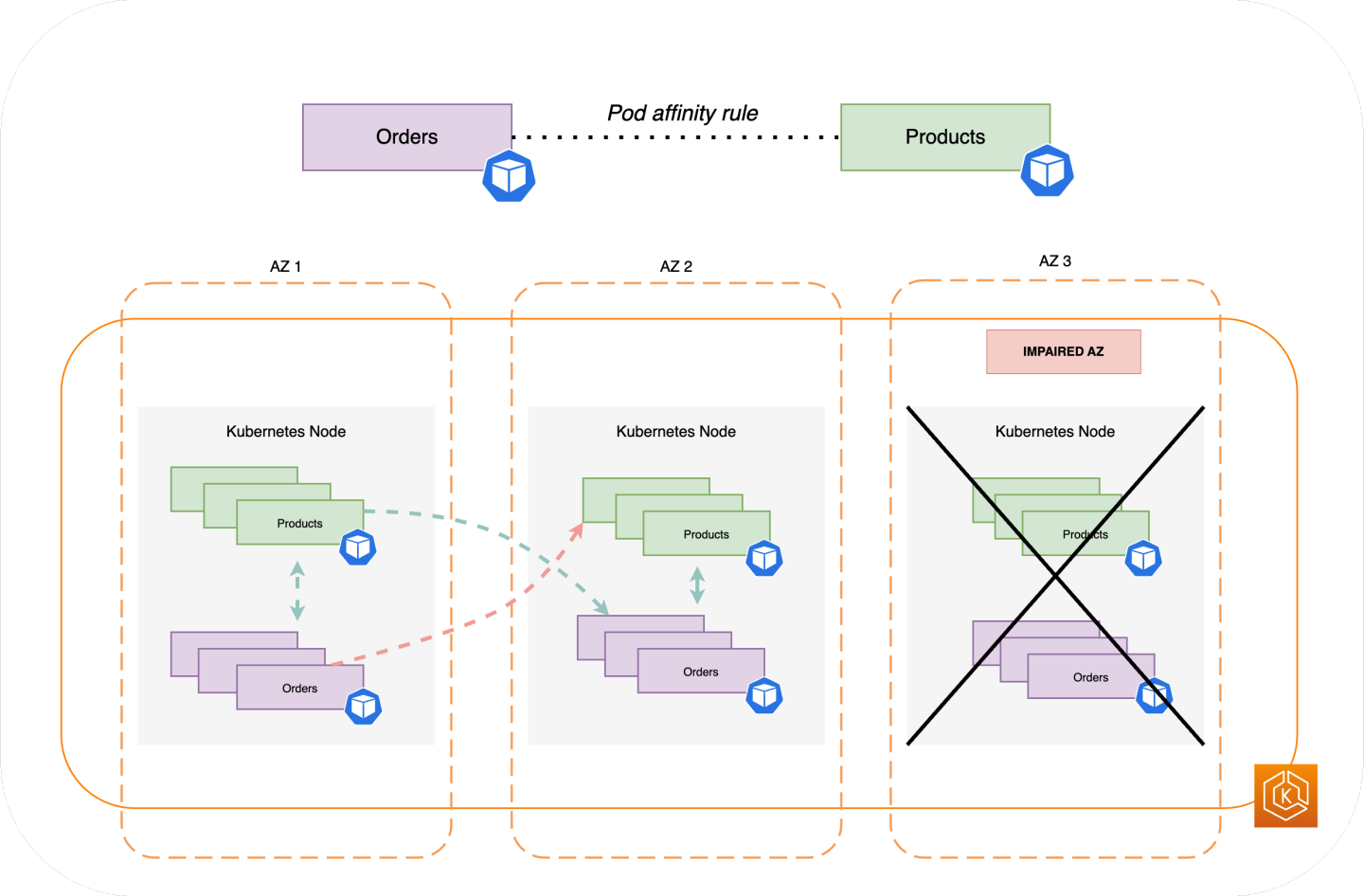
Beziehung zwischen bestimmten Pods konfiguriert, um sicherzustellen, dass sie zusammen platziert sind.

Mit Pod-Affinitätsregeln können Sie Beziehungen zwischen Workloads definieren, um das Verhalten des Kubernetes-Schedulers so zu beeinflussen, dass Pods auf demselben Worker-Knoten oder in derselben AZ zusammengeführt werden. Sie können auch konfigurieren, wie streng die Planungsbeschränkungen sein sollen.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: products
  namespace: ecommerce
  labels:
    app.kubernetes.io/version: "0.1.6"

    spec:
      serviceAccountName: graphql-service-account
      affinity:
        podAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
          - labelSelector:
              matchExpressions:
              - key: app
                operator: In
                values:
                - orders
            topologyKey: "kubernetes.io/hostname"
```

Das folgende Diagramm zeigt mehrere Pods, die mithilfe von Pod-Affinitätsregeln auf demselben Knoten zusammen platziert wurden.



Testen Ihrer Cluster-Umgebung für die Bewältigung eines Ausfalls einer AZ

Nachdem Sie die in den vorherigen Abschnitten beschriebenen Anforderungen erfüllt haben, müssen Sie im nächsten Schritt überprüfen, ob Sie über ausreichende Rechen- und Workload-Kapazitäten verfügen, um den Ausfall einer AZ zu bewältigen. Dazu können Sie manuell eine Zonenverschiebung in EKS starten. Alternativ können Sie die automatische Zonenverschiebung aktivieren und Testläufe konfigurieren, mit denen ebenfalls überprüft wird, ob Ihre Anwendungen mit einer AZ weniger in Ihrer Cluster-Umgebung wie erwartet funktionieren.

Häufig gestellte Fragen

Warum sollte ich dieses Feature nutzen?

Durch die Verwendung von ARC-Zonenverschiebung oder automatischer Zonenverschiebung in Ihrem EKS-Cluster können Sie die Verfügbarkeit von Kubernetes-Anwendungen besser aufrechterhalten. Dazu wird der schnelle Wiederherstellungsprozess automatisiert, bei dem der Netzwerk-Datenverkehr innerhalb des Clusters von einer beeinträchtigten AZ weg verlagert wird.

Mit ARC können Sie langwierige, komplizierte Schritte vermeiden, die bei Störungen in AZs zu einer verlängerten Wiederherstellungszeit führen können

Wie funktioniert diese Funktion mit anderen Diensten? AWS

EKS ist in ARC integriert, das die primäre Schnittstelle für die Durchführung von Wiederherstellungsvorgängen darstellt AWS. Um sicherzustellen, dass der Datenverkehr innerhalb des Clusters ordnungsgemäß von einer beeinträchtigten AZ weggeleitet wird, nimmt EKS Änderungen an der Liste der Netzwerkendpunkte für Pods vor, die in der Kubernetes-Datenebene ausgeführt werden. Wenn Sie Elastic Load Balancing verwenden, um externen Datenverkehr in den Cluster zu leiten, können Sie Ihre Load Balancer bei ARC registrieren und eine Zonenverschiebung auf ihnen starten, um zu verhindern, dass Datenverkehr in die beeinträchtigte AZ gelangt. Zonal Shift funktioniert auch mit Amazon EC2 Auto Scaling Scaling-Gruppen, die von EKS verwalteten Knotengruppen erstellt werden. Um zu verhindern, dass eine beeinträchtigte AZ für neue Kubernetes-Pods oder Knoten-Starts verwendet wird, entfernt EKS die beeinträchtigte AZ aus den Auto-Scaling-Gruppen.

Inwiefern unterscheidet sich dieses Feature von den standardmäßigen Kubernetes-Schutzmaßnahmen?

Dieses Feature arbeitet mit mehreren in Kubernetes integrierten Schutzmaßnahmen zusammen, welche die Ausfallsicherheit von Kundenanwendungen verbessern Sie können Pod-Bereitschafts- und Aktivitätsprüfung konfigurieren, die entscheiden, wann ein Pod Datenverkehr aufnehmen soll. Wenn diese Prüfungen fehlschlagen, entfernt Kubernetes diese Pods als Ziele für einen Service, und es wird kein Datenverkehr mehr an den Pod gesendet. Dies ist zwar nützlich, für Kunden ist es jedoch nicht einfach, diese Integritätsprüfungen so zu konfigurieren, dass sie garantiert fehlschlagen, wenn eine AZ beeinträchtigt wird. Feature für die ARC-Zonenverschiebung bietet ein zusätzliches Sicherheitsnetz, mit dem Sie eine beeinträchtigte AZ vollständig isolieren können, wenn der native Schutz von Kubernetes nicht ausreicht. Durch Zonenverschiebung können Sie außerdem auf einfache Weise die Betriebsbereitschaft und Ausfallsicherheit Ihrer Architektur überprüfen.

Kann ich in meinem Namen eine Zonenverschiebung AWS starten?

Ja, wenn Sie eine vollständig automatisierte Nutzung der ARC-Zonenverschiebung wünschen, können Sie die automatische ARC-Zonenverschiebung aktivieren. Mit Zonal Autoshift können Sie sich AWS darauf verlassen, dass Sie den Zustand Ihres EKS-Clusters überwachen und automatisch eine Zonenverschiebung starten, wenn eine Beeinträchtigung der AZ festgestellt wird. AZs

Was geschieht, wenn ich dieses Feature verwende und meine Worker-Knoten und Workloads nicht vorab skaliert sind?

Wenn Sie nicht vorab skaliert haben und während einer Zonenverschiebung auf die Bereitstellung zusätzlicher Knoten oder Pods angewiesen sind, riskieren Sie eine verzögerte Wiederherstellung. Das Hinzufügen neuer Knoten zur Kubernetes-Datenebene nimmt einige Zeit in Anspruch, was sich auf die Echtzeitleistung und Verfügbarkeit Ihrer Anwendungen auswirken kann, insbesondere bei einer Beeinträchtigung der Zone. Darüber hinaus kann es bei einer Beeinträchtigung der Zonen zu einer potenziellen Einschränkung der Rechenkapazität kommen, die verhindern könnte, dass neu benötigte Knoten zu den fehlerfreien Knoten hinzugefügt werden. AZs

Wenn Ihre Workloads nicht vorab skaliert und auf alle Bereiche Ihres Clusters verteilt sind, kann sich eine zonale Beeinträchtigung auf die Verfügbarkeit einer Anwendung auswirken, die nur auf Worker-Knoten AZs in einer betroffenen AZ ausgeführt wird. Um das Risiko eines vollständigen Ausfalls der Verfügbarkeit Ihrer Anwendung zu mindern, verfügt EKS über eine Ausfallsicherung für den Datenverkehr, der an Pod-Endpunkte in einer beeinträchtigten Zone gesendet wird, wenn alle Endpunkte dieser Workload in der nicht funktionsfähigen AZ liegen. Wir empfehlen jedoch dringend, dass Sie Ihre Anwendungen vorab skalieren und auf alle verteilen, AZs um die Verfügbarkeit im Falle eines zonalen Problems aufrechtzuerhalten.

Wie funktioniert dies, wenn ich eine zustandsbehaftete Anwendung ausführe?

Wenn Sie eine zustandsbehaftete Anwendung ausführen, müssen Sie deren Fehlertoleranz auf der Grundlage Ihres Anwendungsfalls und Ihrer Architektur bewerten. Wenn Sie über eine active/standby Architektur oder ein Muster verfügen, kann es vorkommen, dass sich die aktive Komponente in einer beeinträchtigten AZ befindet. Auf Anwendungsebene kann es zu Problemen mit Ihrer Anwendung kommen, wenn der Standby-Modus nicht aktiviert ist. Möglicherweise treten auch Probleme auf, wenn neue Kubernetes-Pods im fehlerfreien Zustand gestartet werden AZs, da sie sich dann nicht mit den persistenten Volumes verbinden können, die auf die beeinträchtigte AZ beschränkt sind.

Ist dieses Feature mit Karpenter kompatibel?

Derzeit wird Karpenter nicht mit der ARC-Zonenverschiebung und Zonal Autoshift in EKS unterstützt. Wenn eine AZ beeinträchtigt ist, können Sie die entsprechende NodePool Karpenter-Konfiguration anpassen, indem Sie die fehlerhafte AZ entfernen, sodass neue Worker-Knoten nur in der anderen gestartet werden. AZs

Ist dieses Feature mit EKS Fargate kompatibel?

Dieses Feature ist mit EKS Fargate nicht kompatibel. Wenn EKS Fargate ein zonales Gesundheitsereignis erkennt, ziehen es Pods standardmäßig vor, in dem anderen System ausgeführt zu werden. AZs

Wird die von EKS verwaltete Kubernetes- Steuerebene davon betroffen sein?

Nein, Amazon EKS führt standardmäßig die Kubernetes-Steuerebene aus und skaliert sie über mehrere, AZs um eine hohe Verfügbarkeit zu gewährleisten. Die ARC-Zonenverschiebung und die automatische Zonenverschiebung wirken sich ausschließlich auf die Kubernetes-Datenebene aus.

Sind mit diesem neuen Feature Kosten verbunden?

Sie können ARC-Zonenverschiebung und automatische Zonenverschiebung in Ihrem EKS-Cluster ohne zusätzliche Kosten nutzen. Sie müssen jedoch weiterhin für bereitgestellte Instances bezahlen. Wir empfehlen Ihnen dringend, Ihre Kubernetes-Datenebene vor der Nutzung dieses Features vorab zu skalieren. Sie sollten ein Gleichgewicht zwischen Kosten und Anwendungsverfügbarkeit anstreben.

Weitere Ressourcen

- Funktionsweise einer Zonenverschiebung
- Bewährte Methoden für Zonenverschiebungen in ARC
- Unterstützte Ressourcen und Szenarien für Zonenverschiebung und automatische Zonenverschiebung
- Ausführung ausfallsicherer Workloads in Amazon EKS
- Beseitigung von Verzögerungen bei der Skalierung von Kubernetes-Knoten durch Pod-Priorität und Überbereitstellung
- Skalierung von CoreDNS-Pods für hohen DNS-Datenverkehr

Aktivierung der EKS-Zonenverschiebung zur Vermeidung beeinträchtigter Availability Zones

Amazon Application Recovery Controller (ARC) unterstützt Sie bei der Verwaltung und Koordination der Wiederherstellung Ihrer Anwendungen in allen Availability Zones (AZs) und funktioniert mit vielen Services, einschließlich Amazon EKS. Mithilfe des EKS-Supports für ARC-Zonenverschiebung können Sie den Netzwerkverkehr innerhalb des Clusters von einer beeinträchtigten AZ weg verlagern. Sie können auch autorisieren AWS , den Zustand Ihres Netzwerkverkehrs zu überwachen AZs und den Netzwerkverkehr in Ihrem Namen vorübergehend von einer fehlerhaften AZ wegzuverlagern.

Verwendung der EKS-Zonenverschiebung:

1. Aktivieren Sie Ihren EKS-Cluster mit Amazon Application Recovery Controller (ARC). Dies erfolgt auf Cluster-Ebene mithilfe der Amazon EKS-Konsole, der AWS CLI oder eksctl. CloudFormation
2. Nach der Aktivierung können Sie Zonal Shifts oder Zonal Autoshift mit der ARC-Konsole, der AWS CLI oder Zonal Shift und Zonal Autoshift verwalten. APIs

Beachten Sie, dass Sie nach der Registrierung eines EKS-Clusters bei ARC noch ARC konfigurieren müssen. Sie können beispielsweise die ARC-Konsole verwenden, um die automatische Zonenverschiebung zu konfigurieren.

Ausführlichere Informationen zur Funktionsweise der EKS-Zonenverschiebung und zur Gestaltung Ihrer Workloads für den Umgang mit beeinträchtigten Availability Zones finden Sie unter the section called “Weitere Informationen zur Zonenverschiebung”.

Überlegungen

- EKS Auto Mode unterstützt weder den Amazon Application Recovery Controller noch Zonenverschiebung und automatische Zonenverschiebung.
- Wir empfehlen, zwischen den Vorgängen von Zonenverschiebungen mindestens 60 Sekunden zu warten, um eine ordnungsgemäße Verarbeitung jeder Anfrage sicherzustellen.

  Bei dem Versuch, Zonenverschiebungen in schneller Folge (innerhalb von 60 Sekunden) durchzuführen, kann es vorkommen, dass der Amazon-EKS-Service nicht alle Verschiebungsanforderungen ordnungsgemäß verarbeitet. Dies ist auf den aktuellen Abfragemechanismus zurückzuführen, der den Zonen-Status des Clusters aktualisiert. Wenn Sie mehrere Zonenverschiebungen durchführen müssen, stellen Sie sicher, dass zwischen den Vorgängen ausreichend Zeit liegt, damit das System jede Änderung verarbeiten kann.

Was ist Amazon Application Recovery Controller?

Der Amazon Application Recovery Controller (ARC) unterstützt Sie bei der Vorbereitung und Durchführung einer schnelleren Wiederherstellung für Anwendungen, die in AWS ausgeführt werden. Zonal Shift ermöglicht es Ihnen, sich schnell von Beeinträchtigungen der Availability Zone (AZ) zu erholen, indem Sie den Verkehr für eine unterstützte Ressource vorübergehend von einer AZ auf eine gesunde in der Region verlagern. AZs AWS

Weitere Informationen zu Amazon Application Recovery Controller (ARC)

Was ist eine Zonenverschiebung?

Zonal Shift ist eine Funktion in ARC, mit der Sie den Traffic für eine Ressource wie einen EKS-Cluster oder einen Elastic Load Balancer von einer Availability Zone in einer AWS Region wegverlagern können, um ein Problem schnell zu beheben und Ihre Anwendung schnell wiederherzustellen. Sie können sich beispielsweise für eine Verlagerung des Datenverkehrs entscheiden, weil eine fehlerhafte Bereitstellung zu Latenzproblemen führt oder weil die Availability Zone beeinträchtigt ist. Eine Zonenverschiebung erfordert keine vorherigen Konfigurationsschritte.

Weitere Informationen zur ARC-Zonenverschiebung

Was ist die automatische Zonenverschiebung?

Zonal Autoshift ist eine Funktion in ARC, die Sie aktivieren können, um zu autorisieren AWS , dass der Datenverkehr in Ihrem Namen von einer AZ für unterstützte Ressourcen auf gesunde Ressourcen in der Region umgeleitet wird. AZs AWS AWS startet eine automatische Verschiebung, wenn die interne Telemetrie anzeigt, dass in einer AZ in einer Region eine Beeinträchtigung vorliegt, die sich möglicherweise auf Kunden auswirken könnte. Die interne Telemetrie beinhaltet Metriken aus verschiedenen Quellen, darunter dem AWS Netzwerk und den Amazon- EC2 und Elastic Load Balancing Balancing-Diensten.

AWS beendet Autoshifts, wenn Indikatoren anzeigen, dass kein Problem oder potenzielles Problem mehr besteht.

Weitere Informationen zur automatischen Zonenverschiebung von ARC

Was unternimmt EKS während einer automatischen Verschiebung?

EKS aktualisiert die Netzwerkkonfigurationen, um zu verhindern, dass der Datenverkehr an beeinträchtigte Personen weitergeleitet wird. AZs Wenn Sie verwaltete Knotengruppen verwenden, startet EKS außerdem nur AZs während einer Zonenverschiebung neue Knoten im fehlerfreien Zustand. AZs. Wenn die Verschiebung abläuft oder abgebrochen wird, werden die Netzwerkkonfigurationen wiederhergestellt, sodass die zuvor als nicht funktionsfähig erkannte AZ wieder einbezogen wird.

Weitere Informationen zur EKS-Zonenverschiebung

Registrieren Sie den EKS-Cluster beim Amazon Application Recovery Controller (ARC) (AWS Konsole)

1. Suchen Sie den Namen und die Region des EKS-Clusters, den Sie bei ARC registrieren möchten.
2. Navigieren Sie zur EKS-Konsole in dieser Region und wählen Sie Ihren Cluster aus.
3. Wählen Sie auf der Seite Cluster-Informationen die Registerkarte Übersicht aus.
4. Klicken Sie unter der Überschrift Zonenverschiebung auf die Schaltfläche Verwalten.
5. Aktivieren oder deaktivieren Sie die EKS-Zonenverschiebung.

Ihr EKS-Cluster ist nun bei ARC registriert.

Wenn Sie beeinträchtigte Availability Zones erkennen und vermeiden möchten AWS , müssen Sie ARC Zonal Autoshift konfigurieren. Sie können dies beispielsweise in der ARC-Konsole tun.

Nächste Schritte

- Erfahren Sie, wie Sie die automatische Zonenverschiebung aktivieren.
- Erfahren Sie, wie Sie eine automatische Zonenverschiebung manuell starten.

Erfahren Sie, wie die Zugriffskontrolle in Amazon EKS funktioniert

Erfahren Sie, wie Sie den Zugriff auf Ihren Amazon-EKS-Cluster verwalten. Für die Verwendung von Amazon EKS sind Kenntnisse über die Zugriffskontrolle in Kubernetes und AWS Identity and Access Management (AWS IAM) erforderlich.

Dieser Abschnitt enthält:

the section called “Kubernetes-API-Zugriff” – Erfahren Sie, wie Sie Anwendungen oder Benutzern die Authentifizierung bei der Kubernetes-API ermöglichen. Sie können Zugriffseinträge, die aws-auth-ConfigMap oder einen externen OIDC-Anbieter verwenden

the section called “Zugriff auf Cluster-Ressourcen” – Erfahren Sie, wie Sie AWS-Managementkonsole für die Kommunikation mit Ihrem Amazon-EKS-Cluster konfigurieren. Verwenden Sie die Konsole, um Kubernetes-Ressourcen im Cluster anzuzeigen, z. B. Namespaces, Knoten und Pods.

the section called “Zugriff auf den Cluster mit kubectl” – Erfahren Sie, wie Sie kubectl für die Kommunikation mit Ihrem Amazon-EKS-Cluster konfigurieren. Verwenden Sie die AWS-CLI, um eine kubeconfig-Datei zu erstellen.

the section called “Workload-Zugriff auf AWS ” – Erfahren Sie, wie Sie ein Kubernetes-Servicekonto einer AWS-IAM-Rolle zuordnen. Sie können Pod Identity oder IAM Roles for Service Accounts (IRSA) verwenden.

Allgemeine Aufgaben

- Gewähren Sie Entwicklern Zugriff auf die Kubernetes-API. Zeigen Sie Kubernetes-Ressourcen im AWS-Managementkonsole an.
  - Lösung: Verwenden Sie Zugriffseinträge, um Kubernetes-RBAC-Berechtigungen AWS-IAM-Benutzern oder -Rollen zuzuordnen.
- Konfigurieren Sie kubectl, um mithilfe von AWS-Anmeldeinformationen mit einem Amazon-EKS-Cluster zu kommunizieren.
  - Lösung: Verwenden Sie die AWS-CLI, um eine kubeconfig-Datei zu erstellen.
- Verwenden Sie einen externen Identitätsanbieter wie Ping Identity, um Benutzer bei der Kubernetes-API zu authentifizieren.

  - Lösung: Verbinden Sie einen externen OIDC-Anbieter.
- Gewähren Sie Workloads in Ihrem Kubernetes-Cluster die Berechtigung, AWS-APIs aufzurufen.
  - Lösung: Verwenden Sie Pod Identity, um eine AWS-IAM-Rolle einem Kubernetes-Servicekonto zuzuordnen.

Hintergrund

- Erfahren Sie, wie Kubernetes-Servicekonten funktionieren.
- Überprüfen Sie das rollenbasierte Zugriffskontrollmodell (RBAC) von Kubernetes.
- Weitere Informationen zu AWS-Ressourcen finden Sie im AWS-IAM-Benutzerhandbuch. Alternativ können Sie an einem kostenlosen Einführungs-Training zur Verwendung von AWS IAM teilnehmen.

Überlegungen zu EKS Auto Mode

EKS Auto Mode lässt sich mit EKS Pod Identity und EKS-Zugriffseinträgen integrieren.

- EKS Auto Mode verwendet Zugriffseinträge, um der EKS-Steuerebene Kubernetes-Berechtigungen zu gewähren. Beispielsweise ermöglichen die Zugriffsrichtlinien EKS Auto Mode, Informationen über Netzwerkendpunkte und -services zu lesen
  - Es ist nicht möglich, Zugriffseinträge in einem EKS-Auto-Mode-Cluster zu deaktivieren.
  - Optional können Sie die `aws-auth ConfigMap` aktivieren.
  - Die Zugriffseinträge für EKS Auto Mode werden automatisch konfiguriert. Sie können diese Zugriffseinträge anzeigen, jedoch nicht ändern.
  - Wenn Sie eine NodeClass verwenden, um eine benutzerdefinierte Node-IAM-Rolle zu erstellen, müssen Sie einen Zugriffseintrag für die Rolle mithilfe der Zugriffsrichtlinie AmazonEKSAutoNodePolicy erstellen.
- Wenn Sie Workloads Berechtigungen für AWS-Services gewähren möchten, verwenden Sie EKS Pod Identity.
  - Sie müssen den Pod-Identity-Agenten nicht auf EKS-Auto-Mode-Clustern installieren.

Gewähren Sie IAM-Benutzern und -Rollen Zugriff auf Kubernetes APIs

Ihr Cluster verfügt über einen Kubernetes-API-Endpunkt. Kubectl verwendet diese API. Sie können sich mit zwei Identitätstypen bei dieser API authentifizieren:

- Ein AWS Identity and Access Management (IAM) -Prinzipal (Rolle oder Benutzer) — Dieser Typ erfordert eine Authentifizierung bei IAM. Benutzer können sich AWS als IAM-Benutzer oder mit einer föderierten Identität anmelden, indem sie Anmeldeinformationen verwenden, die über eine Identitätsquelle bereitgestellt wurden. Benutzer können sich nur dann mit einer Verbundidentität anmelden, wenn Ihr Administrator zuvor mithilfe von IAM-Rollen einen Identitätsverbund eingerichtet hat. Wenn Benutzer AWS mithilfe eines Verbunds darauf zugreifen, übernehmen sie indirekt eine Rolle. Wenn Benutzer diese Art von Identität verwenden, gilt Folgendes:
  - Kann ihnen Kubernetes-Berechtigungen zuweisen, damit sie mit Kubernetes-Objekten in Ihrem Cluster arbeiten können. Weitere Informationen dazu, wie Sie Ihren IAM-Prinzipalen Berechtigungen zuweisen, damit sie auf Kubernetes-Objekte in Ihrem Cluster zugreifen können, finden Sie unter the section called “Zugriffseinträge”.
  - Kann ihnen IAM-Berechtigungen zuweisen, sodass sie mithilfe der Amazon EKS-API, AWS CLI,, oder `eksctl` mit Ihrem Amazon EKS-Cluster und seinen Ressourcen arbeiten können. AWS CloudFormation AWS-Managementkonsole Weitere Informationen finden Sie in der Service-Authorization-Referenz unter Von Amazon Elastic Kubernetes Service definierte Aktionen.
  - Wenn Knoten in Ihren Cluster eingebunden werden, nehmen sie eine IAM-Rolle an. Die Möglichkeit, mit IAM-Prinzipalen auf Ihren Cluster zuzugreifen, wird durch den AWS IAM Authenticator für Kubernetes bereitgestellt, der auf der Steuerebene von Amazon EKS ausgeführt wird.
- Ein Benutzer in Ihrem eigenen OpenID Connect (OIDC)-Anbieter – Dieser Typ erfordert eine Authentifizierung bei Ihrem OIDC-Anbieter. Weitere Informationen zur Einrichtung Ihres eigenen OIDC-Anbieters mit Ihrem Amazon-EKS-Cluster finden Sie unter the section called “OIDC-Anbieter verknüpfen”. Wenn Benutzer diese Art von Identität verwenden, gilt Folgendes:
  - Kann ihnen Kubernetes-Berechtigungen zuweisen, damit sie mit Kubernetes-Objekten in Ihrem Cluster arbeiten können.
  - Ihnen können keine IAM-Berechtigungen zugewiesen werden, sodass sie mithilfe der Amazon EKS-API, AWS CLI,, oder `eksctl` mit Ihrem Amazon EKS-Cluster und seinen Ressourcen arbeiten können. AWS CloudFormation AWS-Managementkonsole

Sie können beide Arten von Identitäten mit Ihrem Cluster verwenden. Die IAM-Authentifizierungsmethode kann nicht deaktiviert werden. Die OIDC-Authentifizierungsmethode ist optional.

IAM-Identitäten Kubernetes-Berechtigungen zuordnen

Der AWS IAM Authenticator für Kubernetes ist auf der Steuerebene Ihres Clusters installiert. Er ermöglicht AWS Identity and Access Management (IAM)-Prinzipale (Rollen und Benutzer), denen Sie den Zugriff auf Kubernetes-Ressourcen in Ihrem Cluster gestatten. Sie können eine der folgenden Methoden verwenden, um IAM-Prinzipalen Zugriff auf Kubernetes-Objekte in Ihrem Cluster zu gewähren:

- Erstellen von Zugriffseinträgen – Wenn Ihr Cluster mindestens über die Plattformversion verfügt, die im Abschnitt Voraussetzungen für die Kubernetes-Version Ihres Clusters aufgeführt ist, sollten Sie diese Option verwenden.

  Verwenden Sie Zugriffseinträge, um die Kubernetes-Berechtigungen von IAM-Prinzipalen außerhalb des Clusters zu verwalten. Sie können den Zugriff auf den Cluster mithilfe der EKS-API, der AWS Befehlszeilenschnittstelle, AWS SDKs AWS CloudFormation, und hinzufügen und AWS-Managementkonsole verwalten. Für die Benutzerverwaltung können also die gleichen Tools verwendet werden wie für die Clustererstellung.

  Folgen Sie zunächst den Anweisungen Authentifizierungsmodus zur Verwendung von Zugriffseinträgen ändern und anschließend Vorhandene aws-auth-Einträge zu ConfigMap Zugriffseinträgen migrieren.
- Hinzufügen von Einträgen zu **`aws-auth ConfigMap`** : Wenn die Plattformversion Ihres Clusters älter ist als die im Abschnitt Voraussetzungen aufgeführte Version, muss diese Option verwendet werden. Falls die Plattformversion Ihres Clusters mindestens der Plattformversion entspricht, die im Abschnitt Voraussetzungen für die Kubernetes-Version Ihres Clusters angegeben ist, und Sie Einträge zu `ConfigMap` hinzugefügt haben, empfiehlt es sich, diese Einträge zu Zugriffseinträgen zu migrieren. Sie können allerdings keine Einträge migrieren, die von Amazon EKS zu `ConfigMap` hinzugefügt wurden (also beispielsweise Einträge für IAM-Rollen, die mit verwalteten Knotengruppen oder Fargate-Profilen verwendet werden). Weitere Informationen finden Sie unter the section called “Kubernetes-API-Zugriff”.
  - Wenn Sie `aws-auth ConfigMap` verwenden müssen, können Sie mithilfe des Befehls `eksctl create iamidentitymapping` Einträge zu `ConfigMap` hinzufügen. Weitere Informationen finden Sie in der Dokumentation zu `eksctl` unter Manage IAM users and roles.

Einstellen des Cluster-Authentifizierungsmodus

Jeder Cluster hat einen Authentifizierungsmodus. Der Authentifizierungsmodus bestimmt, welche Methoden Sie verwenden können, um IAM-Prinzipalen den Zugriff auf Kubernetes-Objekte in Ihrem Cluster zu ermöglichen. Es gibt drei Authentifizierungsmodi.

> **Important**
>
> Sobald die Zugriffseingabemethode aktiviert ist, kann sie nicht mehr deaktiviert werden. Wenn die `ConfigMap`-Methode während der Cluster-Erstellung nicht aktiviert wird, kann sie später nicht mehr aktiviert werden. Für alle Cluster, die vor der Einführung von Zugriffseinträgen erstellt wurden, ist die `ConfigMap`-Methode aktiviert.
> Wenn Sie Hybridknoten mit Ihrem Cluster verwenden, müssen Sie die Cluster-Authentifizierungsmodi `API` oder `API_AND_CONFIG_MAP` verwenden.

Die **`aws-auth` `ConfigMap`** innerhalb des Clusters

Dies ist der ursprüngliche Authentifizierungsmodus für Amazon-EKS-Cluster. Der IAM-Prinzipal, der den Cluster erstellt hat, ist zunächst der einzige Benutzer, der über `kubectl` auf den Cluster zugreifen kann. Dieser Benutzer muss der Liste in `aws-auth` `ConfigMap` weitere Benutzer hinzufügen und Berechtigungen zuweisen, die sich auf die anderen Benutzer innerhalb des Clusters auswirken. Die anderen Benutzer können den ursprünglichen Benutzer nicht verwalten oder entfernen, da `ConfigMap` keinen zu verwaltenden Eintrag enthält.

Sowohl die als auch die Zugriffseinträge **`ConfigMap`**

In diesem Authentifizierungsmodus können Sie beide Methoden verwenden, um dem Cluster IAM-Prinzipale hinzuzufügen. Beachten Sie, dass jede Methode separate Einträge speichert. Wenn Sie beispielsweise einen Zugriffseintrag über die AWS CLI hinzufügen, `aws-auth` `ConfigMap` wird dieser nicht aktualisiert.

Nur auf Einträge zugreifen

In diesem Authentifizierungsmodus können Sie die EKS-API, die AWS Befehlszeilenschnittstelle,, und verwenden AWS SDKs AWS CloudFormation, AWS-Managementkonsole um den Zugriff auf den Cluster für IAM-Prinzipale zu verwalten.

Jeder Zugriffseintrag hat einen Typ und Sie können eine Kombination aus Zugriffsbereich und Zugriffsrichtlinie verwenden, um mithilfe des Zugriffsbereichs den Prinzipal auf einen bestimmten

Namespace zu beschränken und mithilfe der Zugriffsrichtlinie vorkonfigurierte, wiederverwendbare Berechtigungsrichtlinien festzulegen. Alternativ können Sie den Typ STANDARD und Kubernetes RBAC-Gruppen verwenden, um benutzerdefinierte Berechtigungen zuzuweisen.

Authentifizierungsmodus	Methoden
Nur `ConfigMap` (`CONFIG_MAP`)	`aws-auth ConfigMap`
EKS-API und `ConfigMap` (`API_AND_CONFIG_MAP`)	auf Einträge in der EKS-API, der AWS Befehlszeilenschnittstelle, AWS SDKs AWS CloudFormation, und zugreifen AWS-Managementkonsole `aws-auth ConfigMap`
Nur EKS-API (`API`)	auf Einträge in der EKS-API, der AWS Befehlszeilenschnittstelle, AWS SDKs, AWS CloudFormation, zugreifen und AWS-Managementkonsole

> **Note**
>
> Amazon EKS Auto Mode erfordert Zugriffseinträge.

IAM-Benutzern mit EKS-Zugriffseinträgen Zugriff auf Kubernetes gewähren

Dieser Abschnitt zeigt Ihnen, wie Sie den IAM-Prinzipalzugriff auf Kubernetes-Cluster in Amazon Elastic Kubernetes Service (EKS) mithilfe von Zugriffseinträgen und Richtlinien verwalten. Sie finden detaillierte Informationen zum Ändern von Authentifizierungsmodi, zur Migration von veralteten `aws-auth`-ConfigMap-Einträgen, zum Erstellen, Aktualisieren und Löschen von Zugriffseinträgen, zum Verknüpfen von Richtlinien mit Einträgen, zum Überprüfen vordefinierter Richtlinienberechtigungen sowie zu wichtigen Voraussetzungen und Überlegungen für eine sichere Zugriffsverwaltung.

Übersicht

EKS-Zugriffseinträge sind die optimale Methode, um Benutzern Zugriff auf die Kubernetes-API zu gewähren. Beispielsweise können Sie Zugriffseinträge verwenden, um Entwicklern Zugriff auf die

Verwendung von kubectl zu gewähren. Grundsätzlich ordnet ein EKS-Zugriffseintrag eine Reihe von Kubernetes-Berechtigungen einer IAM-Identität zu, beispielsweise einer IAM-Rolle. Ein Entwickler kann beispielsweise eine IAM-Rolle übernehmen und diese zur Authentifizierung bei einem EKS-Cluster verwenden.

Feature

- Zentralisierte Authentifizierung und Autorisierung: Steuert den Zugriff auf Kubernetes-Cluster direkt über Amazon-EKS-APIs, sodass für Benutzerberechtigungen kein Wechsel zwischen AWS- und Kubernetes-APIs erforderlich ist.
- Differenzierte Berechtigungsverwaltung: Verwendet Zugriffseinträge und Richtlinien, um differenzierte Berechtigungen für AWS-IAM-Prinzipale zu definieren, einschließlich der Änderung oder Aufhebung des Cluster-Admin-Zugriffs durch den Ersteller.
- IaC-Tool-Integration: Unterstützt Infrastructure-as-Code-Tools wie AWS CloudFormation, Terraform und AWS CDK, um Zugriffskonfigurationen während der Cluster-Erstellung zu definieren.
- Wiederherstellung nach Fehlkonfiguration: Ermöglicht die Wiederherstellung des Cluster-Zugriffs über die Amazon-EKS-API ohne direkten Zugriff auf die Kubernetes-API.
- Reduzierte Gemeinkosten und verbesserte Sicherheit: Zentralisiert den Betrieb, um die Gemeinkosten zu senken, und nutzt gleichzeitig AWS-IAM-Features wie CloudTrail-Auditprotokollierung und Multi-Faktor-Authentifizierung.

So fügen Sie Berechtigungen hinzu

Sie können Kubernetes-Berechtigungen für den Zugriff auf Einträge auf zwei Arten anfügen:

- Verwenden Sie eine Zugriffsrichtlinie. Zugriffsrichtlinien sind vordefinierte Kubernetes-Berechtigungsvorlagen, die von AWS verwaltet werden. Weitere Informationen finden Sie unter the section called “Überprüfung von Zugriffsrichtlinien”.
- Verweisen Sie auf eine Kubernetes-Gruppe. Wenn Sie eine IAM-Identität einer Kubernetes-Gruppe zuordnen, können Sie Kubernetes-Ressourcen erstellen, die der Gruppe Berechtigungen gewähren. Weitere Informationen finden Sie unter Using RBAC Authorization in der Kubernetes-Dokumentation.

Überlegungen

Beachten Sie bei der Aktivierung von EKS-Zugriffseinträgen in vorhandenen Clustern Folgendes:

- Verhalten älterer Cluster: Bei Clustern, die vor der Einführung von Zugriffseinträgen erstellt wurden (d. h. Clustern mit einer Plattformversion, die älter ist als in den Anforderungen an die Plattformversion angegeben), erstellt EKS automatisch einen Zugriffseintrag, der die bereits vorhandenen Berechtigungen widerspiegelt. Dieser Eintrag enthält die IAM-Identität, mit welcher der Cluster ursprünglich erstellt wurde, sowie die Verwaltungsberechtigungen, die dieser Identität bei der Cluster-Erstellung gewährt wurden.
- Umgang mit der veralteten **`aws-auth`**-ConfigMap: Wenn Ihr Cluster für die Zugriffsverwaltung auf die veraltete `aws-auth`-ConfigMap angewiesen ist, wird bei der Aktivierung von Zugriffseinträgen automatisch nur der Zugriffseintrag für den ursprünglichen Cluster-Ersteller erstellt. Zusätzliche Rollen oder Berechtigungen, die der ConfigMap hinzugefügt werden (z. B. benutzerdefinierte IAM-Rollen für Entwickler oder Services), werden nicht automatisch migriert. Um dies zu beheben, erstellen Sie manuell die entsprechenden Zugriffseinträge.

Erste Schritte

1. Legen Sie die IAM-Identitäts- und Zugriffsrichtlinie fest, die Sie verwenden möchten.
   - the section called “Überprüfung von Zugriffsrichtlinien”
2. Aktivieren Sie EKS-Zugriffseinträge in Ihrem Cluster. Bestätigen Sie, dass Sie über eine unterstützte Plattformversion verfügen.
   - the section called “Authentifizierungsmodus”
3. Erstellen Sie einen Zugriffseintrag, der eine IAM-Identität einer Kubernetes-Berechtigung zuordnet.
   - the section called “Zugriffseinträge erstellen”
4. Authentifizieren Sie sich beim Cluster mit der IAM-Identität.
   - the section called “AWS-CLI einrichten”
   - the section called “kubectl und eksctl einrichten”

Zugriffsrichtlinien mit Zugriffseinträgen verknüpfen

Sie können Zugriffseinträgen vom Typ `STANDARD` eine oder mehrere Zugriffsrichtlinien zuweisen. Amazon EKS erteilt den anderen Arten von Zugriffseinträgen automatisch die erforderlichen Berechtigungen, damit sie in Ihrem Cluster ordnungsgemäß funktionieren. Die Amazon-EKS-Zugriffsrichtlinien beinhalten Kubernetes-Berechtigungen, keine IAM-Berechtigungen. Machen Sie sich mit den Kubernetes-Berechtigungen vertraut, die in den einzelnen Zugriffsrichtlinien-Optionen enthalten sind, bevor Sie einem Zugriffseintrag eine Zugriffsrichtlinie zuweisen. Weitere

Informationen finden Sie unter the section called “Überprüfung von Zugriffsrichtlinien”. Wenn keine der Zugriffsrichtlinien Ihre Anforderungen erfüllt, ordnen Sie einem Zugriffseintrag keine Zugriffsrichtlinie zu. Geben Sie stattdessen mindestens einen Gruppennamen für den Zugriffseintrag an und erstellen und verwalten Sie Kubernetes-Objekte für die rollenbasierte Zugriffskontrolle. Weitere Informationen finden Sie unter the section called “Zugriffseinträge erstellen”.

- Ein vorhandener Zugriffseintrag. Informationen zum Erstellen finden Sie unter the section called “Zugriffseinträge erstellen”.
- Eine Rolle oder ein entsprechender Benutzer für AWS Identity and Access Management mit folgenden Berechtigungen: `ListAccessEntries`, `DescribeAccessEntry`, `UpdateAccessEntry`, `ListAccessPolicies`, `AssociateAccessPolicy` und `DisassociateAccessPolicy`. Weitere Informationen finden Sie unter Von Amazon Elastic Kubernetes Service definierte Aktionen in der Service-Autorisierungsreferenz.

Berücksichtigen Sie die folgenden Anforderungen, bevor Sie Zugriffsrichtlinien mit Zugriffseinträgen verknüpfen:

- Sie können jedem Zugriffseintrag mehrere Zugriffsrichtlinien zuordnen, aber Sie können jede Richtlinie nur einmal einem Zugriffseintrag zuordnen. Wenn Sie mehrere Zugriffsrichtlinien zuordnen, verfügt der IAM-Prinzipal des Zugriffseintrags über alle Berechtigungen aus allen zugeordneten Zugriffsrichtlinien.
- Sie können eine Zugriffsrichtlinie auf alle Ressourcen in einem Cluster ausrichten oder den Namen eines oder mehrerer Kubernetes-Namespaces angeben. Sie können Platzhalterzeichen für einen Namespace-Namen verwenden. Wenn Sie beispielsweise eine Zugriffsrichtlinie auf alle Namespaces ausrichten möchten, die mit `dev-` beginnen, können Sie `dev-*` als Namespace-Namen angeben. Stellen Sie sicher, dass die Namespaces in Ihrem Cluster vorhanden sind und dass Ihre Schreibweise dem tatsächlichen Namespace-Namen im Cluster entspricht. Amazon EKS überprüft weder die Schreibweise noch das Vorhandensein der Namespaces in Ihrem Cluster.
- Der Zugriffsbereich für eine Zugriffsrichtlinie kann geändert werden, nachdem sie einem Zugriffseintrag zugeordnet wurde. Wenn Sie die Zugriffsrichtlinie auf Kubernetes-Namespaces beschränkt haben, können Sie bei Bedarf Namespaces für die Zuordnung hinzufügen und entfernen.
- Wenn Sie eine Zugriffsrichtlinie einem Zugriffseintrag zuordnen, für den auch Gruppennamen angegeben sind, verfügt der IAM-Prinzipal über alle Berechtigungen in allen zugehörigen Zugriffsrichtlinien. Außerdem verfügt er über alle Berechtigungen aus jedem `Role`-Objekt vom

Typ `ClusterRole` oder `Role`, das in einem beliebigen Kubernetes-Objekt vom Typ oder `RoleBinding` zum Angeben der Gruppennamen angegeben ist.

- Wenn Sie den Befehl `kubectl auth can-i --list` ausführen, werden keine Kubernetes-Berechtigungen angezeigt, die durch Zugriffsrichtlinien zugewiesen wurden, die einem Zugriffseintrag für den IAM-Prinzipal zugeordnet sind, den Sie beim Ausführen des Befehls verwenden. Der Befehl zeigt nur Kubernetes-Berechtigungen an, wenn Sie sie in Kubernetes-Objekten vom Typ `Role` oder `ClusterRole` gewährt haben, die Sie an die Gruppennamen oder an den Benutzernamen gebunden haben, die bzw. den Sie für einen Zugriffseintrag angegeben haben.
- Wenn Sie bei der Interaktion mit Kubernetes-Objekten in Ihrem Cluster die Identität eines Kubernetes-Benutzers oder einer Kubernetes-Gruppe annehmen, beispielsweise indem Sie den Befehl `kubectl` mit `--as` *`username`* oder `--as-group` *`group-name`* verwenden, erzwingen Sie die Verwendung der Kubernetes RBAC-Autorisierung. Daher werden dem IAM-Prinzipal keine Berechtigungen durch eine Zugriffsrichtlinie zugewiesen, die dem Zugriffseintrag zugeordnet ist. Die einzigen Kubernetes-Berechtigungen, über die der Benutzer oder die Gruppe verfügt, dessen bzw. deren Identität der IAM-Prinzipal angenommen hat, sind die Kubernetes-Berechtigungen, die Sie in Kubernetes-Objekten vom Typ `Role` oder `ClusterRole` gewährt haben, die von Ihnen an die Gruppennamen oder an den Benutzernamen gebunden wurden. Damit Ihr IAM-Prinzipal über die Berechtigungen in den zugehörigen Zugriffsrichtlinien verfügt, dürfen Sie sich nicht als Kubernetes-Benutzer oder -Gruppe ausgeben. Der IAM-Prinzipal verfügt auch weiterhin über alle Berechtigungen, die Sie ihm in den Kubernetes-Objekten vom Typ `Role` oder `ClusterRole` gewährt haben, die Sie an die Gruppennamen oder an den Benutzernamen gebunden haben, die bzw. den Sie für den Zugriffseintrag angegeben haben. Weitere Informationen finden Sie unter Benutzeridentitätswechsel in der Kubernetes-Dokumentation.

Sie können die AWS-Managementkonsole oder die AWS CLI verwenden, um eine Zugriffsrichtlinie einem Zugriffseintrag zuzuordnen.

AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie den Namen des Clusters aus, der über einen Zugriffseintrag verfügt, dem Sie eine Zugriffsrichtlinien zuordnen möchten.
3. Wählen Sie die Registerkarte Zugriff aus.

4. Wenn es sich um einen Zugriffseintrag vom Typ Standard handelt, können Sie Zugriffsrichtlinien von Amazon EKS zuordnen oder deren Zuordnung aufheben. Bei Zugriffseinträgen eines anderen Typs (also nicht Standard) steht diese Option nicht zur Verfügung.
5. Wählen Sie Zugriffsrichtlinie zuordnen aus.
6. Wählen Sie unter Richtlinienname die Richtlinie mit den Berechtigungen aus, über die der IAM-Prinzipal verfügen soll. Informationen zu den Berechtigungen, die in der jeweiligen Richtlinie enthalten sind, finden Sie unter the section called “Überprüfung von Zugriffsrichtlinien”.
7. Wählen Sie unter Zugriffsbereich einen Zugriffsbereich aus. Bei Verwendung der Option Cluster werden die Berechtigungen in der Zugriffsrichtlinie dem IAM-Prinzipal für Ressourcen in allen Kubernetes-Namespaces gewährt. Bei Verwendung der Option Kubernetes-Namespace können Sie anschließend Neuen Namespace hinzufügen auswählen. In dem daraufhin angezeigten Feld Namespace können Sie den Namen eines Kubernetes-Namespace in Ihrem Cluster eingeben. Wenn die Berechtigungen für den IAM-Prinzipal in mehreren Namespaces gelten sollen, können Sie mehrere Namespaces eingeben.
8. Wählen Sie Zugriffsrichtlinie hinzufügen aus.

AWS-CLI

1. Version `2.12.3` oder höher oder die Version `1.27.160` oder höher der AWS-Befehlszeilenschnittstelle (AWS-CLI), die auf Ihrem Gerät oder in AWS CloudShell installiert und konfiguriert sein muss. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Paket-Manager wie `yum`, `apt-get` oder Homebrew für macOS sind oft mehrere Versionen hinter der neuesten Version der AWS CLI. Um die neueste Version zu installieren, lesen Sie Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch zur AWS-Befehlszeilenschnittstelle. Die in AWS CloudShell installierte AWS-CLI-Version kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zum Aktualisieren finden Sie unter Installieren der AWS CLI in Ihrem Stammverzeichnis im AWS-CloudShell-Benutzerhandbuch.
2. Sehen Sie sich die verfügbaren Zugriffsrichtlinien an.

```
aws eks list-access-policies --output table
```

Eine Beispielausgabe sieht wie folgt aus.

```
----------------------------------------------------------------------------------------
```

```
|                                          ListAccessPolicies
                  |
+-------------------------------------------------------------------------------------------
+
||                                            accessPolicies
                 ||
|+-----------------------------------------------------------------
+------------------------------+|
||                                arn                               |
 name              ||
|+-----------------------------------------------------------------
+------------------------------+|
||  {arn-aws}eks::aws:cluster-access-policy/AmazonEKSAdminPolicy        |
 AmazonEKSAdminPolicy         ||
||  {arn-aws}eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy |
 AmazonEKSClusterAdminPolicy  ||
||  {arn-aws}eks::aws:cluster-access-policy/AmazonEKSEditPolicy         |
 AmazonEKSEditPolicy          ||
||  {arn-aws}eks::aws:cluster-access-policy/AmazonEKSViewPolicy         |
 AmazonEKSViewPolicy          ||
|+-----------------------------------------------------------------
+------------------------------+|
```

Informationen zu den Berechtigungen, die in der jeweiligen Richtlinie enthalten sind, finden Sie unter the section called “Überprüfung von Zugriffsrichtlinien”.

3. Sehen Sie sich Ihre vorhandenen Zugriffseinträge an. Ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters.

```
aws eks list-access-entries --cluster-name my-cluster
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "accessEntries": [
        "arn:aws:iam::111122223333:role/my-role",
        "arn:aws:iam::111122223333:user/my-user"
    ]
}
```

4. Ordnen Sie eine Zugriffsrichtlinie einem Zugriffseintrag zu. Im folgenden Beispiel wird die Zugriffsrichtlinie `AmazonEKSViewPolicy` einem Zugriffseintrag zugeordnet. Wenn die IAM-Rolle

my-role versucht, auf Kubernetes-Objekte im Cluster zuzugreifen, autorisiert Amazon EKS die Rolle nur dazu, die Berechtigungen in der Richtlinie für den Zugriff auf Kubernetes-Objekte in den Kubernetes-Namespaces *my-namespace1* und *my-namespace2* zu verwenden. Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters, *111122223333* durch Ihre AWS-Konto-ID und *my-role* durch den Namen der IAM-Rolle, für die Amazon EKS den Zugriff auf Kubernetes-Cluster-Objekte autorisieren soll.

```
aws eks associate-access-policy --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role \
    --access-scope type=namespace,namespaces=my-namespace1,my-namespace2 --policy-arn
 arn:aws:eks::aws:cluster-access-policy/AmazonEKSViewPolicy
```

Wenn die Berechtigungen für den IAM-Prinzipal clusterweit gelten sollen, ersetzen Sie `type=namespace,namespaces=`*my-namespace1*`,`*my-namespace2* durch `type=cluster`. Wenn Sie dem Zugriffseintrag mehrere Zugriffsrichtlinien zuordnen möchten, führen Sie den Befehl mehrmals aus und verwenden Sie dabei jeweils eine individuelle Zugriffsrichtlinie. Jede zugeordnete Zugriffsrichtlinie hat ihren eigenen Geltungsbereich.

> **Note**
>
> Wenn Sie später den Geltungsbereich einer zugehörigen Zugriffsrichtlinie ändern möchten, können Sie den vorherigen Befehl erneut ausführen und dabei den neuen Bereich angeben. Wenn Sie also beispielsweise *my-namespace2* entfernen möchten, führen Sie den Befehl erneut aus und verwenden Sie dabei nur `type=namespace,namespaces=`*my-namespace1* . Wenn Sie den Bereich von `namespace` in `cluster` ändern möchten, führen Sie den Befehl erneut aus und verwenden Sie dabei `type=cluster`, um `type=namespace,namespaces=`*my-namespace1*`,`*my-namespace2* zu entfernen.

5. Ermitteln Sie, welche Zugriffsrichtlinien einem Zugriffseintrag zugeordnet sind.

```
aws eks list-associated-access-policies --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "clusterName": "my-cluster",
```

```
    "principalArn": "arn:aws:iam::111122223333",
    "associatedAccessPolicies": [
        {
            "policyArn": "arn:aws:eks::aws:cluster-access-policy/
AmazonEKSViewPolicy",
            "accessScope": {
                "type": "cluster",
                "namespaces": []
            },
            "associatedAt": "2023-04-17T15:25:21.675000-04:00",
            "modifiedAt": "2023-04-17T15:25:21.675000-04:00"
        },
        {
            "policyArn": "arn:aws:eks::aws:cluster-access-policy/
AmazonEKSAdminPolicy",
            "accessScope": {
                "type": "namespace",
                "namespaces": [
                    "my-namespace1",
                    "my-namespace2"
                ]
            },
            "associatedAt": "2023-04-17T15:02:06.511000-04:00",
            "modifiedAt": "2023-04-17T15:02:06.511000-04:00"
        }
    ]
}
```

Im vorherigen Beispiel verfügt der IAM-Prinzipal für diesen Zugriffseintrag über Leseberechtigungen für alle Namespaces im Cluster und über Administratorberechtigungen für zwei Kubernetes-Namespaces.

6. Heben Sie die Zuordnung zwischen einer Zugriffsrichtlinie und einem Zugriffseintrag auf. In diesem Beispiel wird die Zuordnung zwischen der Richtlinie `AmazonEKSAdminPolicy` und einem Zugriffseintrag aufgehoben. Die in der Zugriffsrichtlinie `AmazonEKSViewPolicy` enthaltenen Berechtigungen für Objekte in den Namespaces *`my-namespace1`* und *`my-namespace2`* bleiben für den IAM-Prinzipal allerdings erhalten, da die Zuordnung zwischen dieser Zugriffsrichtlinie und dem Zugriffseintrag nicht aufgehoben wird.

```
aws eks disassociate-access-policy --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role \
    --policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminPolicy
```

Eine Liste der verfügbaren Zugriffsrichtlinien finden Sie unter the section called “Überprüfung von Zugriffsrichtlinien”.

Migrieren vorhandener **`aws-auth ConfigMap`**-Einträge zu Zugriffseinträgen

Wenn Sie Einträge zu `aws-auth ConfigMap` für Ihren Cluster hinzugefügt haben, empfiehlt es sich, Zugriffseinträge für die vorhandenen Einträge in `aws-auth ConfigMap` zu erstellen. Nach der Erstellung der Zugriffseinträge können Sie die Einträge aus `ConfigMap` entfernen. Einträgen in `aws-auth ConfigMap` können keine Zugriffsrichtlinien zugeordnet werden. Wenn Sie Ihren IAM-Prinzipalen Zugriffsrichtlinien zuordnen möchten, müssen Sie Zugriffseinträge erstellen.

⚠ Important

- Wenn sich ein Cluster im `API_AND_CONFIGMAP`-Authentifizierungsmodus befindet und sowohl in den `aws-auth ConfigMap`- als auch in den Zugriffseinträgen eine Zuordnung für dieselbe IAM-Rolle vorhanden ist, verwendet die Rolle die Zuordnung des Zugriffseintrags zur Authentifizierung. Zugriffseinträge haben Vorrang vor `ConfigMap`-Einträgen für denselben IAM-Prinzipal.
- Bevor Sie vorhandene `aws-auth ConfigMap`-Einträge entfernen, die von Amazon EKS für eine verwaltete Knotengruppe oder ein Fargate-Profil in Ihrem Cluster erstellt wurden, überprüfen Sie, ob die richtigen Zugriffseinträge für diese spezifischen Ressourcen in Ihrem Amazon-EKS-Cluster vorhanden sind. Wenn Sie Einträge entfernen, die Amazon EKS in `ConfigMap` erstellt hat, ohne die entsprechenden Zugriffseinträge zu haben, funktioniert Ihr Cluster nicht ordnungsgemäß.

Voraussetzungen

- Vertrautheit mit Zugriffseinträgen und Zugriffsrichtlinien. Weitere Informationen erhalten Sie unter the section called “Zugriffseinträge” und the section called “Zuordnen von Zugriffsrichtlinien”.
- Ein vorhandener Cluster mit einer Plattformversion, die den in den Voraussetzungen des the section called “Zugriffseinträge”-Themas aufgeführten Versionen entspricht oder höher ist.
- Version `0.215.0` oder höher des `eksctl`-Befehlszeilen-Tools, das auf Ihrem Computer oder in der AWS CloudShell installiert ist. Informationen zum Installieren und Aktualisieren von `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.
- Kubernetes-Berechtigungen zum Ändern von `aws-auth ConfigMap` im `kube-system`-Namespace.

- Eine AWS Identity and Access Management Zugriffsverwaltungsrolle oder ein Benutzer mit den folgenden Berechtigungen: `CreateAccessEntry` und`ListAccessEntries`. Weitere Informationen finden Sie in der Service-Authorization-Referenz unter Von Amazon Elastic Kubernetes Service definierte Aktionen.

`eksctl`

1. Sehen Sie sich die vorhandenen Einträge in `aws-auth ConfigMap` an. Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters.

   ```
   eksctl get iamidentitymapping --cluster my-cluster
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   ARN
              USERNAME                                      GROUPS
                           ACCOUNT
   arn:aws: iam::111122223333:role/EKS-my-cluster-Admins
               Admins                                        system:masters
   arn:aws: iam::111122223333:role/EKS-my-cluster-my-namespace-Viewers
               my-namespace-Viewers                          Viewers
   arn:aws: iam::111122223333:role/EKS-my-cluster-self-managed-ng-1
               system:node:{{EC2PrivateDNSName}}             system:bootstrappers,system:nodes
   arn:aws: iam::111122223333:user/my-user
               my-user
   arn:aws: iam::111122223333:role/EKS-my-cluster-fargateprofile1
                                 system:node:{{SessionName}}
    system:bootstrappers,system:nodes,system:node-proxier
   arn:aws: iam::111122223333:role/EKS-my-cluster-managed-ng
               system:node:{{EC2PrivateDNSName}}             system:bootstrappers,system:nodes
   ```

2. the section called “Zugriffseinträge erstellen” für alle von Ihnen erstellten `ConfigMap`-Einträge aus der vorherigen Ausgabe. Achten Sie beim Erstellen der Zugriffseinträge darauf, für `ARN`, `USERNAME`, `GROUPS` und `ACCOUNT` die gleichen Werte anzugeben, die in der Ausgabe zurückgegeben wurden. Im Falle der Beispielausgabe würden Sie Zugriffseinträge für alle Einträge außer den letzten beiden Einträgen erstellen, da diese von Amazon EKS für ein Fargate-Profil bzw. für eine verwaltete Knotengruppe erstellt wurden.
3. Löschen Sie die Einträge aus `ConfigMap` für alle von Ihnen erstellten Zugriffseinträge. Wenn Sie den Eintrag nicht aus `ConfigMap` löschen, wird der `ConfigMap`-Eintrag durch die Einstellungen

für den Zugriffseintrag für den IAM-Prinzipal-ARN außer Kraft gesetzt. *111122223333*Ersetzen Sie es durch Ihre AWS Konto-ID und *EKS-my-cluster-my-namespace-Viewers* durch den Namen der Rolle im Eintrag in Ihrem`ConfigMap`. Wenn der Eintrag, den Sie entfernen möchten, für einen IAM-Benutzer und nicht für eine IAM-Rolle bestimmt ist, `role` ersetzen Sie ihn durch `user` und *EKS-my-cluster-my-namespace-Viewers* durch den Benutzernamen.

```
eksctl delete iamidentitymapping --arn arn:aws: iam::111122223333:role/EKS-my-
cluster-my-namespace-Viewers --cluster my-cluster
```

Berechtigungen von Zugriffsrichtlinien überprüfen

Zugriffsrichtlinien beinhalten `rules`, die Kubernetes `verbs` (Berechtigungen) und `resources` enthalten. Zugriffsrichtlinien beinhalten keine IAM-Berechtigungen oder -Ressourcen. Ähnlich wie bei Kubernetes `Role` und `ClusterRole`-Objekten beinhalten Zugriffsrichtlinien nur `allow` `rules`. Der Inhalt einer Zugriffsrichtlinie kann nicht geändert werden. Sie können keine eigenen Zugriffsrichtlinien erstellen. Wenn die Berechtigungen in den Zugriffsrichtlinien Ihre Anforderungen nicht erfüllen, können Sie Kubernetes-RBAC-Objekte erstellen und Gruppennamen für Ihre Zugriffseinträge angeben. Weitere Informationen finden Sie unter the section called “Zugriffseinträge erstellen”. Die in den Zugriffsrichtlinien enthaltenen Berechtigungen ähneln den Berechtigungen in den benutzerorientierten Cluster-Rollen von Kubernetes. Weitere Informationen finden Sie unterBenutzerorientierte Rollen in der Kubernetes-Dokumentation.

Alle Richtlinien auflisten

Verwenden Sie eine der auf dieser Seite aufgeführten Zugriffsrichtlinien oder rufen Sie mit der AWS CLI eine Liste aller verfügbaren Zugriffsrichtlinien ab:

```
aws eks list-access-policies
```

Die erwartete Ausgabe sollte wie folgt aussehen (der Kürze halber abgekürzt):

```
{
    "accessPolicies": [
        {
            "name": "AmazonAIOpsAssistantPolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/AmazonAIOpsAssistantPolicy"
        },
        {
            "name": "AmazonARCRegionSwitchScalingPolicy",
```

```
            "arn": "arn:aws:eks::aws:cluster-access-policy/
AmazonARCRegionSwitchScalingPolicy"
        },
        {
            "name": "AmazonEKSAdminPolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminPolicy"
        },
        {
            "name": "AmazonEKSAdminViewPolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminViewPolicy"
        },
        {
            "name": "AmazonEKSAutoNodePolicy",
            "arn": "arn:aws:eks::aws:cluster-access-policy/AmazonEKSAutoNodePolicy"
        }
        // Additional policies omitted
    ]
}
```

EKSAdminAmazon-Richtlinie

Diese Zugriffsrichtlinie umfasst Berechtigungen, die einem IAM-Prinzipal die meisten Berechtigungen für Ressourcen erteilen. Wenn dies einem Zugriffseintrag zugeordnet ist, umfasst sein Zugriffsbereich normalerweise einen oder mehrere Kubernetes-Namespaces. Wenn ein IAM-Prinzipal über Administratorzugriff auf alle Ressourcen in Ihrem Cluster verfügen soll, ordnen Sie die Zugriffsrichtlinie the section called “Amazon EKSCluster AdminPolicy” stattdessen Ihrem Zugriffseintrag zu.

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSAdminPolicy`

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
`apps`	`daemonsets` , `deployments` , `deployments/rollback` , `deployments/scale` , `replicasets` , `replicasets/scale`, `statefulsets` , `statefulsets/scale`	`create`, `delete`, `deletecollection` , `patch`, `update`

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
apps	controllerrevisions , daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , replicasets , replicasets/scale , replicasets/status , statefulsets , statefulsets/scale , statefulsets/status	get, list, watch
authorization.k8s.io	localsubjectaccessreviews	create
autoscaling	horizontalpodautoscalers	create, delete, deletecollection , patch, update
autoscaling	horizontalpodautoscalers , horizontalpodautoscalers/status	get, list, watch
batch	cronjobs, jobs	create, delete, deletecollection , patch, update
batch	cronjobs, cronjobs/status , jobs, jobs/status	get, list, watch
discovery.k8s.io	endpointslices	get, list, watch

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
extensions	daemonsets , deployments , deployments/rollback , deployments/scale , ingresses , networkpolicies , replicasets , replicasets/scale , replicationcontrollers/scale	create, delete, deletecollection , patch, update
extensions	daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , ingresses , ingresses/status , networkpolicies , replicasets , replicasets/scale , replicasets/status , replicationcontrollers/scale	get, list, watch
networking.k8s.io	ingresses , ingresses/status , networkpolicies	get, list, watch
networking.k8s.io	ingresses , networkpolicies	create, delete, deletecollection , patch, update
policy	poddisruptionbudgets	create, delete, deletecollection , patch, update
policy	poddisruptionbudgets , poddisruptionbudgets/status	get, list, watch

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
rbac.authorization.k8s.io	rolebindings , roles	create, delete, deletecollection , get, list, patch, update, watch
	configmaps , endpoints , persistentvolumeclaims , persistentvolumeclaims/status , pods, replicationcontrollers , replicationcontrollers/scale , serviceaccounts , services, services/status	get,list, watch
	pods/attach , pods/exec , pods/portforward , pods/proxy , secrets, services/proxy	get, list, watch
	configmaps , events, persistentvolumeclaims , replicationcontrollers , replicationcontrollers/scale , secrets, serviceaccounts , services, services/proxy	create, delete, deletecollection , patch, update
	pods, pods/attach , pods/exec , pods/portforward , pods/proxy	create, delete, deletecollection , patch, update

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
	`serviceaccounts`	`impersonate`
	`bindings`, `events`, `limitranges` , `namespaces/status` , `pods/log`, `pods/status` , `replicationcontrollers/status` , `resourcequotas` , `resourcequotas/status`	`get`, `list`, `watch`
	`namespaces`	`get`,`list`, `watch`

Amazon EKSCluster AdminPolicy

Diese Zugriffsrichtlinie umfasst Berechtigungen, die einem IAM-Prinzipaladministrator Zugriff auf einen Cluster gewähren. Wenn dies einem Zugriffseintrag zugeordnet ist, umfasst der Zugriffsbereich in der Regel keinen Kubernetes-Namespace, sondern den Cluster. Wenn Sie möchten, dass ein IAM-Prinzipal einen eingeschränkteren Verwaltungsbereich hat, sollten Sie stattdessen die the section called “EKSAdminAmazon-Richtlinie”-Zugriffsrichtlinie mit Ihrem Zugriffseintrag verknüpfen.

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy`

Kubernetes-API-Gruppen	Kubernetes, keine Ressource URLs	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
*		*	*
	*		*

Amazon EKSAdmin ViewPolicy

Diese Zugriffsrichtlinie umfasst Berechtigungen, die einem IAM-Prinzipal Zugriff auf list/view alle Ressourcen in einem Cluster gewähren. Beachten Sie, dass dies Kubernetes-Geheimnisse umfasst.

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSAdminViewPolicy`

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
*	*	get, list, watch

EKSEditAmazon-Richtlinie

Diese Zugriffsrichtlinie umfasst Berechtigungen, mit denen ein IAM-Prinzipal die meisten Kubernetes-Ressourcen bearbeiten kann.

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSEditPolicy`

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
apps	daemonsets , deployments , deployments/rollback , deployments/scale , replicasets , replicasets/scale, statefulsets , statefulsets/scale	create, delete, deletecollection , patch, update
apps	controllerrevisions , daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , replicasets , replicasets/scale , replicasets/status , statefulsets , statefulsets/scale , statefulsets/status	get, list, watch

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
autoscaling	horizontalpodautoscalers , horizontalpodautoscalers/status	get, list, watch
autoscaling	horizontalpodautoscalers	create, delete, deletecollection , patch, update
batch	cronjobs, jobs	create, delete, deletecollection , patch, update
batch	cronjobs, cronjobs/status , jobs, jobs/status	get, list, watch
discovery.k8s.io	endpointslices	get, list, watch
extensions	daemonsets , deployments , deployments/rollback , deployments/scale , ingresses , networkpolicies , replicasets , replicasets/scale , replicationcontrollers/scale	create, delete, deletecollection , patch, update

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
extensions	daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , ingresses , ingresses/status , networkpolicies , replicasets , replicasets/scale , replicasets/status , replicationcontrollers/scale	get, list, watch
networking.k8s.io	ingresses , networkpolicies	create, delete, deletecollection , patch, update
networking.k8s.io	ingresses , ingresses/status , networkpolicies	get, list, watch
policy	poddisruptionbudgets	create, delete, deletecollection , patch, update
policy	poddisruptionbudgets , poddisruptionbudgets/status	get, list, watch
	namespaces	get, list, watch
	pods/attach , pods/exec , pods/portforward , pods/proxy , secrets, services/proxy	get, list, watch
	serviceaccounts	impersonate

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
	pods, pods/attach , pods/exec , pods/portforward , pods/proxy	create, delete, deletecollection , patch, update
	configmaps , events, persistentvolumeclaims , replicationcontrollers , replicationcontrollers/scale , secrets, serviceaccounts , services, services/proxy	create, delete, deletecollection , patch, update
	configmaps , endpoints , persistentvolumeclaims , persistentvolumeclaims/status , pods, replicationcontrollers , replicationcontrollers/scale , serviceaccounts , services, services/status	get, list, watch
	bindings, events, limitranges , namespaces/status , pods/log, pods/status , replicationcontrollers/status , resourcequotas , resourcequotas/status	get, list, watch

EKSViewAmazon-Richtlinie

Diese Zugriffsrichtlinie umfasst Berechtigungen, mit denen ein IAM-Prinzipal die meisten Kubernetes-Ressourcen anzeigen kann.

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSViewPolicy`

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
apps	controllerrevisions , daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , replicasets , replicasets/scale , replicasets/status , statefulsets , statefulsets/scale , statefulsets/status	get, list, watch
autoscaling	horizontalpodautoscalers , horizontalpodautoscalers/status	get, list, watch
batch	cronjobs, cronjobs/status , jobs, jobs/status	get, list, watch
discovery.k8s.io	endpointslices	get, list, watch
extensions	daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , ingresses , ingresses/status , networkpo	get, list, watch

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
	licies , replicasets , replicasets/scale , replicasets/status , replicationcontrollers/scale	
networking.k8s.io	ingresses , ingresses/status , networkpolicies	get, list, watch
policy	poddisruptionbudgets , poddisruptionbudgets/status	get, list, watch
	configmaps , endpoints , persistentvolumeclaims , persistentvolumeclaims/status , pods, replicationcontrollers , replicationcontrollers/scale , serviceaccounts , services, services/status	get, list, watch
	bindings, events, limitranges , namespaces/status , pods/log, pods/status , replicationcontrollers/status , resourcequotas , esourcequotas/status	get, list, watch
	namespaces	get, list, watch

Amazon EKSSecret ReaderPolicy

Diese Zugriffsrichtlinie beinhaltet Berechtigungen, die es einem IAM-Prinzipal ermöglichen, Kubernetes Secrets zu lesen.

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSSecretReaderPolicy`

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
	`secrets`	`get`, `list`, `watch`

Amazon EKSAuto NodePolicy

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSAutoNodePolicy`

Diese Richtlinie umfasst die folgenden Berechtigungen, mit denen Amazon-EKS-Komponenten die folgenden Aufgaben ausführen können:

- `kube-proxy` – Überwachen Sie Netzwerkendpunkte und -services und verwalten Sie zugehörige Ereignisse. Dies ermöglicht eine Cluster-weite Netzwerk-Proxy-Funktionalität.
- `ipamd`— Verwaltung von AWS VPC-Netzwerkressourcen und Container-Netzwerkschnittstellen (CNI). Dadurch kann der IP-Adressverwaltungs-Daemon die Pod-Vernetzung verwalten.
- `coredns` – Greifen Sie auf Ressourcen zur Serviceerkennung wie Endpunkte und Services zu. Dies ermöglicht die DNS-Auflösung innerhalb des Clusters.
- `ebs-csi-driver` – Arbeiten Sie mit speicherbezogenen Ressourcen für Amazon-EBS-Volumes. Dies ermöglicht die dynamische Bereitstellung und Anbindung von persistenten Volumes.
- `neuron`— Überwachen Sie Knoten und Pods für AWS Neuron-Geräte. Dies ermöglicht die Verwaltung von AWS Inferentia- und Trainium-Beschleunigern.
- `node-monitoring-agent` – Zugriff auf Knotendiagnosen und Ereignisse. Dies ermöglicht die Überwachung des Cluster-Zustands und die Erfassung von Diagnosedaten.

Jede Komponente verwendet ein dediziertes Servicekonto und ist auf die für ihre spezifische Funktion erforderlichen Berechtigungen beschränkt.

Wenn Sie eine Node-IAM-Rolle in einem manuell angeben NodeClass, müssen Sie einen Zugriffseintrag erstellen, der die neue Node-IAM-Rolle dieser Zugriffsrichtlinie zuordnet.

Amazon EKSBlock StoragePolicy

> **Note**
>
> Diese Richtlinie ist nur für AWS dienstbezogene Rollen vorgesehen und kann nicht für vom Kunden verwaltete Rollen verwendet werden.

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSBlockStoragePolicy`

Diese Richtlinie umfasst Berechtigungen, mit denen Amazon EKS die Leader-Wahl und die Koordinierung von Ressourcen für Speichervorgänge verwalten kann:

- `coordination.k8s.io` – Erstellen und verwalten Sie Lease-Objekte für die Leader-Wahl. Dadurch können EKS-Speicherkomponenten ihre Aktivitäten über einen Mechanismus für die Leader-Wahl Cluster-weit koordinieren.

Die Richtlinie gilt für bestimmte Lease-Ressourcen, die von den EKS-Speicherkomponenten verwendet werden, um Konflikte beim Zugriff auf andere Koordinierungsressourcen im Cluster zu vermeiden.

Amazon EKS erstellt automatisch einen Zugriffseintrag mit dieser Zugriffsrichtlinie für die Cluster-IAM-Rolle, wenn der Automatikmodus aktiviert ist. Dadurch wird sichergestellt, dass die erforderlichen Berechtigungen vorhanden sind, damit die Blockspeicherfunktion ordnungsgemäß funktioniert.

Amazon EKSLoad BalancingPolicy

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSLoadBalancingPolicy`

Diese Richtlinie umfasst Berechtigungen, mit denen Amazon EKS Ressourcen für die Leader-Wahl zum Load Balancing verwalten kann:

- `coordination.k8s.io` – Erstellen und verwalten Sie Lease-Objekte für die Leader-Wahl. Dadurch können die EKS-Komponenten für das Load Balancing durch die Leader-Wahl Aktivitäten über mehrere Replikate hinweg koordinieren.

Die Richtlinie ist speziell auf das Load Balancing von Lease-Ressourcen ausgerichtet, um eine ordnungsgemäße Koordination sicherzustellen und gleichzeitig den Zugriff auf andere Leasing-Ressourcen im Cluster zu verhindern.

Amazon EKS erstellt automatisch einen Zugriffseintrag mit dieser Zugriffsrichtlinie für die Cluster-IAM-Rolle, wenn der Automatikmodus aktiviert ist. Dadurch wird sichergestellt, dass die erforderlichen Berechtigungen vorhanden sind, damit die Netzwerkfunktion ordnungsgemäß funktioniert.

EKSNetworkingAmazon-Richtlinie

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSNetworkingPolicy`

Diese Richtlinie umfasst Berechtigungen, mit denen Amazon EKS Ressourcen für die Leader-Wahl für die Vernetzung verwalten kann:

- `coordination.k8s.io` – Erstellen und verwalten Sie Lease-Objekte für die Leader-Wahl. Dadurch können EKS-Netzwerkkomponenten die Aktivitäten zur IP-Adresszuweisung durch die Leader-Wahl koordinieren.

Die Richtlinie ist speziell auf die Vernetzung von Lease-Ressourcen ausgerichtet, um eine ordnungsgemäße Koordination zu gewährleisten und gleichzeitig den Zugriff auf andere Lease-Ressourcen im Cluster zu verhindern.

Amazon EKS erstellt automatisch einen Zugriffseintrag mit dieser Zugriffsrichtlinie für die Cluster-IAM-Rolle, wenn der Automatikmodus aktiviert ist. Dadurch wird sichergestellt, dass die erforderlichen Berechtigungen vorhanden sind, damit die Netzwerkfunktion ordnungsgemäß funktioniert.

EKSComputeAmazon-Richtlinie

> **Note**
>
> Diese Richtlinie ist nur für AWS dienstbezogene Rollen vorgesehen und kann nicht für vom Kunden verwaltete Rollen verwendet werden.

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSComputePolicy`

Diese Richtlinie umfasst Berechtigungen, mit denen Amazon EKS Ressourcen für die Leader-Wahl für Rechenoperationen verwalten kann:

- `coordination.k8s.io` – Erstellen und verwalten Sie Lease-Objekte für die Leader-Wahl. Dadurch können EKS-Rechenkomponenten die Skalierungsaktivitäten der Knoten durch die Leader-Wahl koordinieren.

Die Richtlinie ist speziell auf die Berechnung von Verwaltungs-Lease-Ressourcen ausgerichtet und ermöglicht gleichzeitig den grundlegenden Lesezugriff (`get`, `watch`) auf alle Lease-Ressourcen im Cluster.

Amazon EKS erstellt automatisch einen Zugriffseintrag mit dieser Zugriffsrichtlinie für die Cluster-IAM-Rolle, wenn der Automatikmodus aktiviert ist. Dadurch wird sichergestellt, dass die erforderlichen Berechtigungen vorhanden sind, damit die Netzwerkfunktion ordnungsgemäß funktioniert.

Amazon EKSBlock StorageClusterPolicy

> **Note**
>
> Diese Richtlinie ist nur für AWS dienstbezogene Rollen vorgesehen und kann nicht für vom Kunden verwaltete Rollen verwendet werden.

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSBlockStorageClusterPolicy`

Diese Richtlinie gewährt die erforderlichen Berechtigungen für die Blockspeicherfunktion von Amazon EKS Auto Mode. Sie ermöglicht eine effiziente Verwaltung der Blockspeicherressourcen innerhalb von Amazon-EKS-Clustern. Die Richtlinie umfasst folgende Berechtigungen:

CSI-Treiberverwaltung:

- Erstellen, Lesen, Aktualisieren und Löschen von CSI-Treibern, insbesondere für Blockspeicher.

Volume-Verwaltung:

- Auflisten, Beobachten, Erstellen, Aktualisieren, Patchen und Löschen persistenter Volumes.
- Auflisten, Überwachen und Aktualisieren von persistenten Volume-Ansprüchen.
- Patchen Sie des Status von persistenten Volume-Ansprüchen.

Interaktion zwischen Knoten und Pods:

- Lesen Sie Informationen zu Knoten und Pods.
- Speichern Sie Ereignisse im Zusammenhang mit Speichervorgängen.

Speicherklassen und Attribute:

- Lesen Sie Speicherklassen und CSI-Knoten.
- Lesen Sie die Volume-Attributklassen.

Anhänge von Volumes:

- Auflisten, Überwachen und Ändern von Volume-Anhängen und deren Status.

Snapshot-Vorgänge:

- Verwalten Sie Volumen-Snapshots, Snapshot-Inhalte und Snapshot-Klassen.
- Verwalten Sie Vorgänge für Volume-Gruppen-Snapshots und zugehörige Ressourcen.

Diese Richtlinie wurde entwickelt, um eine umfassende Verwaltung von Blockspeichern innerhalb von Amazon-EKS-Clustern zu unterstützen, die im Automatikmodus ausgeführt werden. Sie umfasst Berechtigungen für verschiedene Vorgänge, darunter die Bereitstellung, das Anfügen, die Größenänderung und die Erstellung von Snapshots von Blockspeicher-Volumes.

Amazon EKS erstellt automatisch einen Zugriffseintrag mit dieser Zugriffsrichtlinie für die Cluster-IAM-Rolle, wenn der Automatikmodus aktiviert ist. Dadurch wird sichergestellt, dass die erforderlichen Berechtigungen vorhanden sind, damit die Blockspeicherfunktion ordnungsgemäß funktioniert.

Amazon EKSCompute ClusterPolicy

> **Note**
>
> Diese Richtlinie ist nur für AWS dienstbezogene Rollen vorgesehen und kann nicht für vom Kunden verwaltete Rollen verwendet werden.

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSComputeClusterPolicy`

Diese Richtlinie gewährt die erforderlichen Berechtigungen für die Rechenmanagement-Funktionen von Amazon EKS Auto Mode. Sie ermöglicht eine effiziente Orchestrierung und Skalierung von Rechenressourcen innerhalb von Amazon-EKS-Clustern Die Richtlinie umfasst folgende Berechtigungen:

Knotenverwaltung:

- Den Status von und erstellen, lesen, aktualisieren, löschen und verwalten. NodePools NodeClaims
- Verwalten NodeClasses, einschließlich Erstellung, Änderung und Löschung.

Terminplanung und Ressourcenmanagement:

- Leserechte für Pods, Knoten, persistente Volumes, persistente Volume-Ansprüche, Replikationscontroller und Namespaces.
- Lesezugriff auf Speicherklassen, CSI-Knoten und Volume-Anhänge.
- Auflistung und Überwachung von Bereitstellungen, Daemon-Sets, Replica-Sets und Stateful-Sets.
- Lesen Sie die Budgets für Pod-Unterbrechungen.

Ereignisbehandlung:

- Erstellen, Lesen und Verwalten von Cluster-Ereignissen.

Entzug der Berechtigungen für Knoten und Pod-Bereinigung:

- Aktualisieren, patchen und löschen Sie Knoten.
- Erstellen Sie Pod-Bereinigungen und löschen Sie Pods bei Bedarf.

Verwaltung benutzerdefinierter Ressourcendefinitionen (CRD):

- Neues erstellen CRDs.
- Spezifische Funktionen CRDs im Zusammenhang mit der Knotenverwaltung verwalten (NodeClasses NodePools NodeClaims,, und NodeDiagnostics).

Diese Richtlinie wurde entwickelt, um eine umfassende Datenverarbeitungsverwaltung innerhalb von Amazon-EKS-Clustern zu unterstützen, die im Automatikmodus ausgeführt werden. Es vereint Berechtigungen für verschiedene Vorgänge, darunter die Bereitstellung von Knoten, die Planung, die Skalierung und die Ressourcenoptimierung.

Amazon EKS erstellt automatisch einen Zugriffseintrag mit dieser Zugriffsrichtlinie für die Cluster-IAM-Rolle, wenn der Automatikmodus aktiviert ist. Dadurch wird sichergestellt, dass die erforderlichen

Berechtigungen vorhanden sind, damit die Datenverarbeitungsverwaltungsfunktion ordnungsgemäß funktioniert.

Amazon EKSLoad BalancingClusterPolicy

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSLoadBalancingClusterPolicy`

Diese Richtlinie gewährt die erforderlichen Berechtigungen für die Load-Balancing-Funktion von Amazon EKS Auto Mode. Es ermöglicht die effiziente Verwaltung und Konfiguration von Load-Balancing-Ressourcen innerhalb von Amazon-EKS-Clustern. Die Richtlinie umfasst folgende Berechtigungen:

Ereignis- und Ressourcenmanagement:

- Erstellen und Patchen von Ereignissen.
- Lesezugriff auf Pods, Knoten, Endpunkte und Namespaces.
- Aktualisieren des Pod-Status.

Service- und Ingress-Management:

- Vollständige Verwaltung von Services und deren Status.
- Umfassende Kontrolle über Ingresses und deren Status.
- Lesezugriff auf Endpunkt-Slices und Ingress-Klassen.

Zielgruppenbindungen:

- Erstellen und Ändern von Zielgruppenbindungen und deren Status.
- Lesezugriff auf Ingress-Klassenparameter.

Verwaltung benutzerdefinierter Ressourcendefinitionen (CRD):

- Alles erstellen und lesen CRDs.
- Spezifische Verwaltung von targetgroupbindings.eks.amazonaws.com und ingressclassparams.eks.amazonaws.com. CRDs

Webhook-Konfiguration:

- Erstellen und Lesen von sich ändernden und validierenden Webhook-Konfigurationen.
- eks-load-balancing-webhookVerwalte die Konfiguration.

Diese Richtlinie wurde entwickelt, um ein umfassendes Lastenausgleichsmanagement innerhalb von Amazon-EKS-Clustern zu unterstützen, die im Automatikmodus ausgeführt werden. Es kombiniert Berechtigungen für verschiedene Operationen, darunter die Bereitstellung von Diensten, das Routing eingehender Zugriffe und die Integration mit AWS Lastenausgleichsdiensten.

Amazon EKS erstellt automatisch einen Zugriffseintrag mit dieser Zugriffsrichtlinie für die Cluster-IAM-Rolle, wenn der Automatikmodus aktiviert ist. Dadurch wird sichergestellt, dass die erforderlichen Berechtigungen vorhanden sind, damit die Load-Balancing-Funktion ordnungsgemäß funktioniert.

Amazon EKSNetworking ClusterPolicy

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSNetworkingClusterPolicy`

Amazon EKSNetworking ClusterPolicy

Diese Richtlinie gewährt die erforderlichen Berechtigungen für die Netzwerkfunktionen von Amazon EKS Auto Mode. Sie ermöglicht die effiziente Verwaltung und Konfiguration von Netzwerkressourcen innerhalb von Amazon-EKS-Clustern. Die Richtlinie umfasst folgende Berechtigungen:

Knoten- und Pod-Verwaltung:

- Lesezugriff auf NodeClasses und deren Status.
- Lesezugriff auf NodeClaims und deren Status.
- Lesezugriff auf Pods.

CNI-Knotenverwaltung:

- Berechtigungen für CNINodes und deren Status, einschließlich Erstellen, Lesen, Aktualisieren, Löschen und Patchen.

Verwaltung benutzerdefinierter Ressourcendefinitionen (CRD):

- Alles CRDs erstellen und lesen.

- Spezifische Verwaltung (Aktualisieren, Patchen, Löschen) des CRD cninodes.eks.amazonaws.com.

Ereignis-Management:

- Erstellen und Patchen von Ereignissen.

Diese Richtlinie wurde entwickelt, um ein umfassendes Netzwerkmanagement innerhalb von Amazon-EKS-Clustern zu unterstützen, die im Automatikmodus ausgeführt werden. Sie kombiniert Berechtigungen für verschiedene Vorgänge, einschließlich Knoten-Netzwerkkonfiguration, CNI-Verwaltung (Container Network Interface) und zugehörige benutzerdefinierte Ressourcenverwaltung.

Die Richtlinie ermöglicht es den Netzwerkkomponenten, mit knotenbezogenen Ressourcen zu interagieren, CNI-spezifische Knotenkonfigurationen zu verwalten und benutzerdefinierte Ressourcen zu verarbeiten, die für den Netzwerkbetrieb im Cluster von entscheidender Bedeutung sind.

Amazon EKS erstellt automatisch einen Zugriffseintrag mit dieser Zugriffsrichtlinie für die Cluster-IAM-Rolle, wenn der Automatikmodus aktiviert ist. Dadurch wird sichergestellt, dass die erforderlichen Berechtigungen vorhanden sind, damit die Netzwerkfunktion ordnungsgemäß funktioniert.

EKSHybridAmazon-Richtlinie

> **Note**
>
> Diese Richtlinie ist nur für AWS dienstbezogene Rollen vorgesehen und kann nicht für vom Kunden verwaltete Rollen verwendet werden.

Diese Zugriffsrichtlinie umfasst Berechtigungen, die EKS Zugriff auf die Knoten eines Clusters gewähren. Wenn dies einem Zugriffseintrag zugeordnet ist, umfasst der Zugriffsbereich in der Regel keinen Kubernetes-Namespace, sondern den Cluster. Diese Richtlinie wird von Amazon-EKS-Hybridknoten verwendet.

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSHybridPolicy`

Kubernetes-API-Gruppen	Kubernetes (keine Ressource) URLs	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
*		nodes	list

Amazon EKSCluster InsightsPolicy

> **Note**
>
> Diese Richtlinie ist nur für AWS dienstbezogene Rollen vorgesehen und kann nicht für vom Kunden verwaltete Rollen verwendet werden.

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSClusterInsightsPolicy`

Diese Richtlinie gewährt Lesezugriffsberechtigungen für Erkenntnisfunktionalität von Amazon-EKS-Cluster Die Richtlinie umfasst folgende Berechtigungen:

Knoten-Zugriff: – Cluster-Knoten auflisten und anzeigen – Knotenstatusinformationen lesen

DaemonSet Zugriff: — Lesezugriff auf die Kube-Proxy-Konfiguration

Diese Richtlinie wird automatisch vom EKS-Service für Cluster-Ereignisse verwaltet. Weitere Informationen finden Sie unter the section called “Cluster-Einblicke”.

AWSBackupFullAccessPolicyForBackup

ARN – `arn:aws: eks::aws:cluster-access-policy/AWSBackupFullAccessPolicyForBackup`

AWSBackupFullAccessPolicyForBackup

Diese Richtlinie gewährt die Berechtigungen, die AWS Backup benötigt, um Backups des EKS-Clusters zu verwalten und zu erstellen. Diese Richtlinie umfasst die folgenden Berechtigungen:

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
*	*	list, get

AWSBackupFullAccessPolicyForRestore

ARN – `arn:aws: eks::aws:cluster-access-policy/AWSBackupFullAccessPolicyForRestore`

AWSBackupFullAccessPolicyForRestore

Diese Richtlinie gewährt die Berechtigungen, die AWS Backup benötigt, um Backups des EKS-Clusters zu verwalten und wiederherzustellen. Diese Richtlinie umfasst die folgenden Berechtigungen:

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
*	*	`list`, `get`, `create`

Amazon EKSACKPolicy

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSACKPolicy`

Diese Richtlinie gewährt die erforderlichen Berechtigungen, damit die Funktionen von AWS Controllers for Kubernetes (ACK) AWS Ressourcen von Kubernetes verwalten können. Die Richtlinie umfasst folgende Berechtigungen:

Benutzerdefiniertes ACK-Ressourcenmanagement:

- Voller Zugriff auf alle benutzerdefinierten Ressourcen des ACK-Dienstes für mehr als 50 AWS Dienste, darunter S3, RDS, DynamoDB, Lambda und mehr. EC2
- Benutzerdefinierte ACK-Ressourcendefinitionen erstellen, lesen, aktualisieren und löschen.

Namespace-Zugriff:

- Lesezugriff auf Namespaces für die Ressourcenorganisation.

Wahl des Vorsitzenden:

- Erstellen und lesen Sie Koordinationsverträge für die Wahl des Vorsitzenden.
- Aktualisieren und löschen Sie bestimmte ACK Service Controller-Leases.

Ereignis-Management:

- Ereignisse für ACK-Operationen erstellen und patchen.

Diese Richtlinie wurde entwickelt, um ein umfassendes AWS Ressourcenmanagement über APIs Kubernetes zu unterstützen. Amazon EKS erstellt automatisch einen Zugriffseintrag mit dieser Zugriffsrichtlinie für die Capability IAM-Rolle, die Sie bei der Erstellung der ACK-Fähigkeit angeben.

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
	`namespaces`	`get`, `watch`, `list`
`services.k8s.aws` , `acm.services.k8s.aws` , `acmpca.services.k8s.aws` , `apigateway.services.k8s.aws` , `apigatewayv2.services.k8s.aws` , `applicationautoscaling.services.k8s.aws` , `athena.services.k8s.aws` , `bedrock.services.k8s.aws` , `bedrockagent.services.k8s.aws` , `bedrockagentcorecontrol.services.k8s.aws` , `cloudfront.services.k8s.aws` , `cloudtrail.services.k8s.aws` , `cloudwatch.services.k8s.aws` , `cloudwatchlogs.services.k8s.aws` , `codeartif`	*	*

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
act.services.k8s.aws , cognitoidentityprovider.services.k8s.aws , documentdb.services.k8s.aws , dynamodb.services.k8s.aws , ec2.services.k8s.aws , ecr.services.k8s.aws , ecrpublic.services.k8s.aws , ecs.services.k8s.aws , efs.services.k8s.aws , eks.services.k8s.aws , elasticache.services.k8s.aws , elbv2.services.k8s.aws , emrcontainers.services.k8s.aws , eventbridge.services.k8s.aws , iam.services.k8s.aws , kafka.services.k8s.aws , keyspaces.services.k8s.aws , kinesis.services.k8s.aws , kms.services.k8s.aws , lambda.services.k8s.aws , memorydb.services.k8s.aws , mq.services.k8s.aws , networkfirewall.se		

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
rvices.k8s.aws , opensearchservice.services.k8s.aws , organizations.services.k8s.aws , pipes.services.k8s.aws , prometheusservice.services.k8s.aws , ram.services.k8s.aws , rds.services.k8s.aws , recyclebin.services.k8s.aws , route53.services.k8s.aws , route53resolver.services.k8s.aws , s3.services.k8s.aws , s3control.services.k8s.aws , sagemaker.services.k8s.aws , secretsmanager.services.k8s.aws , ses.services.k8s.aws , sfn.services.k8s.aws , sns.services.k8s.aws , sqs.services.k8s.aws , ssm.services.k8s.aws , wafv2.services.k8s.aws		
coordination.k8s.io	leases	create, get, list, watch

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
`coordination.k8s.io`	`leases`(nur für bestimmte ACK-Service-Controller-Leases)	`delete`, `update`, `patch`
	`events`	`create`, `patch`
`apiextensions.k8s.io`	`customresourcedefinitions`	`*`

EKSArgoCDClusterAmazon-Richtlinie

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSArgoCDClusterPolicy`

Diese Richtlinie gewährt Berechtigungen auf Clusterebene, die für die Argo-CD-Funktion erforderlich sind, um Ressourcen zu erkennen und Objekte im Clusterbereich zu verwalten. Die Richtlinie umfasst folgende Berechtigungen:

Namespace-Verwaltung:

- Namespaces für die Namespace-Verwaltung von Anwendungen erstellen, lesen, aktualisieren und löschen.

Verwaltung benutzerdefinierter Ressourcendefinitionen:

- Argo CD-spezifisch verwalten CRDs (Anwendungen, AppProjects, ApplicationSets).

API-Erkennung:

- Lesezugriff auf Kubernetes-API-Endpunkte zur Ressourcenerkennung.

Diese Richtlinie wurde entwickelt, um Argo-CD-Operationen auf Clusterebene zu unterstützen, einschließlich Namespace-Verwaltung und CRD-Installation. Amazon EKS erstellt automatisch einen Zugriffseintrag mit dieser Zugriffsrichtlinie für die Capability IAM-Rolle, die Sie bei der Erstellung der Argo-CD-Funktion angeben.

Kubernetes-API-Gruppen	Kubernetes, keine Ressource URLs	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
		`namespaces`	`create`, `get`, `update`, `patch`, `delete`
`apiextensions.k8s.io`		`customresourcedefinitions`	`create`
`apiextensions.k8s.io`		`customresourcedefinitions` (Nur Argo-CD) CRDs	`get`, `update`, `patch`, `delete`
	`/api`, `/api/*`, `/apis`, `/apis/*`		`get`

Amazon EKSArgo CDPolicy

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSArgoCDPolicy`

Diese Richtlinie gewährt Berechtigungen auf Namespace-Ebene, die für die Argo-CD-Funktion zur Bereitstellung und Verwaltung von Anwendungen erforderlich sind. Die Richtlinie umfasst folgende Berechtigungen:

Geheime Verwaltung:

- Voller Zugriff auf Geheimnisse für Git-Anmeldeinformationen und Clustergeheimnisse.

ConfigMap Zugriff:

- Lesezugriff auf, um Warnungen ConfigMaps zu senden, wenn Kunden versuchen, Argo CD zu verwenden, die nicht unterstützt wird. ConfigMaps

Ereignis-Management:

- Lesen und erstellen Sie Ereignisse für die Überwachung des Anwendungslebenszyklus.

Argo CD Ressourcenmanagement:

- Voller Zugriff auf Anwendungen ApplicationSets, und AppProjects.
- Verwalten Sie die Finalizer und den Status der Argo-CD-Ressourcen.

Diese Richtlinie wurde entwickelt, um Argo-CD-Operationen auf Namespace-Ebene zu unterstützen, einschließlich der Anwendungsbereitstellung und -verwaltung. Amazon EKS erstellt automatisch einen Zugriffseintrag mit dieser Zugriffsrichtlinie für die Capability IAM-Rolle, die Sie angeben, wenn die Argo-CD-Funktion erstellt wird, und zwar für den Argo-CD-Namespace.

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
	`secrets`	`*`
	`configmaps`	`get`, `list`, `watch`
	`events`	`get`, `list`, `watch`, `patch`, `create`
`argoproj.io`	`applications` , `applications/finalizers` , `applications/status` , `applicationsets` , `applicationsets/finalizers` , `applicationsets/status` , `appprojects` , `appprojects/finalizers` , `appprojects/status`	`*`

Amazon EKSKROPolicy

ARN – `arn:aws: eks::aws:cluster-access-policy/AmazonEKSKROPolicy`

Diese Richtlinie gewährt Berechtigungen, die für die Kro-Funktion (Kube Resource Orchestrator) erforderlich sind, um benutzerdefinierte Kubernetes zu erstellen und zu verwalten. APIs Die Richtlinie umfasst folgende Berechtigungen:

Kro-Ressourcenverwaltung:

- Voller Zugriff auf alle Kro-Ressourcen, einschließlich ResourceGraphDefinitions benutzerdefinierter Ressourceninstanzen.

Verwaltung benutzerdefinierter Ressourcendefinitionen:

- APIs Benutzerdefiniert von erstellen, lesen, aktualisieren und löschen CRDs ResourceGraphDefinitions.

Wahl des Vorsitzenden:

- Erstellen und lesen Sie Koordinationsverträge für die Wahl des Vorsitzenden.
- Aktualisieren und löschen Sie den Kro-Controller-Lease.

Ereignis-Management:

- Ereignisse für Kro-Operationen erstellen und patchen.

Diese Richtlinie wurde entwickelt, um eine umfassende Ressourcenzusammenstellung und ein benutzerdefiniertes API-Management über Kro zu unterstützen. Amazon EKS erstellt automatisch einen Zugriffseintrag mit dieser Zugriffsrichtlinie für die Capability IAM-Rolle, die Sie bei der Erstellung der Kro-Fähigkeit angeben.

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
`kro.run`	`*`	`*`
`apiextensions.k8s.io`	`customresourcedefinitions`	`*`
`coordination.k8s.io`	`leases`	`create`, `get`, `list`, `watch`

Kubernetes-API-Gruppen	Kubernetes-Ressourcen	Kubernetes-Verben (Berechtigungen)
`coordination.k8s.io`	`leases`(Nur Kro-Controller-Lease)	`delete`, `update`, `patch`
	`events`	`create`, `patch`

Aktualisierungen für Zugriffsrichtlinien

Sehen Sie sich an, welche Aktualisierungen für Zugriffsrichtlinien seit ihrer Einführung vorgenommen wurden. Abonnieren Sie den RSS-Feed in Dokumentverlauf, um automatische Benachrichtigungen über Änderungen an dieser Seite zu erhalten.

Änderungen	Beschreibung	Date
Fügen Sie Richtlinien für EKS-Funktionen hinzu	Veröffentlichen`AmazonEKSACKPolicy` , `AmazonEKSArgoCDClusterPolicy` `AmazonEKSArgoCDPolicy` , und `AmazonEKSKROPolicy` für die Verwaltung von EKS-Funktionen	22. November 2025
Add `AmazonEKSSecretReaderPolicy`	Fügen Sie eine neue Richtlinie für den schreibgeschützten Zugriff auf geheime Daten hinzu	6. November 2025
Richtlinie für EKS-Cluster-Erkenntnisse hinzufügen	`AmazonEKSClusterInsightsPolicy` veröffentlichen	2. Dezember 2024
Richtlinien für Amazon EKS Hybrid hinzufügen	`AmazonEKSHybridPolicy` veröffentlichen	2. Dezember 2024

Änderungen	Beschreibung	Date
Richtlinien für Amazon EKS Auto Mode hinzufügen	Diese Zugriffsrichtlinien gewähren der Cluster-I AM-Rolle und der Node-IAM-Rolle die Erlaubnis, Kubernetes aufzurufen. APIs AWS verwendet diese, um Routineaufgaben für Speicher-, Rechen- und Netzwerkr essourcen zu automatisieren.	2. Dezember 2024
Add `AmazonEKSAdminViewPolicy`	Fügen Sie eine neue Richtlini e für erweiterten Zugriff auf Ansichten hinzu, einschließlich Ressourcen wie Geheimnisse.	23. April 2024
Zugriffsrichtlinien wurden eingeführt.	In Amazon EKS wurden Zugriffsrichtlinien eingeführt.	29. Mai 2023

Ändern des Authentifizierungsmodus, um Zugriffseinträge zu verwenden

Wenn Sie Zugriffseinträge verwenden möchten, müssen Sie zunächst den Authentifizierungsmodus des Clusters in `API_AND_CONFIG_MAP` oder `API` ändern. Dadurch wird die API für Zugriffseinträge hinzugefügt.

AWS-Konsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie den Namen des Clusters aus, in dem Sie einen Zugriffseintrag erstellen möchten.
3. Wählen Sie die Registerkarte Zugriff aus.
4. Der Authentifizierungsmodus zeigt den aktuellen Authentifizierungsmodus des Clusters. Wenn der Modus EKS-API lautet, können Sie bereits Zugriffseinträge hinzufügen und die restlichen Schritte überspringen.
5. Wählen Sie Zugriff verwalten aus.

6. Wählen Sie für den Cluster-Authentifizierungsmodus einen Modus mit der EKS-API aus. Beachten Sie, dass der Authentifizierungsmodus nicht mehr in einen Modus geändert werden kann, in dem die EKS-API und die Zugriffseinträge entfernt werden.
7. Wählen Sie Änderungen speichern aus. Amazon EKS beginnt mit der Aktualisierung des Clusters, der Status des Clusters ändert sich in „Wird aktualisiert“ und die Änderung wird auf der Registerkarte Aktualisierungsverlauf erfasst.
8. Warten Sie, bis der Status des Clusters wieder zu „Aktiv“ wechselt. Wenn der Cluster aktiv ist, können Sie die Schritte unter the section called “Zugriffseinträge erstellen” befolgen, um IAM-Prinzipalen Zugriff auf den Cluster zu gewähren.

AWS-CLI

1. Installieren Sie die AWS-CLI wie unter Installation im Benutzerhandbuch zur AWS-Befehlszeilenschnittstelle beschrieben.
2. Führen Sie den folgenden Befehl aus. Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters. Wenn Sie die `ConfigMap`-Methode dauerhaft deaktivieren möchten, ersetzen Sie `API_AND_CONFIG_MAP` durch `API`.

   Amazon EKS beginnt mit der Aktualisierung des Clusters, der Status des Clusters ändert sich in „WIRD AKTUALISIERT“ und die Änderung wird in der aws eks list-updates aufgezeichnet.

   ```
   aws eks update-cluster-config --name my-cluster --access-config
    authenticationMode=API_AND_CONFIG_MAP
   ```

3. Warten Sie, bis der Status des Clusters wieder zu „Aktiv“ wechselt. Wenn der Cluster aktiv ist, können Sie die Schritte unter the section called “Zugriffseinträge erstellen” befolgen, um IAM-Prinzipalen Zugriff auf den Cluster zu gewähren.

Erforderliche Plattformversion

Um Zugriffseinträge verwenden zu können, muss der Cluster über eine Plattformversion verfügen, die mit der in der folgenden Tabelle aufgeführten Version identisch oder höher ist, oder über eine Kubernetes-Version, die höher ist als die in der Tabelle aufgeführten Versionen. Wenn Ihre Kubernetes-Version nicht aufgeführt ist, unterstützen alle Plattformversionen Zugriffseinträge.

Kubernetes-Version	Plattformversion
Nicht aufgelistet	Alle unterstützt
1.30	eks.2
1.29	eks.1
1.28	eks.6

Weitere Informationen finden Sie unter platform-versions.

Zugriffseinträge erstellen

Berücksichtigen Sie Folgendes, bevor Sie Zugriffseinträge erstellen:

- Ein ordnungsgemäß eingerichteter Authentifizierungsmodus. Siehe the section called "Authentifizierungsmodus".
- Ein Zugriffseintrag enthält den Amazon-Ressourcennamen (ARN) genau eines vorhandenen IAM-Prinzipals. Ein IAM-Prinzipal kann nicht in mehreren Zugriffseinträgen enthalten sein. Zusätzliche Überlegungen zu dem von Ihnen angegebenen ARN:
  - In den bewährten Methoden für IAM wird empfohlen, für den Zugriff auf Ihren Cluster IAM-Rollen mit kurzfristigen Anmeldeinformationen zu verwenden anstatt IAM-Benutzer mit langfristigen Anmeldeinformationen. Weitere Informationen finden Sie unter Menschliche Benutzer auffordern, den Verbund mit einem Identitätsanbieter zu verwenden, um mit temporären Anmeldeinformationen auf AWS zuzugreifen im IAM-Benutzerhandbuch.
  - Ein ARN für eine IAM-Rolle kann einen Pfad enthalten. ARNs in `aws-auth ConfigMap`-Einträgen können keinen Pfad enthalten. Ihr ARN kann beispielsweise `arn:aws:iam::<111122223333>:role/<development/apps/my-role>` oder `arn:aws:iam::<111122223333>:role/<my-role>` lauten.
  - Wenn der Typ des Zugriffseintrags nicht `STANDARD` ist (siehe nächste Überlegung zu Typen), muss sich der ARN im gleichen AWS-Region befinden wie Ihr Cluster. Wenn der Typ `STANDARD` ist, kann sich der ARN im gleichen oder in einem anderen AWS-Region befinden wie der Cluster.
  - Nach der Erstellung des Zugriffseintrags kann der IAM-Prinzipal nicht mehr geändert werden.
  - Sollten Sie den IAM-Prinzipal mit diesem ARN löschen, wird der Zugriffseintrag nicht automatisch gelöscht. Es empfiehlt sich, den Zugriffseintrag mit einem ARN für einen gelöschten IAM-

Prinzipal zu löschen. Wenn Sie den Zugriffseintrag nicht löschen und den IAM-Prinzipal erneut erstellen, funktioniert der Zugriffseintrag nicht, selbst wenn er dieselbe ARN aufweist. Dies liegt daran, dass, obwohl die ARN für den neu erstellten IAM-Prinzipal dieselbe ist, die `roleID` oder `userID` (Sie können dies mit dem AWS-CLI-Befehl `aws sts get-caller-identity` sehen) für den neu erstellten IAM-Prinzipal anders ist als für den ursprünglichen IAM-Prinzipal. Auch wenn Sie das `roleID` oder `userID` des IAM-Prinzipals für einen Zugriffseintrag nicht sehen, speichert Amazon EKS es mit dem Zugriffseintrag.

- Jeder Zugriffseintrag hat einen Typ. Der Typ des Zugriffseintrags hängt vom Typ der Ressource ab, der er zugeordnet ist, und definiert nicht die Berechtigungen. Wenn Sie keinen Typ angeben, wird er von Amazon EKS automatisch auf `STANDARD` festgelegt
  - `EC2_LINUX` – Für eine IAM-Rolle, die mit selbstverwalteten Linux- oder Bottlerocket-Knoten verwendet wird
  - `EC2_WINDOWS` – Für eine IAM-Rolle, die mit selbstverwalteten Windows-Knoten verwendet wird
  - `FARGATE_LINUX` – Für eine mit AWS Fargate verwendete IAM-Rolle (Fargate)
  - `HYBRID_LINUX` – Für eine IAM-Rolle, die mit Hybridknoten verwendet wird
  - `STANDARD` – Standardtyp, falls keiner angegeben ist
  - `EC2` – Für benutzerdefinierte Knotenklassen im EKS Auto Mode. Weitere Informationen finden Sie unter the section called “Knotenklassen-Zugriffseintrag erstellen”.
  - Nach der Erstellung des Zugriffseintrags ist es nicht möglich, den Typ zu ändern.
- Es ist nicht erforderlich, einen Zugriffseintrag für eine IAM-Rolle zu erstellen, die für eine verwaltete Knotengruppe oder ein Fargate-Profil verwendet wird. EKS erstellt Zugriffseinträge (sofern aktiviert) oder aktualisiert die Konfigurationszuordnung der Authentifizierung (sofern keine Zugriffseinträge verfügbar sind).
- Wenn der Typ des Zugriffseintrags `STANDARD` lautet, können Sie einen Benutzernamen für den Zugriffseintrag angeben. Wenn Sie keinen Wert für den Benutzernamen angeben, legt Amazon EKS einen der folgenden Werte für Sie fest, abhängig vom Typ des Zugriffseintrags und davon, ob es sich bei dem von Ihnen angegebenen IAM-Prinzipal um eine IAM-Rolle oder um einen IAM-Benutzer handelt. Sofern Sie keinen spezifischen Grund für die Angabe eines eigenen Benutzernamens haben, empfehlen wir, keinen anzugeben und ihn automatisch von Amazon EKS generieren zu lassen. Beachten Sie bei Angabe eines eigenen Benutzernamens Folgendes:
  - Er darf nicht mit `system:`, `eks:`, `aws:`, `amazon:` oder `iam:` beginnen.
  - Bei einem Benutzername für eine IAM-Rolle empfiehlt es sich, am Ende des Benutzernamens `{{SessionName}}` oder `{{SessionNameRaw}}` hinzuzufügen. Wenn Sie Ihrem Benutzernamen `{{SessionName}}` oder `{{SessionNameRaw}}` hinzufügen, muss

der Benutzername vor {{SessionName}} einen Doppelpunkt enthalten. Wenn diese Rolle übernommen wird, wird der Name der AWS-STS-Sitzung, der bei der Übernahme der Rolle angegeben wurde, automatisch an den Cluster übergeben und in den CloudTrail-Protokollen angezeigt. Der Benutzername `john{{SessionName}}` ist beispielsweise nicht zulässig. Er müsste `:john{{SessionName}}` oder `jo:hn{{SessionName}}` lauten. Der Doppelpunkt muss sich nur vor `{{SessionName}}` befinden. Der von Amazon EKS generierte Benutzername in der folgenden Tabelle enthält einen ARN. Da ein ARN Doppelpunkte enthält, erfüllt er diese Anforderung. Der Doppelpunkt ist nicht erforderlich, wenn Sie `{{SessionName}}` nicht in Ihren Benutzernamen einschließen. Beachten Sie, dass in `{{SessionName}}` das Sonderzeichen „@“ im Sitzungsnamen durch „-“ ersetzt wird. `{{SessionNameRaw}}` behält alle Sonderzeichen im Sitzungsnamen bei.

IAM-Prinzipaltyp	Typ	Von Amazon EKS automatisch festgelegter Wert für den Benutzernamen
Benutzer	`STANDARD`	Der ARN des Benutzers. Beispiel: `arn:aws:iam::<111122223333>:user/<my-user>`

IAM-Prinzipaltyp	Typ	Von Amazon EKS automatisch festgelegter Wert für den Benutzernamen
Rolle	`STANDARD`	Der STS-ARN der Rolle, wenn sie angenommen wird. Amazon EKS fügt am Ende der Rolle `{{SessionName}}` hinzu. Beispiel: `arn:aws:sts::<111122223333>:assumed-role/<my-role>/{{SessionName}}` Wenn der ARN der von Ihnen angegebenen Rolle einen Pfad enthalten hat, wird dieser von Amazon EKS im generierten Benutzernamen entfernt.
Rolle	`EC2_LINUX` oder `EC2_Windows`	`system:node:{{EC2PrivateDNSName}}`
Rolle	`FARGATE_LINUX`	`system:node:{{SessionName}}`
Rolle	`HYBRID_LINUX`	`system:node:{{SessionName}}`

Sie können den Benutzernamen ändern, nachdem der Zugriffseintrag erstellt wurde.

- Wenn der Typ eines Zugriffseintrags `STANDARD` ist und Sie die Kubernetes-RBAC-Autorisierung verwenden möchten, können Sie dem Zugriffseintrag einen oder mehrere Gruppennamen hinzufügen. Gruppennamen können nach der Erstellung eines Zugriffseintrags hinzugefügt und entfernt werden. Damit der IAM-Prinzipal Zugriff auf Kubernetes-Objekte in Ihrem Cluster

hat, müssen Sie rollenbasierte Autorisierungsobjekte (RBAC) für Kubernetes erstellen und verwalten. Erstellen Sie Kubernetes `RoleBinding` oder `ClusterRoleBinding`-Objekte in Ihrem Cluster, die den Gruppennamen als `subject` für `kind: Group` angeben. Kubernetes autorisiert den IAM-Prinzipalzugriff auf alle Cluster-Objekte, die Sie in einem Kubernetes `Role`- oder `ClusterRole`-Objekt angegeben haben, das Sie auch in der `roleRef` Ihrer Bindung angegeben haben. Wenn Sie Gruppennamen angeben, empfehlen wir Ihnen, sich mit den rollenbasierten Autorisierungsobjekten (RBAC) von Kubernetes vertraut zu machen. Weitere Informationen finden Sie unter Using RBAC Authorization in der Kubernetes-Dokumentation.

> **Important**
>
> Amazon EKS überprüft nicht, ob die in Ihrem Cluster vorhandenen Kubernetes-RBAC-Objekte die von Ihnen angegebenen Gruppennamen enthalten. Wenn Sie beispielsweise einen Zugriffseintrag für eine Gruppe erstellen, die derzeit nicht vorhanden ist, erstellt EKS die Gruppe, anstatt eine Fehlermeldung auszugeben.

Anstelle oder zusätzlich zur Autorisierung des IAM-Prinzips für den Zugriff auf Kubernetes-Objekte in Ihrem Cluster durch Kubernetes können Sie Amazon-EKS-Zugriffsrichtlinien einem Zugriffseintrag zuordnen. Amazon EKS autorisiert für IAM-Prinzipale den Zugriff auf Kubernetes-Objekte in Ihrem Cluster mit den in der Zugriffsrichtlinie festgelegten Berechtigungen. Sie können die Berechtigungen einer Zugriffsrichtlinie auf von Ihnen angegebene Kubernetes-Namespaces beschränken. Für die Verwendung von Zugriffsrichtlinien ist keine Verwaltung von Kubernetes RBAC-Objekten erforderlich. Weitere Informationen finden Sie unter the section called “Zuordnen von Zugriffsrichtlinien”.

- Wenn Sie einen Zugriffseintrag mit dem Typ `EC2_LINUX` oder `EC2_Windows` erstellen, muss der IAM-Prinzipal, der den Zugriffseintrag erstellt, über die Berechtigung `iam:PassRole` verfügen. Weitere Informationen finden Sie unter Erteilen von Berechtigungen, mit denen ein Benutzer eine Rolle an einen AWS-Service übergeben kann im IAM-Benutzerhandbuch.
- Ähnlich wie beim standardmäßigen IAM-Verhalten sind die Erstellung und Aktualisierung von Zugriffseinträgen letztlich konsistent und es kann mehrere Sekunden dauern, bis sie wirksam werden, nachdem der erste API-Aufruf erfolgreich abgeschlossen wurde. Sie müssen Ihre Anwendungen unter Berücksichtigung dieser potenziellen Verzögerungen konzipieren. Es empfiehlt sich, in die kritischen, hochverfügbaren Code-Pfade Ihrer Anwendung keine Erstellungen oder Aktualisierungen von Zugriffseinträgen einzuschließen. Nehmen Sie -Änderungen stattdessen in einer separaten Initialisierungs- oder Einrichtungsroutine vor, die seltener ausgeführt

wird. Vergewissern Sie sich auch, dass die Änderungen weitergegeben wurden, bevor die Produktionsarbeitsabläufe davon abhängen.

- Zugriffseinträge unterstützen keine serviceverknüpften Rollen. Sie können keine Zugriffseinträge erstellen, bei denen die Haupt-ARN eine serviceverknüpfte Rolle ist. Sie können serviceverknüpfte Rollen anhand ihrer ARN im Format `arn:aws:iam::*:role/aws-service-role/*` identifizieren.

Sie können einen Zugriffseintrag über die AWS-Managementkonsole oder mithilfe der AWS CLI erstellen.

AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie den Namen des Clusters aus, in dem Sie einen Zugriffseintrag erstellen möchten.
3. Wählen Sie die Registerkarte Zugriff aus.
4. Wählen Sie Zugriffseintrag erstellen aus.
5. Wählen Sie unter IAM-Prinzipal eine bereits vorhandene IAM-Rolle oder einen bereits vorhandenen IAM-Benutzer aus. In den bewährten Methoden für IAM wird empfohlen, für den Zugriff auf Ihren Cluster IAM-Rollen mit kurzfristigen Anmeldeinformationen zu verwenden anstatt IAM-Benutzer mit langfristigen Anmeldeinformationen. Weitere Informationen finden Sie unter Menschliche Benutzer auffordern, den Verbund mit einem Identitätsanbieter zu verwenden, um mit temporären Anmeldeinformationen auf AWS zuzugreifen im IAM-Benutzerhandbuch.
6. Wählen Sie unter Typ die Option EC2 Linux oder EC2 Windows aus, wenn sich der Zugriffseintrag auf die Knotenrolle bezieht, die für selbstverwaltete Amazon-EC2-Knoten verwendet wird. Übernehmen Sie andernfalls die Standardeinstellung (Standard).
7. Wenn Sie unter Typ die Option Standard ausgewählt haben, können Sie bei Bedarf unter Benutzername einen Benutzernamen angeben.
8. Wenn Sie unter Typ die Option Standard ausgewählt haben und die Kubernetes-RBAC-Autorisierung für den IAM-Prinzipal verwenden möchten, können Sie unter Gruppen einen oder mehrere Namen angeben. Wenn Sie keine Gruppennamen angeben und die Amazon-EKS-Autorisierung verwenden möchten, können Sie in einem späteren Schritt (oder nach der Erstellung des Zugriffseintrags) eine Zugriffsrichtlinie zuordnen.
9. (Optional) Weisen Sie dem Zugriffseintrag unter Tags Beschriftungen zu. So können Sie beispielsweise ganz einfach nach allen Ressourcen mit dem gleichen Tag suchen.
10. Wählen Sie Weiter aus.

11. Wenn Sie auf der Seite Zugriffsrichtlinie hinzufügen den Typ Standard ausgewählt haben und Amazon EKS dem IAM-Prinzipal Berechtigungen für die Kubernetes-Objekte in Ihrem Cluster gewähren soll, führen Sie die folgenden Schritte aus. Klicken Sie andernfalls auf Next (Weiter).
    a. Wählen Sie unter Richtlinienname eine Zugriffsrichtlinie aus. Sie können die Berechtigungen der Zugriffsrichtlinien nicht anzeigen, sie enthalten jedoch ähnliche Berechtigungen wie die benutzerorientierten `ClusterRole`-Objekte von Kubernetes. Weitere Informationen finden Sie unterBenutzerorientierte Rollen in der Kubernetes-Dokumentation.
    b. Wählen Sie eine der folgenden Optionen:
        - Cluster – Wählen Sie diese Option aus, wenn Amazon EKS den IAM-Prinzipal so autorisieren soll, dass er für alle Kubernetes-Objekte in Ihrem Cluster über die in der Zugriffsrichtlinie festgelegten Berechtigungen verfügt.
        - Kubernetes-Namespace – Wählen Sie diese Option aus, wenn Amazon EKS den IAM-Prinzipal so autorisieren soll, dass er für alle Kubernetes-Objekte in einem spezifischen Kubernetes-Namespace Ihres Clusters über die in der Zugriffsrichtlinie festgelegten Berechtigungen verfügt. Geben Sie unter Namespace den Namen des Kubernetes-Namespace in Ihrem Cluster ein. Wenn Sie zusätzliche Namespaces hinzufügen möchten, wählen Sie Neuen Namespace hinzufügen aus und geben Sie den Namespace-Namen ein.
    c. Wenn Sie weitere Richtlinien hinzufügen möchten, wählen Sie Richtlinie hinzufügen aus. Sie können für jede Richtlinie unterschiedliche Geltungsbereiche festlegen, aber jede Richtlinie kann nur einmal hinzugefügt werden.
    d. Wählen Sie Weiter aus.
12. Überprüfen Sie die Konfiguration für Ihren Zugriffseintrag. Sollten Sie einen Fehler entdecken, wählen Sie Zurück aus, um schrittweise zurück zu navigieren und den Fehler zu korrigieren. Ist die Konfiguration korrekt, wählen Sie Erstellen aus.

AWS-CLI

1. Installieren Sie die AWS-CLI wie unter Installation im Benutzerhandbuch zur AWS-Befehlszeilenschnittstelle beschrieben.
2. Um einen Zugriffseintrag zu erstellen, können Sie eines der folgenden Beispiele verwenden:
    - Erstellen Sie einen Zugriffseintrag für eine selbstverwaltete Amazon-EC2-Linux-Knotengruppe. Ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters, *`111122223333`* durch Ihre AWS-Konto-ID und *`EKS-my-cluster-self-managed-ng-1`* durch den Namen Ihrer Knoten-IAM-Rolle. Wenn es sich bei Ihrer Knotengruppe um eine Windows-Knotengruppe handelt, ersetzen Sie *`EC2_Linux`* durch `EC2_Windows`.

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/EKS-my-cluster-self-managed-ng-1 --type EC2_LINUX
```

Bei Angabe eines anderen Typs als `STANDARD` kann die Option `--kubernetes-groups` nicht verwendet werden. Sie können diesem Zugriffseintrag keine Zugriffsrichtlinie zuordnen, da der Typ nicht den Wert `STANDARD` hat.

- Erstellen Sie einen Zugriffseintrag, der eine IAM-Rolle zulässt, die nicht für eine selbstverwaltete Amazon-EC2-Knotengruppe verwendet wird, mit der Kubernetes den Zugriff auf Ihren Cluster autorisieren soll. Ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters, *`111122223333`* durch Ihre AWS-Konto-ID und *`my-role`* durch den Namen Ihrer IAM-Rolle. Ersetzen Sie *`Betrachter`* durch den Namen einer Gruppe, die Sie in Kubernetes `RoleBinding` oder einem `ClusterRoleBinding`-Objekt in Ihrem Cluster angegeben haben.

  ```
  aws eks create-access-entry --cluster-name my-cluster --principal-arn
   arn:aws:iam::111122223333:role/my-role --type STANDARD --user Viewers --
  kubernetes-groups Viewers
  ```

- Erstellen Sie einen Zugriffseintrag, der es einem IAM-Benutzer ermöglicht, sich bei Ihrem Cluster zu authentifizieren. Dieses Beispiel soll zeigen, dass es diese Möglichkeit gibt. In den bewährten Methoden für IAM wird empfohlen, für den Zugriff auf Ihren Cluster IAM-Rollen mit kurzfristigen Anmeldeinformationen zu verwenden anstatt IAM-Benutzer mit langfristigen Anmeldeinformationen. Weitere Informationen finden Sie unter Menschliche Benutzer auffordern, den Verbund mit einem Identitätsanbieter zu verwenden, um mit temporären Anmeldeinformationen auf AWS zuzugreifen im IAM-Benutzerhandbuch.

  ```
  aws eks create-access-entry --cluster-name my-cluster --principal-arn
   arn:aws:iam::111122223333:user/my-user --type STANDARD --username my-user
  ```

  Wenn dieser Benutzer mehr Zugriff auf Ihren Cluster haben soll als ihm durch die Berechtigungen in den Kubernetes-API-Erkennungsrollen gewährt wird, müssen Sie dem Zugriffseintrag eine Zugriffsrichtlinie zuordnen, da die Option `--kubernetes-groups` nicht verwendet wird. Weitere Informationen finden Sie unter the section called “Zuordnen von Zugriffsrichtlinien” und API-Erkennungsrollen in der Kubernetes-Dokumentation.

Zugriffseinträge aktualisieren

Sie können einen Zugriffseintrag über die AWS-Managementkonsole oder mithilfe der AWS-CLI aktualisieren.

AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie den Namen des Clusters aus, in dem Sie einen Zugriffseintrag erstellen möchten.
3. Wählen Sie die Registerkarte Zugriff aus.
4. Wählen Sie den Zugriffseintrag aus, den Sie aktualisieren möchten.
5. Wählen Sie Bearbeiten aus.
6. Unter Benutzername können Sie den vorhandenen Wert ändern.
7. Unter Gruppen können Sie Gruppennamen entfernen oder neue Gruppennamen hinzufügen. Die Gruppennamen system:nodes und system:bootstrappers dürfen nicht entfernt werden, sofern vorhanden. Wenn Sie diese Gruppen entfernen, funktioniert Ihr Cluster unter Umständen nicht mehr ordnungsgemäß. Wenn Sie keine Gruppennamen angeben und die Amazon-EKS-Autorisierung verwenden möchten, ordnen Sie in einem späteren Schritt eine Zugriffsrichtlinie zu.
8. Unter Tags können Sie dem Zugriffseintrag Beschriftungen zuweisen. So können Sie beispielsweise ganz einfach nach allen Ressourcen mit dem gleichen Tag suchen. Außerdem können Sie vorhandene Tags entfernen.
9. Wählen Sie Änderungen speichern aus.
10. Informationen zum Zuordnen einer Zugriffsrichtlinie zu dem Eintrag finden Sie unter the section called “Zuordnen von Zugriffsrichtlinien”.

AWS-CLI

1. Installieren Sie die AWS-CLI wie unter Installation im Benutzerhandbuch zur AWS-Befehlszeilenschnittstelle beschrieben.
2. Um einen Zugriffseintrag zu aktualisieren, ersetzen Sie *my-cluster* durch den Namen Ihres Clusters, *111122223333* durch Ihre AWS-Konto-ID und *EKS-my-cluster-my-namespace-Viewers* durch den Namen einer IAM-Rolle.

```
aws eks update-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/EKS-my-cluster-my-namespace-Viewers --kubernetes-
groups Viewers
```

Die Option `--kubernetes-groups` kann nur verwendet werden, wenn der Zugriffseintrag den Typ `STANDARD` hat. Auch kann eine Zugriffsrichtlinie nur einem Zugriffseintrag mit dem Typ `STANDARD` zugeordnet werden.

Zugriffseinträge löschen

Wenn Sie einen Zugriffseintrag versehentlich gelöscht haben, können Sie ihn jederzeit erneut erstellen. Wenn dem zu löschenden Zugriffseintrag Zugriffsrichtlinien zugeordnet sind, werden die Zuordnungen automatisch gelöscht. Es ist nicht erforderlich, die Zuordnung von Zugriffsrichtlinien zu einem Zugriffseintrag vor dem Löschen aufzuheben.

Sie können einen Zugriffseintrag über die AWS-Managementkonsole oder mithilfe der AWS-CLI löschen.

AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie den Namen des Clusters aus, aus dem Sie einen Zugriffseintrag löschen möchten.
3. Wählen Sie die Registerkarte Zugriff aus.
4. Wählen Sie in der Liste Zugriffseinträge den Zugriffseintrag aus, den Sie löschen möchten.
5. Wählen Sie Delete.
6. Wählen Sie im Bestätigungs-Dialogfeld die Option Delete (Löschen).

AWS-CLI

1. Installieren Sie die AWS-CLI wie unter Installation im Benutzerhandbuch zur AWS-Befehlszeilenschnittstelle beschrieben.
2. Um einen Zugriffseintrag zu löschen, ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters, *`111122223333`* durch Ihre AWS-ID und *`my-role`* durch den Namen der IAM-Rolle, die keinen Zugriff mehr auf Ihren Cluster haben soll.

```
aws eks delete-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role
```

Festlegen eines benutzerdefinierten Benutzernamens für EKS-Zugriffseinträge

Beim Erstellen von Zugriffseinträgen für Amazon EKS können Sie entweder den automatisch generierten Benutzernamen verwenden oder einen benutzerdefinierten Benutzernamen angeben. Auf dieser Seite werden beide Optionen erläutert und Sie werden durch die Einrichtung eines benutzerdefinierten Benutzernamens geführt.

Übersicht

Der Benutzername in einem Zugriffseintrag wird verwendet, um den IAM-Prinzipal in Kubernetes-Protokollen und Prüfpfaden zu identifizieren. Standardmäßig generiert Amazon EKS einen Benutzernamen basierend auf der ARN der IAM-Identität. Sie können bei Bedarf jedoch einen benutzerdefinierten Benutzernamen angeben.

Generierung des Standard-Benutzernamens

Wenn Sie keinen Wert für den Benutzernamen angeben, generiert Amazon EKS automatisch einen Benutzernamen basierend auf der IAM-Identität:

- Für IAM-Benutzer:
  - EKS legt den Kubernetes-Benutzernamen auf die ARN des IAM-Benutzers fest
  - Beispiel:

    ```
    {arn-aws}iam::<111122223333>:user/<my-user>
    ```
- Für IAM-Rollen:
  - EKS legt den Kubernetes-Benutzernamen basierend auf der ARN der IAM-Rolle fest
  - Der STS-ARN der Rolle, wenn sie angenommen wird. Amazon EKS fügt am Ende der Rolle `{{SessionName}}` hinzu. Wenn der ARN der von Ihnen angegebenen Rolle einen Pfad enthalten hat, wird dieser von Amazon EKS im generierten Benutzernamen entfernt.
  - Beispiel:

    ```
    {arn-aws}sts::<111122223333>:assumed-role/<my-role>/{{SessionName}}
    ```

Sofern Sie keinen bestimmten Grund haben, Ihren eigenen Benutzernamen anzugeben, empfehlen wir Ihnen, keinen anzugeben und ihn von Amazon EKS automatisch für Sie generieren zu lassen.

Festlegen eines benutzerdefinierten Benutzernamens

Beim Erstellen eines Zugriffseintrags können Sie mithilfe des `--username`-Parameters einen benutzerdefinierten Benutzernamen angeben:

```
aws eks create-access-entry --cluster-name <cluster-name> --principal-arn <iam-
identity-arn> --type STANDARD --username <custom-username>
```

Anforderungen für benutzerdefinierte Benutzernamen

Wenn Sie einen benutzerdefinierten Benutzernamen angeben:

- Der Benutzername darf nicht mit `system:`, `eks:`, `aws:`, `amazon:` oder `iam:` beginnen.
- Bei einem Benutzername für eine IAM-Rolle empfiehlt es sich, am Ende des Benutzernamens `{{SessionName}}` oder `{{SessionNameRaw}}` hinzuzufügen.
  - Wenn Sie Ihrem Benutzernamen `{{SessionName}}` oder `{{SessionNameRaw}}` hinzufügen, muss der Benutzername vor {{SessionName}} einen Doppelpunkt enthalten.

Erstellen Sie mithilfe einer Zugriffsrichtlinie und der AWS-CLI einen Zugriffseintrag für eine IAM-Rolle oder einen IAM-Benutzer

Erstellen Sie Amazon-EKS-Zugriffseinträge, die von AWS verwaltete EKS-Zugriffsrichtlinien verwenden, um IAM-Identitäten standardisierte Berechtigungen für den Zugriff auf und die Verwaltung von Kubernetes-Clustern zu gewähren.

Übersicht

Zugriffseinträge in Amazon EKS legen fest, wie IAM-Identitäten (Benutzer und Rollen) auf Ihre Kubernetes-Cluster zugreifen und mit diesen interagieren können. Durch die Erstellung von Zugriffseinträgen mit EKS-Zugriffsrichtlinien können Sie:

- Bestimmten IAM-Benutzern oder -Rollen die Berechtigung zum Zugriff auf Ihren EKS-Cluster gewähren
- Berechtigungen mithilfe von AWS-verwalteten EKS-Zugriffsrichtlinien steuern, die standardisierte, vordefinierte Berechtigungssätze bieten
- Berechtigungen auf bestimmte Namespaces oder clusterweit beschränken
- Zugriffsverwaltung vereinfachen, ohne die `aws-auth` ConfigMap zu ändern oder Kubernetes-RBAC-Ressourcen zu erstellen

- AWS-integrierten Ansatz für die Kubernetes-Zugriffskontrolle verwenden, der gängige Anwendungsfälle abdeckt und gleichzeitig bewährte Sicherheitsmethoden einhält

Dieser Ansatz wird für die meisten Anwendungsfälle empfohlen, da er von AWS verwaltete, standardisierte Berechtigungen bietet, ohne dass eine manuelle Kubernetes-RBAC-Konfiguration erforderlich ist. EKS-Zugriffsrichtlinien machen die manuelle Konfiguration von Kubernetes-RBAC-Ressourcen überflüssig und bieten vordefinierte Berechtigungssätze, die gängige Anwendungsfälle abdecken.

Voraussetzungen

- Der Authentifizierungsmodus Ihres Clusters muss konfiguriert sein, um Zugriffseinträge zu ermöglichen. Weitere Informationen finden Sie unter the section called “Authentifizierungsmodus”.
- Installieren und konfigurieren Sie die AWS CLI wie im Benutzerhandbuch zur AWS-Befehlszeilenschnittstelle unter Installieren beschrieben.

Schritt 1: Zugriffseintrag definieren

1. Suchen Sie die ARN der IAM-Identität, z. B. eines Benutzers oder einer Rolle, der Sie Berechtigungen gewähren möchten.
   - Jede IAM-Identität kann nur einen EKS-Zugriffseintrag haben.
2. Legen Sie fest, ob die Berechtigungen der Amazon-EKS-Zugriffsrichtlinie nur für einen bestimmten Kubernetes-Namespace oder für den gesamten Cluster gelten sollen.
   - Wenn Sie die Berechtigungen auf einen bestimmten Namespace beschränken möchten, notieren Sie sich den Namen des Namespace.
3. Wählen Sie die gewünschte EKS-Zugriffsrichtlinie für die IAM-Identität aus. Diese Richtlinie gewährt Berechtigungen innerhalb des Clusters. Notieren Sie sich die ARN der Richtlinie.
   - Eine Liste der Richtlinien finden Sie unter Verfügbare Zugriffsrichtlinien.
4. Prüfen Sie, ob der automatisch generierte Benutzername für den Zugriffseintrag geeignet ist oder ob Sie einen Benutzernamen manuell angeben müssen.
   - AWS generiert diesen Wert automatisch basierend auf der IAM-Identität. Sie können einen benutzerdefinierten Benutzernamen festlegen. Dieser wird in den Kubernetes-Protokollen angezeigt.
   - Weitere Informationen finden Sie unter the section called “Benutzerdefinierten Benutzernamen festlegen”.

Schritt 2: Zugriffseintrag erstellen

Nachdem Sie den Zugriffseintrag geplant haben, erstellen Sie ihn mithilfe der AWS-CLI.

Das folgende Beispiel deckt die meisten Anwendungsfälle ab. Sehen Sie sich die CLI-Referenz für alle Konfigurationsoptionen an.

Im nächsten Schritt fügen Sie die Zugriffsrichtlinie hinzu.

```
aws eks create-access-entry --cluster-name <cluster-name> --principal-arn <iam-
identity-arn> --type STANDARD
```

Schritt 3: Zugriffsrichtlinie zuordnen

Der Befehl unterscheidet sich je nachdem, ob Sie die Richtlinie auf einen bestimmten Kubernetes-Namespace beschränken möchten.

Sie benötigen die ARN der Zugriffsrichtlinie. Überprüfen Sie die verfügbaren Zugriffsrichtlinien.

Erstellung einer Richtlinie ohne Namespace-Gültigkeitsbereich

```
aws eks associate-access-policy --cluster-name <cluster-name> --principal-arn <iam-
identity-arn> --policy-arn <access-policy-arn>
```

Mit Namespace-Gültigkeitsbereich erstellen

```
aws eks associate-access-policy --cluster-name <cluster-name> --principal-arn <iam-
identity-arn> \
    --access-scope type=namespace,namespaces=my-namespace1,my-namespace2 --policy-arn
 <access-policy-arn>
```

Nächste Schritte

- Erstellung einer kubeconfig für die Verwendung von kubectl mit einer IAM-Identität

Erstellung eines Zugriffseintrags mithilfe von Kubernetes-Gruppen über die AWS CLI

Erstellen Sie Amazon-EKS-Zugriffseinträge, die Kubernetes-Gruppen zur Autorisierung verwenden und eine manuelle RBAC-Konfiguration erfordern.

> **Note**
>
> Für die meisten Anwendungsfälle empfehlen wir die Verwendung von EKS-Zugriffsrichtlinien anstelle des auf dieser Seite beschriebenen Ansatzes mit Kubernetes-Gruppen. EKS-Zugriffsrichtlinien bieten eine einfachere, besser in AWS integrierte Möglichkeit zur Zugriffsverwaltung ohne manuelle RBAC-Konfiguration. Verwenden Sie den Kubernetes-Gruppenansatz nur, wenn Sie eine detailliertere Steuerung benötigen, als die EKS-Zugriffsrichtlinien bieten.

Übersicht

Zugriffseinträge definieren, wie IAM-Identitäten (Benutzer und Rollen) auf Ihre Kubernetes-Cluster zugreifen. Der Kubernetes-Gruppenansatz gewährt IAM-Benutzern oder -Rollen die Berechtigung, über standardmäßige Kubernetes-RBAC-Gruppen auf Ihren EKS-Cluster zuzugreifen. Diese Methode erfordert die Erstellung und Verwaltung von Kubernetes-RBAC-Ressourcen (Roles, RoleBindings, ClusterRoles und ClusterRoleBindings). Sie wird empfohlen, wenn Sie stark angepasste Berechtigungssätze und komplexe Autorisierungsanforderungen benötigen oder konsistente Zugriffskontrollmuster in hybriden Kubernetes-Umgebungen beibehalten möchten.

Dieses Thema behandelt nicht das Erstellen von Zugriffseinträgen für IAM-Identitäten, die für Amazon-EC2-Instances zum Beitritt zu EKS-Clustern verwendet werden.

Voraussetzungen

- Der Authentifizierungsmodus Ihres Clusters muss konfiguriert sein, um Zugriffseinträge zu ermöglichen. Weitere Informationen finden Sie unter the section called “Authentifizierungsmodus”.
- Installieren und konfigurieren Sie die AWS CLI wie im Benutzerhandbuch zur AWS-Befehlszeilenschnittstelle unter Installieren beschrieben.
- Vertrautheit mit Kubernetes RBAC wird empfohlen. Weitere Informationen finden Sie unter Using RBAC Authorization in der Kubernetes-Dokumentation.

Schritt 1: Zugriffseintrag definieren

1. Suchen Sie die ARN der IAM-Identität, beispielsweise eines Benutzers oder einer Rolle, der bzw. der Sie Berechtigungen gewähren möchten.
   - Jede IAM-Identität kann nur einen EKS-Zugriffseintrag haben.
2. Legen Sie fest, welche Kubernetes-Gruppen Sie dieser IAM-Identität zuordnen möchten.

- Sie müssen vorhandene Kubernetes-`Role`/`ClusterRole` und `RoleBinding`/`ClusterRoleBinding`-Ressourcen erstellen oder verwenden, die auf diese Gruppen verweisen.

3. Prüfen Sie, ob der automatisch generierte Benutzername für den Zugriffseintrag geeignet ist oder ob Sie einen Benutzernamen manuell angeben müssen.
   - AWS generiert diesen Wert automatisch basierend auf der IAM-Identität. Sie können einen benutzerdefinierten Benutzernamen festlegen. Dieser wird in den Kubernetes-Protokollen angezeigt.
   - Weitere Informationen finden Sie unter the section called “Benutzerdefinierten Benutzernamen festlegen”.

Schritt 2: Zugriffseintrag mit Kubernetes-Gruppen erstellen

Nachdem Sie den Zugriffseintrag geplant haben, erstellen Sie ihn mithilfe der AWS CLI mit den entsprechenden Kubernetes-Gruppen.

```
aws eks create-access-entry --cluster-name <cluster-name> --principal-arn <iam-
identity-arn> --type STANDARD --kubernetes-groups <groups>
```

Ersetzen Sie:

- `<cluster-name>` durch den Namen Ihres EKS-Clusters
- `<iam-identity-arn>` durch die ARN des IAM-Benutzers oder der IAM-Rolle
- `<groups>` durch eine durch Kommas getrennte Liste von Kubernetes-Gruppen (z. B. „system:developers,system:readers“).

Sehen Sie sich die CLI-Referenz für alle Konfigurationsoptionen an.

Schritt 3: Kubernetes-RBAC konfigurieren

Damit der IAM-Prinzipal Zugriff auf Kubernetes-Objekte in Ihrem Cluster hat, müssen Sie rollenbasierte Zugriffskontrollobjekte (RBAC) für Kubernetes erstellen und verwalten:

1. Erstellen Sie Kubernetes `Role` oder `ClusterRole`-Objekte, welche die Berechtigungen definieren.
2. Erstellen Sie Kubernetes `RoleBinding` oder `ClusterRoleBinding`-Objekte in Ihrem Cluster, die den Gruppennamen als `subject` für `kind: Group` angeben.

Ausführliche Informationen zum Konfigurieren von Gruppen und Berechtigungen in Kubernetes finden Sie unter Verwendung der RBAC-Autorisierung in der Kubernetes-Dokumentation.

Nächste Schritte

- Erstellung einer kubeconfig für die Verwendung von kubectl mit einer IAM-Identität

Gewähren Sie IAM-Benutzern Zugriff auf Kubernetes mit einem ConfigMap

> **Important**
>
> Der `aws-auth ConfigMap` ist veraltet. Die empfohlene Methode zur Verwaltung des Zugriffs auf Kubernetes finden Sie unter. APIs the section called “Zugriffseinträge”

Zugriff auf Ihren Cluster mit einem IAM-Prinzipal wird durch AWS IAM Authenticator für Kubernetes aktiviert, das in der Amazon-EKS-Steuerebene ausgeführt wird. Der Authenticator erhält seine Konfigurationsinformationen von `aws-auth ConfigMap`. Alle `aws-auth ConfigMap`-Einstellungen finden Sie unter Vollständiges Konfigurationsformat auf GitHub.

Hinzufügen von IAM-Prinzipal zu Ihrem Amazon EKS-Cluster

Wenn Sie einen Amazon-EKS-Cluster erstellen, werden dem IAM-Prinzipal, der den Cluster erstellt, automatisch `system:masters`-Berechtigungen in der rollenbasierten Konfiguration für Zugriffskontrolle (RBAC) des Clusters in der Amazon-EKS-Steuerebene gewährt. Dieser Prinzipal wird in keiner sichtbaren Konfiguration angezeigt. Achten Sie daher darauf, welcher Prinzipal den Cluster ursprünglich erstellt hat. Um zusätzlichen IAM-Prinzipalen die Möglichkeit zu geben, mit Ihrem Cluster zu interagieren, bearbeiten Sie den `aws-auth ConfigMap` in Kubernetes und erstellen Sie ein Kubernetes-`rolebinding` oder -`clusterrolebinding` mit dem Namen eines `group`, das Sie im `aws-auth ConfigMap` angeben.

> **Note**
>
> Weitere Informationen zur Konfiguration der rollenbasierten Zugriffskontrolle (RBAC) von Kubernetes finden Sie unter Verwendung der RBAC-Autorisierung in der Kubernetes-Dokumentation.

1. Bestimmen Sie, welche Anmeldeinformationen `kubectl` für den Zugriff auf Ihren Cluster verwendet. Amazon Resource Name (ARN)Auf Ihrem Computer können Sie mit dem folgenden Befehl sehen, welche Anmeldeinformationen `kubectl` verwendet. *`~/.kube/config`*Ersetzen Sie ihn durch den Pfad zu Ihrer `kubeconfig` Datei, wenn Sie nicht den Standardpfad verwenden.

```
cat ~/.kube/config
```

Eine Beispielausgabe sieht wie folgt aus.

```
[...]
contexts:
- context:
    cluster: my-cluster.region-code.eksctl.io
    user: admin@my-cluster.region-code.eksctl.io
  name: admin@my-cluster.region-code.eksctl.io
current-context: admin@my-cluster.region-code.eksctl.io
[...]
```

In der vorherigen Beispielausgabe werden die Anmeldeinformationen für einen Benutzer mit dem Namen *`admin`* für einen Cluster mit dem Namen *`my-cluster`* konfiguriert. Wenn dies der Benutzer ist, der den Cluster erstellt hat, hat er bereits Zugriff auf Ihren Cluster. Wenn es nicht der Benutzer ist, der den Cluster erstellt hat, müssen Sie die verbleibenden Schritte ausführen, um den Cluster-Zugriff für andere IAM-Prinzipale zu aktivieren. Bewährte Methoden für IAM empfehlen, dass Sie Rollen statt Benutzern Berechtigungen gewähren. Mit dem folgenden Befehl können Sie sehen, welche anderen Prinzipale derzeit Zugriff auf Ihren Cluster haben:

```
kubectl describe -n kube-system configmap/aws-auth
```

Eine Beispielausgabe sieht wie folgt aus.

```
Name:         aws-auth
Namespace:    kube-system
Labels:       <none>
Annotations:  <none>

Data
====
mapRoles:
----
```

```
- groups:
  - system:bootstrappers
  - system:nodes
  rolearn: arn:aws: iam::111122223333:role/my-node-role
  username: system:node:{{EC2PrivateDNSName}}


BinaryData
====

Events:  <none>
```

Das vorherige Beispiel ist eine standardmäßige `aws-auth-ConfigMap`. Nur die Knoten-Instance-Rolle hat Zugriff auf den Cluster.

2. Stellen Sie sicher, dass `roles` und `rolebindings` oder `clusterroles` und `clusterrolebindings` vorhanden sind, denen Sie IAM-Prinzipale zuordnen können. Weitere Informationen über diese Ressourcen finden Sie unter RBAC-Autorisierung verwenden in der Kubernetes-Dokumentation.

   a. Zeigen Sie Ihre vorhandenen Kubernetes `roles` oder `clusterroles` an. `Roles` sind auf einen `namespace` ausgelegt, aber `clusterroles` sind auf den Cluster ausgelegt.

   ```
   kubectl get roles -A
   ```

   ```
   kubectl get clusterroles
   ```

   b. Zeigen Sie die Details einer beliebigen `role` oder `clusterrole` an, die in der vorherigen Ausgabe zurückgegeben wurden, und bestätigen Sie, dass Sie die Berechtigungen (`rules`) hat, die Ihre IAM-Prinzipale in Ihrem Cluster haben sollen.

   Ersetzen Sie *`role-name`* durch einen `role`-Namen, den der vorherige Befehl zurückgegeben hat. Ersetzen Sie *`kube-system`* durch den Namespace der `role`.

   ```
   kubectl describe role role-name -n kube-system
   ```

   Ersetzen Sie *`cluster-role-name`* durch einen `clusterrole`-Namen, den der vorherige Befehl zurückgegeben hat.

   ```
   kubectl describe clusterrole cluster-role-name
   ```

c. Zeigen Sie Ihre vorhandenen Kubernetes `rolebindings` oder `clusterrolebindings` an. `Rolebindings` sind auf einen `namespace` ausgelegt, aber `clusterrolebindings` sind auf den Cluster ausgelegt.

```
kubectl get rolebindings -A
```

```
kubectl get clusterrolebindings
```

d. Zeigen Sie die Details einer beliebigen `rolebinding` oder `clusterrolebinding` an und bestätigen Sie, dass sie eine `role` oder `clusterrole` aus dem vorherigen Schritt hat, die als `roleRef` aufgeführt wird, und einen Gruppennamen, der für `subjects` aufgeführt wird.

Ersetzen Sie *`role-binding-name`* durch einen `rolebinding`-Namen, den der vorherige Befehl zurückgegeben hat. Ersetzen SIe *`kube-system`* mit dem `namespace` der `rolebinding`.

```
kubectl describe rolebinding role-binding-name -n kube-system
```

Eine Beispielausgabe sieht wie folgt aus.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: eks-console-dashboard-restricted-access-role-binding
  namespace: default
subjects:
- kind: Group
  name: eks-console-dashboard-restricted-access-group
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: eks-console-dashboard-restricted-access-role
  apiGroup: rbac.authorization.k8s.io
```

Ersetzen Sie *`cluster-role-binding-name`* durch einen `clusterrolebinding`-Namen, den der vorherige Befehl zurückgegeben hat.

```
kubectl describe clusterrolebinding cluster-role-binding-name
```

Eine Beispielausgabe sieht wie folgt aus.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks-console-dashboard-full-access-binding
subjects:
- kind: Group
  name: eks-console-dashboard-full-access-group
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: eks-console-dashboard-full-access-clusterrole
  apiGroup: rbac.authorization.k8s.io
```

3. Bearbeiten Sie die `aws-auth-ConfigMap`. Sie können ein Tool wie `eksctl` verwenden, um die `ConfigMap` zu aktualisieren, oder Sie können sie durch manuelle Bearbeitung aktualisieren.

> **Important**
>
> Wir empfehlen die Verwendung von `eksctl` oder einem anderen Tool, um die `ConfigMap` zu bearbeiten. Informationen zu anderen Tools, die Sie verwenden können, finden Sie unter Use tools to make changes to the aws- authConfigMap in den Amazon EKS-Best-Practices-Leitfäden. Ist `aws-auth-ConfigMap` falsch formatiert, können Sie den Zugriff auf Ihren Cluster verlieren.

- Schritte zum Bearbeiten der Konfigurationszuordnung mit eksctl anzeigen.
- Schritte zum manuellen Bearbeiten der Konfigurationszuordnung anzeigen.

Configmap mit Eksctl bearbeiten

1. Sie benötigen eine Version `0.215.0` oder eine neuere Version des `eksctl` Befehlszeilentools, das auf Ihrem Gerät installiert ist oder AWS CloudShell. Informationen zum Installieren und Aktualisieren von `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.
2. Zeigen Sie die aktuellen Mappings in der `ConfigMap` an. Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters. *`region-code`*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet.

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

Eine Beispielausgabe sieht wie folgt aus.

```
ARN
            USERNAME                                      GROUPS
 ACCOUNT
arn:aws: iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
```

3. Fügen Sie ein Mapping für eine Rolle hinzu. Ersetzen Sie *my-role* durch den Namen Ihrer Rolle. *eks-console-dashboard-full-access-group*Ersetzen Sie durch den Namen der Gruppe, die in Ihrem Kubernetes `RoleBinding` oder `ClusterRoleBinding` Objekt angegeben ist. Ersetzen Sie *111122223333* durch Ihre Konto-ID. Sie können *admin* mit einem beliebigen Namen ersetzen, den Sie wählen.

```
eksctl create iamidentitymapping --cluster my-cluster --region=region-code \
    --arn arn:aws: iam::111122223333:role/my-role --username admin --group eks-
console-dashboard-full-access-group \
    --no-duplicate-arns
```

> **Important**
>
> Der Rollen-ARN darf keinen Pfad wie `role/my-team/developers/my-role` enthalten. Das Format des ARN muss `arn:aws: iam::`*111122223333*`:role/`*my-role* sein. In diesem Beispiel muss `my-team/developers/` entfernt werden.

Eine Beispielausgabe sieht wie folgt aus.

```
[...]
2022-05-09 14:51:20 [#]  adding identity "{arn-aws}iam::111122223333:role/my-role" to
 auth ConfigMap
```

4. Fügen Sie ein Mapping für einen Benutzer hinzu. Bewährte Methoden für IAM empfehlen, dass Sie Rollen statt Benutzern Berechtigungen gewähren. Ersetzen Sie *my-user* durch den Benutzernamen. *eks-console-dashboard-restricted-access-group*Ersetzen Sie es

durch den Namen der Gruppe, die in Ihrem `RoleBinding` Kubernetes oder Objekt angegeben ist. `ClusterRoleBinding` Ersetzen Sie *111122223333* durch Ihre Konto-ID. Sie können *my-user* mit einem beliebigen Namen ersetzen, den Sie wählen.

```
eksctl create iamidentitymapping --cluster my-cluster --region=region-code \
    --arn arn:aws: iam::111122223333:user/my-user --username my-user --group eks-
console-dashboard-restricted-access-group \
    --no-duplicate-arns
```

Eine Beispielausgabe sieht wie folgt aus.

```
[...]
2022-05-09 14:53:48 [#]  adding identity "arn:aws: iam::111122223333:user/my-user" to
 auth ConfigMap
```

5. Zeigen Sie wieder die Mappings in der `ConfigMap` an.

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

Eine Beispielausgabe sieht wie folgt aus.

```
ARN
           USERNAME                                GROUPS
      ACCOUNT
arn:aws: iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
arn:aws: iam::111122223333:role/admin
               my-role                                   eks-console-dashboard-full-
access-group
arn:aws: iam::111122223333:user/my-user
            my-user                                  eks-console-dashboard-restricted-
access-group
```

Configmap manuell bearbeiten

1. Öffnen Sie `ConfigMap` zum Bearbeiten.

```
kubectl edit -n kube-system configmap/aws-auth
```

> **Note**
>
> Wenn Sie eine Fehlermeldung mit der Angabe "`Error from server (NotFound): configmaps "aws-auth" not found`" erhalten, wenden Sie das Verfahren unter AWS-Auth auf Ihren Cluster anwenden ConfigMap an, um den Bestand anzuwenden. `ConfigMap`

2. Fügen Sie Ihre IAM-Prinzipale zu der `ConfigMap` hinzu. Eine IAM-Gruppe ist kein IAM-Prinzipal und kann daher nicht zu `ConfigMap` hinzugefügt werden.
   - So fügen Sie eine IAM-Rolle hinzu (z. B. für Verbundbenutzer): Fügen Sie die Rollendetails zum Abschnitt `mapRoles` der `ConfigMap` unter `data` hinzu. Fügen Sie diesen Abschnitt hinzu, wenn er nicht bereits in der Datei vorhanden sind. Jeder Eintrag unterstützt die folgenden Parameter:
     - rolearn: Der ARN der IAM-Rolle, den Sie hinzufügen möchten. Dieser Wert darf keinen Pfad enthalten. Sie können beispielsweise keinen ARN wie `arn:aws: iam::111122223333:role/my-team/developers/role-name` angeben. Der ARN muss stattdessen `arn:aws: iam::111122223333:role/role-name` sein.
     - username: Der Benutzername in Kubernetes für die Zuweisung zur IAM-Rolle.
     - Gruppen: Die Gruppe oder Liste der Kubernetes-Gruppen, denen die Rolle zugeordnet werden soll. Die Gruppe kann eine Standardgruppe oder eine Gruppe sein, die in einer `clusterrolebinding` oder `rolebinding` angegeben ist. Weitere Informationen finden Sie unter Standardrollen und Rollenbindungen in der Kubernetes-Dokumentation.
   - Um einen IAM-Benutzer hinzuzufügen: Bewährte Methoden von IAM empfehlen, dass Sie Rollen statt Benutzern Berechtigungen gewähren. Fügen Sie die Benutzerdetails zum `mapUsers`-Abschnitt der `ConfigMap` unter `data` hinzu. Fügen Sie diesen Abschnitt hinzu, wenn er nicht bereits in der Datei vorhanden sind. Jeder Eintrag unterstützt die folgenden Parameter:
     - userarn: Die ARN des IAM-Benutzers, den Sie hinzufügen möchten.
     - username: Der Benutzername in Kubernetes für die Zuweisung zum IAM-Benutzer.
     - Gruppen: Die Gruppe oder Liste der Kubernetes-Gruppen, denen der Benutzer zugeordnet werden soll. Die Gruppe kann eine Standardgruppe oder eine Gruppe sein, die in einer `clusterrolebinding` oder `rolebinding` angegeben ist. Weitere Informationen finden Sie unter Standardrollen und Rollenbindungen in der Kubernetes-Dokumentation.
3. Beispiel: Der folgende YAML-Block enthält:

- Einen `mapRoles`-Abschnitt, der die IAM-Knoten-Instance Kubernetes-Gruppen zuordnet, sodass sich Knoten selbst bei dem Cluster und der `my-console-viewer-role`-IAM-Rolle registrieren können, die einer Kubernetes-Gruppe zugeordnet ist, die alle Kubernetes-Ressourcen für alle Cluster anzeigen kann. Eine Liste der erforderlichen IAM- und Kubernetes-Gruppenberechtigungen für die `my-console-viewer-role`-IAM-Rolle finden Sie unter the section called “Erforderliche Berechtigungen”.
- Ein `mapUsers` Abschnitt, der den `admin` IAM-Benutzer aus dem AWS Standardkonto der `system:masters` Kubernetes-Gruppe und den `my-user` Benutzer aus einem anderen AWS Konto zuordnet, das einer Kubernetes-Gruppe zugeordnet ist, die Kubernetes-Ressourcen für einen bestimmten Namespace anzeigen kann. Eine Liste der erforderlichen IAM- und Kubernetes-Gruppenberechtigungen für die `my-user`-IAM-Benutzer finden Sie unter the section called “Erforderliche Berechtigungen”.

  Fügen Sie nach Bedarf Zeilen hinzu oder entfernen Sie sie und ersetzen Sie alle Beispielwerte durch Ihre eigenen Werte.

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file
 will be
# reopened with the relevant failures.
#
apiVersion: v1
data:
  mapRoles: |
    - groups:
      - system:bootstrappers
      - system:nodes
      rolearn: arn:aws: iam::111122223333:role/my-role
      username: system:node:{{EC2PrivateDNSName}}
    - groups:
      - eks-console-dashboard-full-access-group
      rolearn: arn:aws: iam::111122223333:role/my-console-viewer-role
      username: my-console-viewer-role
  mapUsers: |
    - groups:
      - system:masters
      userarn: arn:aws: iam::111122223333:user/admin
      username: admin
    - groups:
      - eks-console-dashboard-restricted-access-group
      userarn: arn:aws: iam::444455556666:user/my-user
```

```
    username: my-user
```

4. Speichern Sie die Datei und beenden Sie den Text-Editor.

Anwenden von **aws-authConfigMap** auf Ihren Cluster

`aws-auth ConfigMap` wird automatisch erstellt und auf Ihren Cluster angewendet, wenn Sie eine verwaltete Knotengruppe erstellen oder wenn Sie eine Knotengruppe mit `eksctl` erstellen. Sie wird anfänglich erstellt, um Knoten zu erlauben, Ihrem Cluster beizutreten, aber Sie verwenden diese `ConfigMap` auch, um rollenbasierten Zugriffssteuerungs(RBAC)-Zugriff auf IAM-Prinzipale hinzuzufügen. Wenn Sie keine selbstverwalteten Knoten gestartet und `aws-auth ConfigMap` nicht auf Ihren Cluster angewendet haben, können Sie das folgende Verfahren dafür verwenden.

1. Überprüfen Sie, ob Sie `aws-auth ConfigMap` bereits angewendet haben.

   ```
   kubectl describe configmap -n kube-system aws-auth
   ```

   Wenn die Fehlermeldung „ `Error from server (NotFound): configmaps "aws-auth" not found` “ angezeigt wird, fahren Sie mit den folgenden Schritten fort, um die Standard-`ConfigMap` anzuwenden.

2. Laden Sie die Authenticator-Konfigurationsübersicht herunter, bearbeiten Sie sie und wenden Sie sie an. AWS

   a. Laden Sie die Konfigurationszuordnung herunter.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/
   aws-auth-cm.yaml
   ```

   b. Stellen Sie in der Datei `aws-auth-cm.yaml` `rolearn` auf den Amazon-Ressourcennamen (ARN) der IAM-Rolle ein, die Ihren Knoten zugeordnet ist. Hierzu können Sie einen Texteditor verwenden oder *`my-node-instance-role`* ersetzen und den folgenden Befehl ausführen:

   ```
   sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-instance-
   role|' aws-auth-cm.yaml
   ```

   Ändern Sie keine weiteren Zeilen in dieser Datei.

> **Important**
>
> Der Rollen-ARN darf keinen Pfad wie `role/my-team/developers/my-role` enthalten. Das Format des ARN muss `arn:aws: iam::111122223333:role/my-role` sein. In diesem Beispiel muss `my-team/developers/` entfernt werden.

Sie können die AWS CloudFormation Stack-Ausgaben für Ihre Knotengruppen überprüfen und nach den folgenden Werten suchen:

- InstanceRoleARN — Für Knotengruppen, die erstellt wurden mit `eksctl`
- NodeInstanceRole— Für Knotengruppen, die mit von Amazon EKS bereitgestellten AWS CloudFormation Vorlagen erstellt wurden, im AWS-Managementkonsole

c. Wenden Sie die Konfiguration an. Die Ausführung dieses Befehls kann einige Minuten dauern.

```
kubectl apply -f aws-auth-cm.yaml
```

> **Note**
>
> Wenn Sie Autorisierungs- oder Ressourcenfehler erhalten, finden Sie weitere Informationen unter the section called “Nicht autorisiert oder Zugriff verweigert (kubectl)” im Thema zur Fehlerbehebung.

3. Sehen Sie sich den Status Ihrer Knoten an und warten Sie, bis diese in den `Ready`-Status eintreten.

```
kubectl get nodes --watch
```

Geben Sie `Ctrl+C` ein, um zum Shell-Prompt zurückzukehren.

Benutzern Zugriff auf Kubernetes mit einem externen OIDC-Anbieter gewähren

Amazon EKS unterstützt die Verwendung von OpenID-Connect(OIDC)-Identitätsanbietern als Methode zur Authentifizierung von Benutzern bei Ihrem Cluster. OIDC-Identitätsanbieter können zusammen mit oder als Alternative zu AWS Identity and Access Management (IAM)

verwendet werden. Weitere Informationen zur Verwendung von IAM finden Sie unter the section called “Kubernetes-API-Zugriff”. Nachdem Sie die Authentifizierung für Ihren Cluster konfiguriert haben, können Sie Kubernetes `roles` und `clusterroles` erstellen, um den Rollen Berechtigungen zuzuweisen, und dann die Rollen mithilfe von Kubernetes `rolebindings` und `clusterrolebindings` an die Identitäten binden. Weitere Informationen finden Sie unter Verwendung der RBAC-Autorisierung in der Kubernetes-Dokumentation.

- Sie können Ihrem Cluster einen OIDC-Identitätsanbieter zuordnen.
- Kubernetes stellt keinen OIDC-Identitätsanbieter bereit. Sie können einen vorhandenen öffentlichen OIDC-Identitätsanbieter verwenden oder Ihren eigenen Identitätsanbieter ausführen. Eine Liste zertifizierter Anbieter finden Sie unter OpenID-Zertifizierung auf der OpenID-Website.
- Die Aussteller-URL des OIDC-Identitätsanbieters muss öffentlich zugänglich sein, damit Amazon EKS die Signaturschlüssel erkennen kann. Amazon EKS unterstützt keine OIDC-Identitätsanbieter mit selbstsignierten Zertifikaten.
- Sie können die IAM-Authentifizierung in Ihrem Cluster nicht deaktivieren, da sie weiterhin erforderlich ist, um einem Cluster Knoten hinzufügen zu können.
- Ein Amazon EKS-Cluster muss weiterhin von einem AWS -IAM-Prinzipal erstellt werden und nicht von einem OIDC-Identitätsanbieter-Benutzer. Das liegt daran, dass der Cluster-Ersteller mit Amazon EKS APIs und nicht mit APIs Kubernetes interagiert.
- Benutzer, die vom OIDC-Identitätsanbieter authentifiziert wurden, werden im Auditprotokoll des Clusters aufgeführt, wenn die CloudWatch Protokolle für die Kontrollebene aktiviert sind. Weitere Informationen finden Sie unter the section called “Aktivierung oder Deaktivierung der Steuerebenen-Protokolle”.
- Sie können sich nicht AWS-Managementkonsole mit einem Konto eines OIDC-Anbieters bei der anmelden. Sie können dies nur the section called “Zugriff auf Cluster-Ressourcen” tun, indem Sie sich AWS-Managementkonsole mit einem AWS Identity and Access Management-Konto anmelden.

Verknüpfen Sie einen OIDC-Identitätsanbieter

Bevor Sie Ihrem Cluster einen OIDC-Identitätsanbieter zuordnen können, benötigen Sie die folgenden Informationen von Ihrem Anbieter:

URL des Ausstellers

Die URL des OIDC-Identitätsanbieters, die es dem API-Server ermöglicht, öffentliche Signierschlüssel zur Verifizierung von Token zu ermitteln. Die URL muss mit `https://` beginnen und sollte dem `iss`-Anspruch in den OIDC-ID-Token des Anbieters entsprechen. Gemäß dem OIDC-Standard sind Pfadkomponenten erlaubt, Abfrageparameter jedoch nicht. Normalerweise besteht die URL nur aus einem Hostnamen wie `https://server.example.org` oder `https://example.com`. Diese URL sollte auf die Ebene unterhalb von `.well-known/openid-configuration` verweisen und muss über das Internet öffentlich zugänglich sein.

Client-ID (auch als Zielgruppe bezeichnet)

Die ID für die Client-Anwendung, die Authentifizierungsanforderungen an den OIDC-Identitätsanbieter stellt.

Sie können einen Identitätsanbieter mit `eksctl` oder AWS-Managementkonsole zuordnen.

Identitätsanbieter mit eksctl zuordnen

1. Erstellen Sie eine Datei mit dem Namen `associate-identity-provider.yaml` und dem folgenden Inhalt. Ersetzen Sie die Beispielwerte durch Ihre eigenen Werte. Die Werte im Abschnitt `identityProviders` erhalten Sie von Ihrem OIDC-Identitätsanbieter. Werte werden nur für die `name`-, `type`-, `issuerUrl`- und `clientId`-Einstellungen unter `identityProviders` benötigt.

```
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: your-region-code

identityProviders:
  - name: my-provider
    type: oidc
    issuerUrl: https://example.com
    clientId: kubernetes
    usernameClaim: email
    usernamePrefix: my-username-prefix
    groupsClaim: my-claim
    groupsPrefix: my-groups-prefix
```

```
    requiredClaims:
      string: string
    tags:
      env: dev
```

> **Important**
>
> Geben Sie weder `system:` noch einen Teil dieser Zeichenfolge für `groupsPrefix` oder `usernamePrefix` an.

2. Erstellen Sie den Anbieter.

```
eksctl associate identityprovider -f associate-identity-provider.yaml
```

3. Informationen zum Verwenden von `kubectl` für die Arbeit mit Ihrem Cluster und OIDC-Identitätsanbieter finden Sie unter Verwendung von kubectl in der Kubernetes-Dokumentation.

Ordnen Sie mithilfe der Konsole einen Identitätsanbieter zu AWS

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie Ihren Cluster aus und anschließend die Registerkarte Zugriff.
3. Wählen Sie im Abschnitt OIDC-Identitätsanbieter die Option „Identitätsanbieter zuordnen“ aus.
4. Geben Sie auf der Seite OIDC-Identitätsanbieter zuordnen die folgenden Optionen ein oder wählen Sie sie aus, und wählen Sie dann Zuordnen aus.
   - Geben Sie für Name einen eindeutigen Namen für den Anbieter ein.
   - Geben Sie für Aussteller-URL die URL Ihres Anbieters ein. Diese URL muss über das Internet zugänglich sein.
   - Geben Sie für Client-ID die Client-ID des OIDC-Identitätsanbieters (auch als Zielgruppe bezeichnet) ein.
   - Geben Sie für Benutzernamenanspruch den Anspruch ein, der als Benutzername verwendet werden soll.
   - Geben Sie für Gruppenanspruch den Anspruch ein, der als Benutzergruppe verwendet werden soll.
   - (Optional) Wählen Sie Erweiterte Optionen, geben Sie die folgenden Informationen ein oder wählen Sie sie aus.

- Benutzernamen-Präfix – Geben Sie ein Präfix ein, das den Benutzernamenansprüchen vorangestellt wird. Das Präfix wird den Benutzernamensansprüchen vorangestellt, um Konflikte mit bestehenden Namen zu vermeiden. Wenn Sie keinen Wert angeben und der Benutzername ein anderer Wert als `email` ist, wird als Präfix standardmäßig der Wert für die Aussteller-URL verwendet. Sie können den Wert `-` verwenden, um alle Präfixe zu deaktivieren. Geben Sie weder `system:`, noch einen Teil dieser Zeichenfolge an.
- Gruppen-Präfix – Geben Sie ein Präfix ein, das den Gruppenansprüchen vorangestellt werden soll. Das Präfix wird Gruppenansprüchen vorangestellt, um Konflikte mit vorhandenen Namen (wie beispielsweise `system: groups`) zu vermeiden. Der Wert `oidc:` erstellt beispielsweise Gruppennamen wie `oidc:engineering` und `oidc:infra`. Geben Sie weder `system:`, noch einen Teil dieser Zeichenfolge an.
- Erforderliche Ansprüche – Wählen Sie Anspruch hinzufügen aus, und geben Sie ein oder mehrere Schlüsselwertpaare ein, die erforderliche Ansprüche im Client-ID-Token beschreiben. Die Paare beschreiben erforderliche Ansprüche im ID-Token. Wenn festgelegt, wird überprüft, ob jeder Anspruch im ID-Token mit einem übereinstimmenden Wert vorhanden ist.
  a. Informationen zum Verwenden von `kubectl` für die Arbeit mit Ihrem Cluster und OIDC-Identitätsanbieter finden Sie unter Verwendung von kubectl in der Kubernetes-Dokumentation.

Beispiel für eine IAM-Richtlinie

Wenn Sie verhindern möchten, dass ein OIDC-Identitätsanbieter mit einem Cluster verknüpft wird, erstellen Sie die folgende IAM-Richtlinie und verknüpfen Sie sie mit den IAM-Konten Ihrer Amazon-EKS-Administratoren. Weitere Informationen finden Sie unter Erstellen von IAM-Richtlinien und Hinzufügen von IAM-Identitätsberechtigungen im IAM-Benutzerhandbuch sowie unter Aktionen in der Referenz zur Service-Autorisierung.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "denyOIDC",
            "Effect": "Deny",
            "Action": [
                "eks:AssociateIdentityProviderConfig"
            ],
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/*"
```

```
        },
        {
            "Sid": "eksAdmin",
            "Effect": "Allow",
            "Action": [
                "eks:*"
            ],
            "Resource": "*"
        }
    ]
}
```

Die folgende Beispielrichtlinie lässt die Zuordnung von OIDC-Identitätsanbietern zu, wenn `clientID` `kubernetes` und `issuerUrl` `https://cognito-idp.us-west-2amazonaws.com/*` ist.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCognitoOnly",
            "Effect": "Deny",
            "Action": "eks:AssociateIdentityProviderConfig",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-instance",
            "Condition": {
                "StringNotLikeIfExists": {
                    "eks:issuerUrl": "https://cognito-idp.us-west-2.amazonaws.com/*"
                }
            }
        },
        {
            "Sid": "DenyOtherClients",
            "Effect": "Deny",
            "Action": "eks:AssociateIdentityProviderConfig",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-instance",
            "Condition": {
                "StringNotEquals": {
                    "eks:clientId": "kubernetes"
                }
            }
        },
        {
            "Sid": "AllowOthers",
            "Effect": "Allow",
```

```
            "Action": "eks:*",
            "Resource": "*"
        }
    ]
}
```

Trennen Sie einen OIDC-Identitätsanbieter von Ihrem Cluster

Wenn Sie einen OIDC-Identitätsanbieter von Ihrem Cluster trennen, können Benutzer, die im Anbieter enthalten sind, nicht mehr auf den Cluster zugreifen. Sie können jedoch weiterhin mit IAM-Prinzipalen auf den Cluster zugreifen.

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im Abschnitt OIDC-Identitätsanbieter die Option Zuordnung aufheben, geben Sie den Namen des Identitätsanbieters ein und wählen Sie dann `Disassociate` aus.

Kubernetes-Ressourcen anzeigen in der AWS-Managementkonsole

Mit der AWS-Managementkonsole können Sie die Kubernetes-Ressourcen anzeigen, die in Ihrem Cluster bereitgestellt sind. Sie können Kubernetes-Ressourcen nicht mit der AWS CLI oder eksctl anzeigen. Um Kubernetes-Ressourcen mit einem Befehlszeilen-Tool anzuzeigen, verwenden Sie kubectl.

> **Note**
>
> Um die Registerkarte Ressourcen und den Abschnitt Knoten auf der Registerkarte Compute in der anzeigen zu können AWS-Managementkonsole, muss der von Ihnen verwendete IAM-Prinzipal über bestimmte IAM - und Kubernetes-Berechtigungen verfügen. Weitere Informationen finden Sie unter the section called “Erforderliche Berechtigungen”.

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie in der Liste Cluster den Cluster aus, der die Kubernetes-Ressourcen enthält, die Sie anzeigen möchten.
3. Wählen Sie die Registerkarte für Resources (Ressourcen).

4. Wählen Sie eine Resource type-Gruppe (Ressourcentyp) aus, für die Sie Ressourcen anzeigen möchten, z. B. Workloads. Ihnen wird eine Liste der Ressourcentypen in dieser Gruppe angezeigt.
5. Wählen Sie einen Ressourcentyp aus, z. B. Deployments (Bereitstellungen) in der Gruppe Workloads. Sie sehen eine Beschreibung des Ressourcentyps, einen Link zur Kubernetes-Dokumentation für weitere Informationen über den Ressourcentyp und eine Liste der Ressourcen dieses Typs, die in Ihrem Cluster bereitgestellt werden. Wenn die Liste leer ist, werden keine Ressourcen dieses Typs in Ihrem Cluster bereitgestellt.
6. Wählen Sie eine Ressource aus, um weitere Informationen dazu anzuzeigen. Probieren Sie die folgenden Beispiele aus:
   - Wählen Sie die Gruppe Workloads aus, dann den Ressourcentyp Deployments (Bereitstellungen) gefolgt von der Ressource coredns. Wenn Sie eine Ressource auswählen, befinden Sie sich standardmäßig in der strukturierten Ansicht. Für einige Ressourcentypen wird in der strukturierten Ansicht der Abschnitt Pods angezeigt. In diesem Abschnitt sind die Pods aufgeführt, die von der Workload verwaltet werden. Sie können jeden aufgelisteten Pod auswählen, um Informationen zu dem Pod anzuzeigen. Nicht für alle Ressourcentypen werden Informationen in der strukturierten Ansicht angezeigt. Wenn Sie die Ansicht Raw in der oberen rechten Ecke der Seite für die Ressource auswählen, sehen Sie die vollständige JSON-Antwort von der Kubernetes-API für die Ressource.
   - Wählen Sie die Cluster-Gruppe aus und dann den Ressourcentyp Nodes (Knoten). Eine Liste aller Knoten in Ihrem Cluster wird angezeigt. Die Knoten können von jedem Amazon-EKS-Knoten-Typ sein. Das ist die gleiche Liste, die Sie im Abschnitt Nodes (Knoten) sehen, wenn Sie die Registerkarte Compute (Datenverarbeitung) für Ihren Cluster auswählen. Wählen Sie eine Knotenressource aus der Liste aus. In der strukturierten Ansicht sehen Sie ebenfalls einen Abschnitt Pods. Dieser Abschnitt zeigt Ihnen alle in dem Knoten ausgeführten Pods.

Erforderliche Berechtigungen

Um die Registerkarte Ressourcen und den Abschnitt Knoten auf der Registerkarte Compute in der anzeigen zu können AWS-Managementkonsole, muss der von Ihnen verwendete IAM-Prinzipal über bestimmte Mindestberechtigungen für IAM und Kubernetes verfügen. Führen Sie die folgenden Schritte aus, um Ihren IAM-Prinzipalen die erforderlichen Berechtigungen zuzuweisen.

1. Stellen Sie sicher, dass `eks:AccessKubernetesApi` und andere erforderliche IAM-Berechtigungen zum Anzeigen von Kubernetes-Ressourcen dem von Ihnen verwendeten IAM-Prinzipal zugewiesen sind. Weitere Informationen zum Bearbeiten von Berechtigungen für einen IAM-Prinzipal finden Sie unter Zugriff für Prinzipale steuern und im IAM-Benutzerhandbuch.

Weitere Informationen zum Bearbeiten von Rollenberechtigungen finden Sie unter Modifying a role permissions policy (console) (Ändern einer Rollenberechtigungsrichtlinie (Konsole)) im IAM-Benutzerhandbuch.

Die folgende Beispielrichtlinie enthält die erforderlichen Berechtigungen für einen Prinzipal zum Anzeigen von Kubernetes-Ressourcen für alle Cluster in Ihrem Konto. Ersetzen Sie es durch Ihre *111122223333* Konto-ID. AWS

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks:ListFargateProfiles",
                "eks:DescribeNodegroup",
                "eks:ListNodegroups",
                "eks:ListUpdates",
                "eks:AccessKubernetesApi",
                "eks:ListAddons",
                "eks:DescribeCluster",
                "eks:DescribeAddonVersions",
                "eks:ListClusters",
                "eks:ListIdentityProviderConfigs",
                "iam:ListRoles"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ssm:GetParameter",
            "Resource": "arn:aws:ssm:*:111122223333:parameter/*"
        }
    ]
}
```

Um Knoten in verbundenen Clustern anzuzeigen, sollte die IAM-Rolle des Amazon-EKS-Connectors in der Lage sein, den Prinzipal im Cluster anzunehmen. Dadurch kann Amazon EKS Connector den Prinzipal einem Kubernetes-Benutzer zuordnen.

2. Erstellen Sie ein Kubernetes `rolebinding` oder `clusterrolebinding`, das an ein Kubernetes `role` oder `clusterrole` gebunden ist, das über die erforderlichen Berechtigungen zum

Anzeigen der Kubernetes-Ressourcen verfügt. Informationen zu Rollen und Rollenbindungen in Kubernetes finden Sie unter Using RBAC Authorization (Verwenden der RBAC-Autorisierung) in der Kubernetes-Dokumentation. Sie können eines der folgenden Manifeste auf Ihren Cluster anwenden, die eine `role` und eine `rolebinding` oder eine `clusterrole` und eine `clusterrolebinding` mit den erforderlichen Kubernetes-Berechtigungen erstellen:

Kubernetes-Ressourcen in allen Namespaces anzeigen

- Der Gruppenname in der Datei lautet `eks-console-dashboard-full-access-group`. Wenden Sie das Manifest mit dem folgenden Befehl auf Ihren Cluster an:

```
kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-full-access.yaml
```

Kubernetes-Ressourcen in einem bestimmten Namespace anzeigen

- Der Namespace in dieser Datei ist `default`. Der Gruppenname in der Datei lautet `eks-console-dashboard-restricted-access-group`. Wenden Sie das Manifest mit dem folgenden Befehl auf Ihren Cluster an:

```
kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-restricted-access.yaml
```

Wenn Sie den Namen, den Namespace, die Berechtigungen oder eine andere Konfiguration in der Datei der Kubernetes-Gruppe ändern müssen, laden Sie die Datei herunter und bearbeiten Sie sie, bevor Sie sie auf Ihren Cluster anwenden:

a. Laden Sie die Datei mit einem der folgenden Befehle herunter.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-full-access.yaml
```

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-restricted-access.yaml
```

b. Bearbeiten Sie die Datei nach Bedarf.

c. Wenden Sie das Manifest mit einem der folgenden Befehle auf Ihren Cluster an:

```
kubectl apply -f eks-console-full-access.yaml
```

```
kubectl apply -f eks-console-restricted-access.yaml
```

3. Ordnen Sie den IAM-Prinzipal dem Kubernetes-Benutzer oder der Kubernetes-Gruppe in `aws-auth ConfigMap` zu. Sie können ein Tool wie `eksctl` verwenden, um die `ConfigMap` zu aktualisieren, oder Sie können sie durch manuelle Bearbeitung aktualisieren.

> **Important**
>
> Wir empfehlen die Verwendung von `eksctl` oder einem anderen Tool, um die `ConfigMap` zu bearbeiten. Informationen zu anderen Tools, die Sie verwenden können, finden Sie unter Use tools to make changes to the aws- authConfigMap in den Amazon EKS-Best-Practices-Handbüchern. Ist `aws-auth-ConfigMap` falsch formatiert, können Sie den Zugriff auf Ihren Cluster verlieren.

Bearbeiten mit eksctl

1. Sie benötigen eine Version `0.215.0` oder höher des `eksctl` Befehlszeilentools, das auf Ihrem Gerät installiert ist oder AWS CloudShell. Informationen zum Installieren und Aktualisieren von `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.
2. Zeigen Sie die aktuellen Mappings in der `ConfigMap` an. Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters. *`region-code`*Ersetzen Sie durch die AWS Region, in der sich Ihr Cluster befindet.

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

Eine Beispielausgabe sieht wie folgt aus.

```
ARN
            USERNAME                                GROUPS
 ACCOUNT
arn:aws: iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
```

3. Fügen Sie ein Mapping für eine Rolle hinzu. In diesem Beispiel wird davon ausgegangen, dass Sie die IAM-Berechtigungen im ersten Schritt einer Rolle namens *`my-console-viewer-role`* zuweisen. Ersetzen Sie *`111122223333`* durch Ihre Konto-ID.

```
eksctl create iamidentitymapping \
    --cluster my-cluster \
    --region=region-code \
    --arn arn:aws: iam::111122223333:role/my-console-viewer-role \
    --group eks-console-dashboard-full-access-group \
    --no-duplicate-arns
```

> **Important**
>
> Der Rollen-ARN darf keinen Pfad wie `role/my-team/developers/my-role` enthalten. Das Format des ARN muss `arn:aws: iam::111122223333:role/my-role` sein. In diesem Beispiel muss `my-team/developers/` entfernt werden.

Eine Beispielausgabe sieht wie folgt aus.

```
[...]
2022-05-09 14:51:20 [#]  adding identity "arn:aws: iam::111122223333:role/my-console-
viewer-role" to auth ConfigMap
```

4. Fügen Sie ein Mapping für einen Benutzer hinzu. Bewährte Methoden für IAM empfehlen, dass Sie Rollen statt Benutzern Berechtigungen gewähren. In diesem Beispiel wird davon ausgegangen, dass Sie die IAM-Berechtigungen im ersten Schritt einem Benutzer *my-user* zuweisen. Ersetzen Sie *111122223333* durch Ihre Konto-ID.

```
eksctl create iamidentitymapping \
    --cluster my-cluster \
    --region=region-code \
    --arn arn:aws: iam::111122223333:user/my-user \
    --group eks-console-dashboard-restricted-access-group \
    --no-duplicate-arns
```

Eine Beispielausgabe sieht wie folgt aus.

```
[...]
2022-05-09 14:53:48 [#]  adding identity "arn:aws: iam::111122223333:user/my-user" to
 auth ConfigMap
```

5. Zeigen Sie wieder die Mappings in der `ConfigMap` an.

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

Eine Beispielausgabe sieht wie folgt aus.

```
ARN
           USERNAME                                      GROUPS
      ACCOUNT
arn:aws: iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
arn:aws: iam::111122223333:role/my-console-viewer-role
                                                        eks-console-dashboard-full-
access-group
arn:aws: iam::111122223333:user/my-user
                                                     eks-console-dashboard-restricted-
access-group
```

ConfigMap Manuell bearbeiten

Weitere Informationen zum Hinzufügen von Benutzern oder Rollen zur `aws-auth` `ConfigMap` finden Sie unter the section called “Hinzufügen von IAM-Prinzipal zu Ihrem Amazon EKS-Cluster”.

1. Öffnen Sie die `aws-auth` `ConfigMap` zum Bearbeiten.

   ```
   kubectl edit -n kube-system configmap/aws-auth
   ```

2. Fügen Sie die Zuordnungen zur `aws-auth`-`ConfigMap` hinzu, ersetzen Sie jedoch keine der vorhandenen Zuordnungen. Im folgenden Beispiel werden Zuordnungen zwischen IAM-Prinzipalen mit Berechtigungen hinzugefügt, die im ersten Schritt hinzugefügt wurden, sowie die im vorherigen Schritt erstellten Kubernetes-Gruppen:

   - Die *`my-console-viewer-role`*-Rolle und `eks-console-dashboard-full-access-group`.
   - Der *`my-user`*-Benutzer und `eks-console-dashboard-restricted-access-group`.

     In diesen Beispielen wird davon ausgegangen, dass Sie die IAM-Berechtigungen im ersten Schritt einer Rolle namens *`my-console-viewer-role`* und einem Benutzer namens *`my-user`* zuweisen. Ersetze es *`111122223333`* durch deine AWS Konto-ID.

```
apiVersion: v1
data:
mapRoles: |
  - groups:
    - eks-console-dashboard-full-access-group
    rolearn: arn:aws: iam::111122223333:role/my-console-viewer-role
    username: my-console-viewer-role
mapUsers: |
  - groups:
    - eks-console-dashboard-restricted-access-group
    userarn: arn:aws: iam::111122223333:user/my-user
    username: my-user
```

> **Important**
>
> Der Rollen-ARN darf keinen Pfad wie `role/my-team/developers/my-console-viewer-role` enthalten. Das Format des ARN muss `arn:aws:iam::111122223333:role/my-console-viewer-role` sein. In diesem Beispiel muss `my-team/developers/` entfernt werden.

3. Speichern Sie die Datei und beenden Sie den Text-Editor.

kubectl mit einem EKS-Cluster durch Erstellen einer kubeconfig-Datei verbinden

> **Tip**
>
> Melden Sie sich für bevorstehende Amazon EKS-Workshops an.

In diesem Thema erstellen Sie eine `kubeconfig`-Datei für Ihren Cluster (oder aktualisieren eine vorhandene).

Das `kubectl`-Befehlszeilentool verwendet Konfigurationsinformationen in `kubeconfig`-Dateien für die Kommunikation mit dem API-Server eines Clusters. Weitere Informationen finden Sie unter Organisieren des Cluster-Zugriffs mit kubeconfig-Dateien in der Kubernetes-Dokumentation.

Amazon EKS verwendet den Befehl `aws eks get-token` mit `kubectl` für die Cluster-Authentifizierung. Standardmäßig verwendet die AWS CLI dieselben Anmeldeinformationen, die mit dem folgenden Befehl zurückgegeben werden:

```
aws sts get-caller-identity
```

- Ein vorhandener Amazon-EKS-Cluster. Informationen zum Bereitstellen finden Sie unter Erste Schritte.
- Das `kubectl`-Befehlszeilen-Tool ist auf Ihrem Gerät oder in der AWS CloudShell installiert. Die Version kann mit der Kubernetes-Version Ihres Clusters identisch sein oder bis zu einer Nebenversion älter oder neuer sein. Wenn Ihre Clusterversion beispielsweise `1.29` ist, können Sie `kubectl`-Version `1.28`, `1.29`, oder `1.30` damit verwenden. Informationen zum Installieren oder Aktualisieren von `kubectl` finden Sie unter the section called “kubectl und eksctl einrichten”.
- Version `2.12.3` oder höher oder Version `1.27.160` oder höher der auf Ihrem Gerät installierten und konfigurierten AWS Befehlszeilenschnittstelle (AWS CLI) oder AWS CloudShell. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Paketmanager wie `yumapt-get`, oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der AWS CLI. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Die AWS CLI-Version, in der installiert ist, AWS CloudShell kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zur Aktualisierung finden Sie im AWS CloudShell Benutzerhandbuch unter AWS CLI in Ihrem Home-Verzeichnis installieren.
- Ein IAM-Benutzer oder eine Rolle mit Berechtigung zur Verwendung der API-Aktion `eks:DescribeCluster` für den angegebenen Cluster. Weitere Informationen finden Sie unter the section called “Identitätsbasierte Richtlinien”. Wenn Sie für den Zugriff auf Ihren Cluster eine Identität Ihres eigenen OpenID-Connect-Anbieters verwenden, lesen Sie den Abschnitt Verwendung von kubectl in der Kubernetes-Dokumentation, um Ihre `kube config`-Datei zu erstellen oder zu aktualisieren.

`kubeconfig`-Datei automatisch erstellen

- Version `2.12.3` oder höher oder Version `1.27.160` oder höher der auf Ihrem Gerät installierten und konfigurierten AWS Befehlszeilenschnittstelle (AWS CLI) oder AWS CloudShell. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Paketmanager wie `yumapt-get`, oder Homebrew für macOS liegen oft

mehrere Versionen hinter der neuesten Version der AWS CLI. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Die AWS CLI-Version, in der installiert ist, AWS CloudShell kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zur Aktualisierung finden Sie im AWS CloudShell Benutzerhandbuch unter AWS CLI in Ihrem Home-Verzeichnis installieren.

- Berechtigung zur Nutzung der `eks:DescribeCluster`-API-Aktion für den Cluster, den Sie angeben. Weitere Informationen finden Sie unter the section called “Identitätsbasierte Richtlinien”.

1. Erstellen oder aktualisieren Sie eine `kubeconfig`-Datei für Ihren Cluster. *`region-code`*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet, und *`my-cluster`* ersetzen Sie es durch den Namen Ihres Clusters.

   ```
   aws eks update-kubeconfig --region region-code --name my-cluster
   ```

   Standardmäßig wird die resultierende Konfigurationsdatei im Standard `kubeconfig`-Pfad (`.kube`) in Ihrem Stammverzeichnis erstellt oder mit einer vorhandenen `config`-Datei an diesem Speicherort zusammengeführt. Sie können mit der Option `--kubeconfig` einen anderen Pfad angeben.

   Sie können einen IAM-Rollen-ARN mit der Option `--role-arn` für die Authentifizierung verwenden, wenn Sie `kubectl`-Befehle ausgeben. Andernfalls wird der IAM-Prinzipal in Ihrer standardmäßigen AWS CLI- oder SDK-Anmeldeinformationskette verwendet. Sie können Ihre AWS Standard-CLI- oder SDK-Identität anzeigen, indem `aws sts get-caller-identity` Sie den Befehl ausführen.

   Für alle verfügbaren Optionen führen Sie den *`aws eks update-kubeconfig help`* Befehl aus oder sehen Sie sich update-kubeconfig in der AWS CLI-Befehlsreferenz an.

2. Testen Sie Ihre Konfiguration.

   ```
   kubectl get svc
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   NAME             TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
   svc/kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   1m
   ```

Wenn Sie Autorisierungs- oder Ressourcenfehler erhalten, finden Sie weitere Informationen unter the section called “Nicht autorisiert oder Zugriff verweigert (`kubectl`)” im Thema zur Fehlerbehebung.

Kubernetes-Workloads Zugriff auf AWS mithilfe von Kubernetes-Servicekonten gewähren

A Kubernetes service account provides an identity for processes that run in a Pod. For more information see Verwaltung von Servicekonten in the Kubernetes documentation. If your Pod needs access to AWS services, you can map the service account to an AWS Identity and Access Management identity to grant that access. For more information, see the section called “Anmeldeinformationen mit IRSA” or the section called “Pod Identity”.

Servicekonto-Tokens

Das Feature BoundServiceAccountTokenVolume ist in Kubernetes-Versionen standardmäßig aktiviert. Die Funktion verbessert die Sicherheit von Servicekonto-Tokens, indem sie es auf Kubernetes ausgeführten Workloads ermöglicht, JSON-Web-Tokens, die an Zielgruppen, Zeit und Schlüssel gebunden sind, anzufordern. Die Ablaufzeit für Servicekonto-Tokens beträgt eine Stunde. In früheren Kubernetes-Versionen hatten Token kein Ablaufdatum. Clients, die diese Tokens verwenden, müssen die Tokens nun also innerhalb einer Stunde aktualisieren. Die folgenden Kubernetes-Client-SKDs aktualisieren Tokens automatisch im erforderlichen Zeitrahmen:

- Go-Version `0.15.7` und höher
- Python-Version `12.0.0` und höher
- Java-Version `9.0.0` und höher
- JavaScript-Version `0.10.3` und höher
- Ruby `master`-Branch
- Haskell-Version `0.3.0.0`
- C#-Version `7.0.5` und höher

Wenn Ihre Workload eine frühere Client-Version verwendet, ist eine Aktualisierung erforderlich. Um Clients reibungslos zu neueren zeitlich begrenzten Servicekonto-Tokens zu migrieren, fügt Kubernetes eine verlängerte Ablaufzeit für Servicekonto-Tokens, welche die standardmäßige Stunde

übersteigt. Für Amazon-EKS-Cluster gilt eine verlängerte Ablaufzeit von 90 Tagen. Der Kubernetes-API-Server Ihres Amazon-EKS-Clusters lehnt Anfragen mit Token ab, die älter als 90 Tage sind. Sie sollten Ihre Anwendungen und ihre Abhängigkeiten überprüfen, um sicherzustellen, dass die Kubernetes-Client-SDKs mit den zuvor aufgeführten Versionen identisch oder höher sind.

Wenn der API-Server Anfragen mit Token erhält, die älter als eine Stunde sind, kommentiert er das Audit-Protokollereignis der API mit `annotations.authentication.k8s.io/stale-token`. Die Anmerkung sieht zum Beispiel wie folgt aus:

```
subject: system:serviceaccount:common:fluent-bit, seconds after warning threshold:
 4185802.
```

Wenn für Ihren Cluster die Steuerebenen-Protokollierung aktiviert ist, dann befinden sich die Anmerkungen in den Überwachungsprotokollen. Sie können die folgende Abfrage mit CloudWatch Logs Insights verwenden, um alle Pods in Ihrem Amazon-EKS-Cluster zu identifizieren, die veraltete Token verwenden:

```
fields @timestamp
|filter @logStream like /kube-apiserver-audit/
|filter @message like /seconds after warning threshold/
|parse @message "subject: *, seconds after warning threshold:*\"" as subject,
 elapsedtime
```

`subject` bezieht sich auf das Servicekonto, das der Pod verwendet hat. `elapsedtime` gibt die verstrichene Zeit (in Sekunden) nach dem Lesen des neuesten Tokens an. Die Anfragen an den API-Server werden abgelehnt, wenn `elapsedtime` 90 Tage (7 776 000 Sekunden) überschreitet. Sie sollten das Kubernetes-Client-SDK Ihrer Anwendungen proaktiv aktualisieren, um eine der zuvor aufgeführten Versionen zu verwenden, die das Token automatisch aktualisiert. Wenn das verwendete Servicekonto-Token fast 90 Tage erreicht hat, und Sie nicht genügend Zeit haben, Ihre Client-SDK-Versionen vor dem Ablauf des Tokens zu aktualisieren, können Sie die vorhandenen Pods beenden und neue erstellen. Dadurch wird das Servicekonto-Token erneut abgerufen, sodass Sie weitere 90 Tage Zeit haben, die SDKs Ihrer Client-Version zu aktualisieren.

Wenn der Pod Teil einer Bereitstellung ist, besteht die empfohlene Vorgehensweise zum Beenden der Pods und gleichzeitigen Aufrechterhaltung hoher Verfügbarkeit darin, mit dem folgenden Befehl eine Einführung durchzuführen. Ersetzen Sie *`my-deployment`* mit dem Namen Ihrer Bereitstellung.

```
kubectl rollout restart deployment/my-deployment
```

Cluster-Add-ons

Die folgenden Cluster-Add-Ons wurden für die Verwendung der Kubernetes-Client-SDKs aktualisiert, die Servicekonto-Token automatisch neu abrufen können. Wir empfehlen Ihnen, sicherzustellen, dass die aufgeführten Versionen oder neuere Versionen auf Ihrem Cluster installiert sind.

- Amazon-VPC-CNI-Plugin für Kubernetes und Metrik-Hilfs-Plugins Version `1.8.0` und höher. Zum Überprüfen oder Aktualisieren Ihrer aktuellen Version siehe the section called “Amazon VPC CNI” und cni-metrics-helper.
- CoreDNS-Version `1.8.4` und höher. Zum Überprüfen oder Aktualisieren Ihrer aktuellen Version siehe the section called “CoreDNS”.
- Version `2.0.0` von AWS Load Balancer Controller und höher. Zum Überprüfen oder Aktualisieren Ihrer aktuellen Version siehe the section called “ AWS Load Balancer Balancer-Controller”.
- Eine aktuelle `kube-proxy`-Version. Zum Überprüfen oder Aktualisieren Ihrer aktuellen Version siehe the section called “kube-proxy”.
- AWS für Fluent Bit-Version `2.25.0` oder höher. Informationen zum Aktualisieren Ihrer aktuellen Version finden Sie unter Releases auf GitHub:
- Fluentd-Imageversion 1.14.6-1.2 oder höher und Fluentd-Filter-Plugin für Kubernetes-Metadatenversion 2.11.1 oder höher.

Gewährung von AWS Identity and Access Management für Workloads in Clustern von Amazon Elastic Kubernetes Service

Amazon EKS bietet zwei Möglichkeiten, um Workloads, die in Amazon-EKS-Clustern ausgeführt werden, Berechtigungen für AWS Identity and Access Management zu gewähren: IAM-Rollen für Servicekonten und EKS Pod Identities.

IAM-Rollen für Servicekonten

> IAM-Rollen für Servicekonten (IRSA) konfiguriert Kubernetes-Anwendungen, die in AWS ausgeführt werden, mit detaillierten IAM-Berechtigungen für den Zugriff auf verschiedene andere AWS-Ressourcen wie Amazon S3-Buckets, Amazon DynamoDB-Tabellen und mehr. Sie können mehrere Anwendungen zusammen in demselben Amazon EKS-Cluster ausführen und sicherstellen, dass jede Anwendung nur über die erforderlichen Mindestberechtigungen verfügt. IRSA wurde entwickelt, um verschiedene von AWS unterstützte Kubernetes-Bereitstellungsoptionen zu unterstützen, wie z. B. Amazon EKS, Amazon EKS Anywhere, Red Hat

OpenShift Service in AWS und selbstverwaltete Kubernetes-Cluster auf Amazon-EC2-Instances. Daher wurde IRSA mithilfe grundlegender AWS-Services wie IAM erstellt und war nicht direkt vom Amazon EKS-Service und der EKS-API abhängig. Weitere Informationen finden Sie unter the section called “Anmeldeinformationen mit IRSA”.

EKS Pod Identities

EKS Pod Identity bietet Cluster-Administratoren einen vereinfachten Workflow zur Authentifizierung von Anwendungen für den Zugriff auf verschiedene andere AWS-Ressourcen wie Amazon S3-Buckets, Amazon DynamoDB-Tabellen und mehr. EKS Pod Identity ist nur für EKS vorgesehen und vereinfacht Cluster-Administratoren so die Konfiguration von Kubernetes-Anwendungen, um IAM-Berechtigungen zu erhalten. Diese Berechtigungen können jetzt direkt über die AWS-Managementkonsole, die EKS-API und AWS CLI ganz einfach mit weniger Schritten konfiguriert werden und es müssen keine Maßnahmen innerhalb des Clusters von Kubernetes-Objekten ergriffen werden. Cluster-Administratoren müssen nicht zwischen den EKS- und IAM-Services wechseln oder privilegierte IAM-Operationen verwenden, um die für Ihre Anwendungen erforderlichen Berechtigungen zu konfigurieren. IAM-Rollen können jetzt in mehreren Clustern verwendet werden, ohne dass bei der Erstellung neuer Cluster die Rollenvertrauensrichtlinie aktualisiert werden muss. Die von EKS Pod Identity bereitgestellten IAM-Anmeldeinformationen umfassen Rollensitzungs-Tags mit Attributen wie Cluster-Name, Namespace und dem Namen des Servicekontos. Rollensitzungs-Tags ermöglichen es Administratoren, eine einzelne Rolle zu erstellen, die für alle Servicekonten verwendet werden kann, indem sie den Zugriff auf AWS-Ressourcen auf der Grundlage übereinstimmender Tags ermöglicht. Weitere Informationen finden Sie unter the section called “Pod Identity”.

Vergleich von EKS Pod Identity und IRSA

Ganz allgemein ermöglichen es Ihnen sowohl EKS Pod Identity als auch IRSA, Anwendungen, die auf Kubernetes-Clustern ausgeführt werden, IAM-Berechtigungen zu gewähren. Sie unterscheiden sich jedoch grundlegend in ihrer Konfiguration, den unterstützten Grenzwerten und den aktivierten Features. Nachfolgend vergleichen wir einige der wichtigsten Aspekte beider Lösungen.

> **Note**
>
> AWS empfiehlt die Verwendung von EKS Pod Identities, um Ihren Pods nach Möglichkeit Zugriff auf AWS-Ressourcen zu gewähren. Weitere Informationen finden Sie unter the section called “Pod Identity”.

Attribut	EKS Pod Identity	IRSA
Erweiterbarkeit von Rollen	Sie müssen jede Rolle einmal einrichten, um das Vertrauen des neu eingeführten Amazon EKS-Service-Prinzipal zu erhalten `pods.eks.amazonaws.com` . Nach diesem einmaligen Schritt müssen Sie die Vertrauen srichtlinie der Rolle nicht jedes Mal aktualisieren, wenn sie in einem neuen Cluster verwendet wird.	Sie müssen die Vertrauen srichtlinie der IAM-Rolle jedes Mal mit dem neuen Endpunkt des EKS-Cluster-OIDC-A nbieters aktualisieren, wenn Sie die Rolle in einem neuen Cluster verwenden möchten.
Skalierbarkeit von Clustern	Bei EKS Pod Identity müssen Benutzer keinen IAM-OIDC-Anbieter einrichten. Dieses Limit gilt also nicht.	Jedem EKS-Cluster ist eine OpenID Connect (OIDC)-Au ssteller-URL zugeordnet. Um IRSA zu verwenden, muss für jeden EKS-Cluster in IAM ein eindeutiger OpenID-Connect-Anbieter erstellt werden. IAM hat ein globales Standardlimit von 100 OIDC-Anbietern für jedes AWS-Konto. Wenn Sie planen, mehr als 100 EKS-Cluster pro AWS-Konto mit IRSA zu haben, erreichen Sie das IAM-OIDC-Anbieterlimit.
Skalierbarkeit von Rollen	Bei EKS Pod Identity müssen Benutzer in der Vertrauen srichtlinie keine Vertrauen sbeziehung zwischen der IAM-Rolle und dem Servicekonto definieren. Dieses Limit gilt also nicht.	In IRSA muss die Vertrauen sbeziehung zwischen einer IAM-Rolle und einem Servicekonto in der Vertrauen srichtlinie der Rolle definiert werden. Standardmäßig beträgt die Größe der

Attribut	EKS Pod Identity	IRSA
		Vertrauensrichtlinie 2048. Das bedeutet, dass Sie in der Regel vier Vertrauen sbeziehungen in einer Vertrauensrichtlinie definiere n können. Sie können zwar das Limit für die maximale Größe der Vertrauensrichtlin ie erhöhen, sind jedoch in der Regel auf maximal acht Vertrauensbeziehungen innerhalb einer Vertrauen srichtlinie beschränkt.
STS-API-Kontingentnutzung	EKS Pod Identity vereinfacht die Übermittlung von AWS-Anmeldeinformationen an Ihre Pods und erfordert nicht, dass Ihr Code direkt Aufrufe mit dem AWS Security Token Service (STS) durchführt. Der EKS-Service übernimmt die Rollenübernahme und übermittelt Anmeldein formationen an Anwendung en, die mit dem AWS-SDK in Ihren Pods geschrieben wurden, ohne dass Ihre Pods mit AWS STS kommunizieren oder STS-API-Kontingente verwenden müssen.	In IRSA verwenden Anwendungen, die mit dem AWS-SDK in Ihren Pods geschrieben wurden, Tokens, um die `AssumeRol eWithWebIdentity` -API auf dem AWS Security Token Service (STS) aufzurufe n. Abhängig von der Logik Ihres Codes im AWS-SDK ist es möglich, dass Ihr Code unnötige Aufrufe an AWS STS tätigt und Drosselungsfehler erhält.

Attribut	EKS Pod Identity	IRSA
Wiederverwendbarkeit von Rollen	Die von EKS Pod Identity bereitgestellten, temporären AWS-STS-Anmeldeinformationen umfassen Rollensitzungs-Tags wie Cluster-Name, Namespace und den Namen des Servicekontos. Mithilfe von Rollensitzungs-Tags können Administratoren eine einzelne IAM-Rolle erstellen, die von mehreren Servicekonten mit unterschiedlichen effektiven Berechtigungen verwendet werden kann, indem auf Grundlage der ihnen zugewiesenen Tags Zugriff auf AWS-Ressourcen ermöglicht wird. Dies wird auch als attributbasierte Zugriffskontrolle (ABAC) bezeichnet. Weitere Informationen finden Sie unter the section called “Pods Zugriff gewähren”.	AWS-STS-Sitzungs-Tags werden nicht unterstützt. Sie können eine Rolle in verschiedenen Clustern wiederverwenden, aber jeder Pod erhält alle Berechtigungen der Rolle.
Unterstützte Umgebungen	EKS Pod Identity ist nur in Amazon EKS verfügbar.	IRSA kann beispielsweise für Amazon EKS, Amazon EKS Anywhere, Red Hat OpenShift Service in AWS und selbstverwaltete Kubernetes-Cluster in Amazon-EC2-Instances verwendet werden.

Attribut	EKS Pod Identity	IRSA
Unterstützte EKS-Versionen	Alle unterstützten EKS-Cluster-Versionen. Informationen über die jeweiligen Plattformversionen finden Sie unter the section called “Cluster-Versionen von EKS Pod Identity”.	Alle unterstützten EKS-Cluster-Versionen.

IAM-Rollen für Servicekonten

> **Tip**
>
> Melden Sie sich für bevorstehende Amazon EKS-Workshops an.

Anwendungen in den Containern eines Pods können ein AWS SDK oder die AWS CLI verwenden, um API-Anfragen an AWS Dienste mithilfe von AWS Identity and Access Management (IAM) -Berechtigungen zu stellen. Anwendungen müssen ihre AWS API-Anfragen mit AWS Anmeldeinformationen signieren. IAM-Rollen für Servicekonten (IRSA) bieten die Möglichkeit, Anmeldeinformationen für Ihre Anwendungen zu verwalten, ähnlich wie EC2 Amazon-Instance-Profile Anmeldeinformationen für Amazon-Instances bereitstellen. EC2 Anstatt Ihre AWS Anmeldeinformationen zu erstellen und an die Container zu verteilen oder die Rolle der EC2 Amazon-Instance zu verwenden, verknüpfen Sie eine IAM-Rolle mit einem Kubernetes-Dienstkonto und konfigurieren Ihre Pods so, dass sie das Dienstkonto verwenden. Sie können IAM-Rollen nicht für Dienstkonten mit lokalen Clustern für Amazon EKS auf AWS Outposts verwenden.

IAM-Rollen für Servicekonten bietet die folgenden Vorteile:

- Geringste Berechtigung – Sie können IAM-Berechtigungen auf ein Servicekonto beschränken, und nur Pods, die dieses Servicekonto verwenden, haben Zugriff auf diese Berechtigungen. Mit diesem Feature entfällt auch die Notwendigkeit von Drittanbieterlösungen wie `kiam` oder `kube2iam`.
- Isolierung von Anmeldeinformationen — Wenn der Zugriff auf den Amazon EC2 Instance Metadata Service (IMDS) eingeschränkt ist, können die Container eines Pods nur Anmeldeinformationen für die IAM-Rolle abrufen, die dem Dienstkonto zugeordnet ist, das der Container verwendet. Ein Container hat nie Zugriff auf Anmeldeinformationen, die von anderen Containern in anderen

Pods verwendet werden. Wenn IMDS nicht eingeschränkt ist, haben die Container des Pods auch Zugriff auf die IAM-Rolle des Amazon-EKS-Knotens und die Container können möglicherweise auf Anmeldeinformationen von IAM-Rollen anderer Pods auf demselben Knoten zugreifen. Weitere Informationen finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.

> **Note**
>
> Pods, `hostNetwork: true` die mit konfiguriert sind, haben immer IMDS-Zugriff, aber die AWS SDKs AND-CLI verwendet IRSA-Anmeldeinformationen, wenn sie aktiviert sind.

- Überprüfbarkeit — Zugriffs- und Ereignisprotokollierung sind über verfügbar, um eine nachträgliche Prüfung AWS CloudTrail zu gewährleisten.

> **Important**
>
> Container stellen keine Sicherheitsgrenze dar, und die Verwendung von IAM-Rollen für Servicekonten ändert daran nichts. Pods, die demselben Knoten zugewiesen sind, teilen sich einen Kernel und möglicherweise andere Ressourcen, abhängig von Ihrer Pod-Konfiguration. Während Pods, die auf separaten Knoten ausgeführt werden, auf der Rechenebene isoliert sind, gibt es Knotenanwendungen, die über zusätzliche Berechtigungen in der Kubernetes-API verfügen, die über den Umfang einer einzelnen Instance hinausgehen. Einige Beispiele sind `kubelet`, `kube-proxy`, CSI-Speichertreiber oder Ihre eigenen Kubernetes-Anwendungen.

Aktivieren Sie IAM-Rollen für Servicekonten, indem Sie die folgenden Verfahren ausführen:

1. IAM-OIDC-Anbieter für Ihren Cluster erstellen – Sie müssen diesen Vorgang nur einmal für einen Cluster durchführen.

   > **Note**
   >
   > Wenn Sie den EKS-VPC-Endpunkt aktiviert haben, kann von innerhalb dieser VPC nicht auf den EKS-OIDC-Service-Endpunkt zugegriffen werden. Folglich funktionieren Ihre Operationen wie das Erstellen eines OIDC-Anbieters mit `eksctl`

in der VPC nicht und führen zu einem Timeout, wenn Sie versuchen, `https://oidc.eks.region.amazonaws.com` anzufordern. Es folgt ein Beispiel für eine Fehlermeldung:

```
server cant find oidc.eks.region.amazonaws.com: NXDOMAIN
```

Um diesen Schritt abzuschließen, können Sie den Befehl außerhalb der VPC ausführen, z. B. in AWS CloudShell oder auf einem Computer, der mit dem Internet verbunden ist. Alternativ können Sie einen Split-Horizon-Conditional-Resolver in der VPC erstellen, z. B. Route 53 Resolver, um einen anderen Resolver für die OIDC-Aussteller-URL zu verwenden und dafür nicht das VPC-DNS zu nutzen. Ein Beispiel für bedingte Weiterleitung in CoreDNS finden Sie in der Amazon EKS-Funktionsanfrage unter. GitHub

2. IAM-Rollen Kubernetes-Servicekonten zuweisen – Führen Sie diesen Vorgang für jede einzelne Berechtigungsgruppe durch, über die eine Anwendung verfügen soll.
3. Pods so konfigurieren, dass sie ein Kubernetes-Dienstkonto verwenden — Führen Sie dieses Verfahren für jeden Pod durch, der Zugriff auf Dienste benötigt. AWS
4. IRSA mit dem AWS SDK verwenden — Vergewissern Sie sich, dass der Workload ein AWS SDK einer unterstützten Version verwendet und dass der Workload die standardmäßige Anmeldeinformationskette verwendet.

Hintergrundinformationen zu IAM, Kubernetes und OpenID Connect (OIDC)

2014 fügte AWS Identity and Access Management die Unterstützung für föderierte Identitäten mithilfe von OpenID Connect (OIDC) hinzu. Mit dieser Funktion können Sie AWS API-Aufrufe bei unterstützten Identitätsanbietern authentifizieren und ein gültiges OIDC-JSON-Webtoken (JWT) erhalten. Sie können dieses Token an den AWS `AssumeRoleWithWebIdentity` STS-API-Vorgang übergeben und temporäre IAM-Rollenanmeldedaten erhalten. Sie können diese Anmeldeinformationen verwenden, um mit jedem AWS Service zu interagieren, einschließlich Amazon S3 und DynamoDB.

Jedes JWT-Token ist mit einem Signaturschlüsselpaar signiert. Die Schlüssel werden für den von Amazon EKS verwalteten OIDC-Anbieter bereitgestellt und der private Schlüssel wechselt alle 7 Tage. Amazon EKS bewahrt die öffentlichen Schlüssel auf, bis sie ablaufen. Wenn Sie externe OIDC-Clients verbinden, beachten Sie, dass Sie die Signaturschlüssel aktualisieren müssen, bevor der öffentliche Schlüssel abläuft. Erfahren Sie, wie Sie Signaturschlüssel abrufen, um OIDC-Token zu validieren.

Kubernetes verwendet seit langem Servicekonten als eigenes internes Identitätssystem. Pods können sich beim Kubernetes-API-Server mithilfe eines automatisch gemounteten Tokens (ein Nicht-OIDC-JWT) authentifizieren, das nur der Kubernetes-API-Server validieren kann. Diese alten Servicekonto-Token laufen nicht ab und das Rotieren des Signaturschlüssels ist ein schwieriger Prozess. In der Kubernetes-Version `1.12` wurde Support für ein neues `ProjectedServiceAccountToken`-Feature hinzugefügt. Dieses Feature ist ein OIDC-JSON-Web-Token, das auch die Identität des Servicekontos enthält und eine konfigurierbare Zielgruppe unterstützt.

Amazon EKS hostet jetzt einen öffentlichen OIDC-Erkennungsendpunkt pro Cluster, der die Signaturschlüssel für die `ProjectedServiceAccountToken`-JSON-Web-Token enthält, sodass externe Systeme wie IAM die von Kubernetes ausgegebenen OIDC-Token validieren und akzeptieren können.

Erstellen Sie einen IAM-OIDC-Anbieter für Ihren Cluster

Ihrem Cluster ist eine OpenID Connect (OIDC)-Aussteller-URL zugeordnet. Um AWS Identity and Access Management (IAM) -Rollen für Dienstkonten verwenden zu können, muss ein IAM-OIDC-Anbieter für die OIDC-Aussteller-URL Ihres Clusters vorhanden sein.

Voraussetzungen

- Ein vorhandener Amazon-EKS-Cluster. Informationen zum Bereitstellen finden Sie unter Erste Schritte.
- Version `2.12.3` oder höher oder Version `1.27.160` oder höher der auf Ihrem Gerät installierten und konfigurierten AWS Befehlszeilenschnittstelle (AWS CLI) oder AWS CloudShell. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Paketmanager wie `yum` `apt-get`, oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der AWS CLI. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Die AWS CLI-Version, in der installiert ist, AWS CloudShell kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zur Aktualisierung finden Sie im AWS CloudShell Benutzerhandbuch unter AWS CLI in Ihrem Home-Verzeichnis installieren.
- Das `kubectl`-Befehlszeilen-Tool ist auf Ihrem Gerät oder in der AWS CloudShell installiert. Die Version kann mit der Kubernetes-Version Ihres Clusters identisch sein oder bis zu einer Nebenversion älter oder neuer sein. Wenn Ihre Clusterversion beispielsweise `1.29` ist, können Sie

`kubectl`-Version `1.28`, `1.29`, oder `1.30` damit verwenden. Informationen zum Installieren oder Aktualisieren von `kubectl` finden Sie unter the section called “kubectl und eksctl einrichten”.

- Eine vorhandene `kubectl config`-Datei, die Ihre Clusterkonfiguration enthält. Informationen zum Erstellen einer `kubectl config`-Datei finden Sie unter the section called “Zugriff auf den Cluster mit kubectl”.

Sie können einen IAM-OIDC-Anbieter für Ihren Cluster mit `eksctl` oder AWS-Managementkonsole erstellen.

OIDC-Anbieter erstellen (eksctl)

1. Version `0.215.0` oder höher des `eksctl`-Befehlszeilen-Tools, das auf Ihrem Computer oder in der AWS CloudShell installiert ist. Informationen zum Installieren und Aktualisieren von `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.
2. Ermitteln Sie die OIDC-Aussteller-ID für Ihren Cluster.

   Rufen Sie die OIDC-Aussteller-ID Ihres Clusters ab und speichern Sie sie in einer Variable. Ersetzen Sie `<my-cluster>` durch Ihren eigenen Wert.

   ```
   cluster_name=<my-cluster>
   oidc_id=$(aws eks describe-cluster --name $cluster_name --query
    "cluster.identity.oidc.issuer" --output text | cut -d '/' -f 5)
   echo $oidc_id
   ```

3. Bestimmen Sie, ob ein IAM-OIDC-Anbieter mit der Aussteller-ID Ihres Clusters bereits in Ihrem Konto vorhanden ist.

   ```
   aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
   ```

   Wenn die Ausgabe zurückgegeben wird, verfügen Sie bereits über einen IAM-OIDC-Anbieter für Ihren Cluster und können den nächsten Schritt überspringen. Wenn keine Ausgabe erfolgt, müssen Sie einen IAM-OIDC-Anbieter für Ihr Cluster erstellen.

4. Erstellen Sie einen IAM-OIDC-Identitätsanbieter für Ihren Cluster mit dem folgenden Befehl.

   ```
   eksctl utils associate-iam-oidc-provider --cluster $cluster_name --approve
   ```

> **Note**
>
> Wenn Sie den EKS-VPC-Endpunkt aktiviert haben, kann von innerhalb dieser VPC nicht auf den EKS-OIDC-Service-Endpunkt zugegriffen werden. Folglich funktionieren Ihre Operationen wie das Erstellen eines OIDC-Anbieters mit `eksctl` in der VPC nicht und führen zu einem Timeout. Es folgt ein Beispiel für eine Fehlermeldung:
>
> ```
> ** server cant find oidc.eks.<region-code>.amazonaws.com: NXDOMAIN
> ```

Um diesen Schritt abzuschließen, können Sie den Befehl außerhalb der VPC ausführen, z. B. in AWS CloudShell oder auf einem Computer, der mit dem Internet verbunden ist. Alternativ können Sie einen Split-Horizon-Conditional-Resolver in der VPC erstellen, z. B. Route 53 Resolver, um einen anderen Resolver für die OIDC-Aussteller-URL zu verwenden und dafür nicht das VPC-DNS zu nutzen. Ein Beispiel für bedingte Weiterleitung in CoreDNS finden Sie in der Amazon EKS-Funktionsanfrage unter. GitHub

OIDC-Anbieter erstellen (Konsole)AWS

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Bereich Cluster aus und wählen Sie dann den Namen Ihres Clusters auf der Seite Cluster.
3. Notieren Sie im Abschnitt Details der Registerkarte Overview (Übersicht) den Wert der OpenID-Connect-Provider-URL.
4. Öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
5. Wählen Sie im linken Navigationsbereich Identity Providers (Identitätsanbieter) unter Access management (Zugriffsverwaltung) aus. Wenn ein Anbieter aufgeführt ist, der mit der URL für Ihren Cluster übereinstimmt, haben Sie bereits einen Anbieter für Ihren Cluster. Wenn kein Anbieter aufgeführt ist, der mit der URL für Ihren Cluster übereinstimmt, müssen Sie einen erstellen.
6. Um einen Anbieter zu erstellen, wählen Sie Add provider (Anbieter hinzufügen) aus.
7. Wählen Sie als Anbietertyp die Option OpenID Connect aus.
8. Geben Sie für Anbieter URL die OIDC-Anbieter-URL für Ihren Cluster ein.
9. Geben Sie für Zielgruppe `sts.amazonaws.com` ein

10. (Optional) Fügen Sie beliebige Tags hinzu, beispielsweise ein Tag, um zu identifizieren, welcher Cluster für diesen Anbieter ist.
11. Wählen Sie Add provider (Anbieter hinzufügen) aus.

Nächster Schritt: the section called “IAM-Rolle zuweisen”

IAM-Rollen Kubernetes-Servicekonten zuweisen

In diesem Thema wird beschrieben, wie Sie ein Kubernetes-Dienstkonto so konfigurieren, dass es eine AWS Identity and Access Management (IAM) -Rolle übernimmt. Alle Pods, die für die Verwendung des Dienstkontos konfiguriert sind, können dann auf jeden AWS Dienst zugreifen, für den die Rolle Zugriffsberechtigungen besitzt.

Voraussetzungen

- Einen vorhandenen -Cluster. Falls Sie noch kein Service-Konto haben, können Sie eines erstellen, indem Sie einer der Anleitungen in Erste Schritte folgen.
- Ein vorhandener Anbieter für IAM OpenID Connect (OIDC) für Ihren Cluster. Informationen zum Feststellen, ob Sie bereits einen haben oder wie Sie einen erstellen können, finden Sie unter the section called “IAM-OIDC-Anbieter”.
- Version `2.12.3` oder höher oder Version `1.27.160` oder höher der auf Ihrem Gerät installierten und konfigurierten AWS Befehlszeilenschnittstelle (AWS CLI) oder AWS CloudShell. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Paketmanager wie `yum` `apt-get`, oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der AWS CLI. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Die AWS CLI-Version, in der installiert ist, AWS CloudShell kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zur Aktualisierung finden Sie im AWS CloudShell Benutzerhandbuch unter AWS CLI in Ihrem Home-Verzeichnis installieren.
- Das `kubectl`-Befehlszeilen-Tool ist auf Ihrem Gerät oder in der AWS CloudShell installiert. Die Version kann mit der Kubernetes-Version Ihres Clusters identisch sein oder bis zu einer Nebenversion älter oder neuer sein. Wenn Ihre Clusterversion beispielsweise `1.29` ist, können Sie `kubectl`-Version `1.28`, `1.29`, oder `1.30` damit verwenden. Informationen zum Installieren oder Aktualisieren von `kubectl` finden Sie unter the section called “kubectl und eksctl einrichten”.

- Eine vorhandene `kubectl config`-Datei, die Ihre Clusterkonfiguration enthält. Informationen zum Erstellen einer `kubectl config`-Datei finden Sie unter the section called “Zugriff auf den Cluster mit kubectl”.

Schritt 1: IAM-Richtlinie erstellen

Wenn Sie Ihrer IAM-Rolle eine vorhandene IAM-Richtlinie zuordnen möchten, fahren Sie mit dem nächsten Schritt fort.

1. Erstellen Sie eine IAM-Richtlinie. Sie können Ihre eigene Richtlinie erstellen oder eine AWS verwaltete Richtlinie kopieren, die bereits einige der benötigten Berechtigungen gewährt, und sie an Ihre spezifischen Anforderungen anpassen. Weitere Informationen finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.
2. Erstellen Sie eine Datei, die die Berechtigungen für die AWS Dienste enthält, auf die Ihre Pods zugreifen sollen. Eine Liste aller Aktionen für alle AWS Dienste finden Sie in der Referenz zur Dienstautorisierung.

   Sie können den folgenden Befehl ausführen, um eine Beispiel-Richtliniendatei zu erstellen, die schreibgeschützten Zugriff auf einen Amazon-S3-Bucket gewährt. Sie können optional Konfigurationsinformationen oder ein Bootstrap-Skript in diesem Bucket speichern und die Container in Ihrem Pod können die Datei aus dem Bucket lesen und in Ihre Anwendung laden. Wenn Sie diese Beispielrichtlinie erstellen möchten, kopieren Sie den folgenden Inhalt auf Ihr Gerät. *`my-pod-secrets-bucket`*Ersetzen Sie es durch Ihren Bucket-Namen und führen Sie den Befehl aus.

   ```
   {
       "Version":"2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": "s3:GetObject",
               "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
           }
       ]
   }
   ```

3. Erstellen Sie die IAM-Richtlinie.

   ```
   aws iam create-policy --policy-name my-policy --policy-document file://my-policy.json
   ```

Schritt 2: IAM-Rolle erstellen und zuordnen

Erstellen Sie eine IAM-Rolle und ordnen Sie diese einem Kubernetes-Servicekonto zu. Sie können entweder `eksctl` oder die AWS CLI verwenden.

Rolle erstellen und zuordnen (eksctl)

Dieser `eksctl`-Befehl erstellt ein Kubernetes-Servicekonto im angegebenen Namespace, erstellt eine IAM-Rolle (sofern diese nicht bereits vorhanden ist) mit dem angegebenen Namen, fügt der Rolle eine vorhandene IAM-Richtlinie-ARN hinzu und annotiert das Servicekonto mit der IAM-Rollen-ARN. Denken Sie daran, die Beispiel-Platzhalterwerte in diesem Befehl durch Ihre spezifischen Werte zu ersetzen. Informationen zum Installieren und Aktualisieren von `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.

```
eksctl create iamserviceaccount --name my-service-account --namespace default --cluster
 my-cluster --role-name my-role \
    --attach-policy-arn arn:aws: iam::111122223333:policy/my-policy --approve
```

> **Important**
>
> Wenn die Rolle oder das Servicekonto bereits vorhanden ist, schlägt der vorherige Befehl möglicherweise fehl. `eksctl` hat verschiedene Optionen, die Sie in diesen Situationen angeben können. Führen Sie `eksctl create iamserviceaccount --help` aus, um weitere Informationen zu erhalten.

Rolle erstellen und zuordnen (AWS CLI)

Wenn Sie ein vorhandenes Kubernetes-Servicekonto haben, das eine IAM-Rolle annehmen soll, können Sie diesen Schritt überspringen.

1. Erstellen Sie ein Kubernetes-Servicekonto. Kopieren Sie den folgenden Inhalt auf Ihr Gerät. *`my-service-account`*Ersetzen Sie ihn durch Ihren gewünschten Namen und gegebenenfalls *`default`* durch einen anderen Namespace. Wenn Sie eine Änderung vornehmen*`default`*, muss der Namespace bereits existieren.

   ```
   cat >my-service-account.yaml <<EOF
   apiVersion: v1
   kind: ServiceAccount
   metadata:
   ```

```
  name: my-service-account
  namespace: default
EOF
kubectl apply -f my-service-account.yaml
```

2. Setzen Sie Ihre AWS Konto-ID mit dem folgenden Befehl auf eine Umgebungsvariable.

```
account_id=$(aws sts get-caller-identity --query "Account" --output text)
```

3. Legen Sie den OIDC-Identitätsanbieter Ihres Clusters mit dem folgenden Befehl auf eine Umgebungsvariable fest. Ersetzen Sie *my-cluster* mit dem Namen Ihres Clusters.

```
oidc_provider=$(aws eks describe-cluster --name my-cluster --region $AWS_REGION --
query "cluster.identity.oidc.issuer" --output text | sed -e "s/^https:\/\///")
```

4. Legen Sie Variablen für den Namespace und den Namen des Servicekontos fest. *my-service-account*Ersetzen Sie es durch das Kubernetes-Dienstkonto, das Sie die Rolle übernehmen möchten. Ersetzen Sie es *default* durch den Namespace des Dienstkontos.

```
export namespace=default
export service_account=my-service-account
```

5. Führen Sie den folgenden Befehl aus, um eine Vertrauensrichtlinie für die IAM-Rolle zu erstellen. Wenn Sie allen Servicekonten innerhalb eines Namespaces die Verwendung der Rolle erlauben möchten, kopieren Sie den folgenden Inhalt auf Ihr Gerät. Ersetze *StringEquals* durch `StringLike` und ersetze *$service_account* durch`*`. Sie können mehrere Einträge in den Bedingungen `StringEquals` und `StringLike` unten hinzufügen, um mehreren Servicekonten oder Namespaces das Annehmen der Rolle zu erlauben. Weitere Informationen dazu, wie Sie zulassen können, dass Rollen von einem anderen AWS Konto als dem Konto, in dem sich Ihr Cluster befindet, diese Rolle übernehmen können, finden Sie unterthe section called “Kontoübergreifendes IAM”.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::123456789012:oidc-provider/$oidc_provider"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
```

```
      "Condition": {
        "StringEquals": {
          "$oidc_provider:aud": "sts.amazonaws.com",
          "$oidc_provider:sub": "system:serviceaccount:$namespace:$service_account"
        }
      }
    }
  ]
}
```

6. Erstellen Sie die Rolle. Ersetzen Sie *my-role* durch einen Namen für Ihre IAM-Rolle und *my-role-description* durch eine Beschreibung für Ihre Rolle.

```
aws iam create-role --role-name my-role --assume-role-policy-document file://trust-
relationship.json --description "my-role-description"
```

7. Hängen Sie eine IAM-Richtlinie an Ihre Rolle an. Ersetzen Sie *my-role* durch den Namen Ihrer IAM-Rolle und *my-policy* durch den Namen einer vorhandenen Richtlinie, die Sie erstellt haben.

```
aws iam attach-role-policy --role-name my-role --policy-arn=arn:aws: iam::
$account_id:policy/my-policy
```

8. Annotieren Sie Ihr Servicekonto mit dem Amazon-Ressourcennamen (ARN) der IAM-Rolle, die das Servicekonto annehmen soll. Ersetzen Sie *my-role* durch den Namen Ihrer bestehenden IAM-Rolle. Angenommen, Sie haben in einem vorherigen Schritt zugelassen, dass eine Rolle von einem anderen AWS Konto als dem Konto, in dem sich Ihr Cluster befindet, diese Rolle übernimmt. Stellen Sie dann sicher, dass Sie das AWS Konto und die Rolle des anderen Kontos angeben. Weitere Informationen finden Sie unter the section called “Kontoübergreifendes IAM”.

```
kubectl annotate serviceaccount -n $namespace $service_account eks.amazonaws.com/
role-arn=arn:aws: iam::$account_id:role/my-role
```

9. (Optional) Konfigurieren Sie den AWS Security Token Service-Endpunkt für ein Dienstkonto. AWS empfiehlt die Verwendung eines regionalen AWS STS-Endpunkts anstelle des globalen Endpunkts. Dies reduziert die Latenz, bietet integrierte Redundanz und erhöht die Gültigkeit der Sitzungstoken.

Schritt 3: Konfiguration bestätigen

1. Stellen Sie sicher, dass die Vertrauensrichtlinie der IAM-Rolle korrekt konfiguriert ist.

```
aws iam get-role --role-name my-role --query Role.AssumeRolePolicyDocument
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/oidc.eks.us-
east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.us-east-1.amazonaws.com/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:default:my-service-
account",
                    "oidc.eks.us-east-1.amazonaws.com/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
                }
            }
        }
    ]
}
```

2. Stellen Sie sicher, dass die Richtlinie, die Sie in einem vorherigen Schritt an Ihre Rolle angehängt haben, an die Rolle angehängt ist.

```
aws iam list-attached-role-policies --role-name my-role --query
 "AttachedPolicies[].PolicyArn" --output text
```

Eine Beispielausgabe sieht wie folgt aus.

```
 arn:aws: iam::111122223333:policy/my-policy
```

3. Legen Sie eine Variable fest, um den Amazon-Ressourcennamen (ARN) der Richtlinie zu speichern, die Sie verwenden möchten. *my-policy*Ersetzen Sie ihn durch den Namen der Richtlinie, für die Sie Berechtigungen bestätigen möchten.

```
export policy_arn=arn:aws: iam::111122223333:policy/my-policy
```

4. Zeigen Sie die Standardversion der Richtlinie an.

```
aws iam get-policy --policy-arn $policy_arn
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "EXAMPLEBIOWGLDEXAMPLE",
        "Arn": "arn:aws: iam::111122223333:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        [...]
    }
}
```

5. Überprüfen Sie den Inhalt der Richtlinie, um sicherzustellen, dass sie alle Berechtigungen enthält, die Ihr Pod benötigt. Falls erforderlich, ersetzen Sie *1* den folgenden Befehl durch die Version, die in der vorherigen Ausgabe zurückgegeben wurde.

```
aws iam get-policy-version --policy-arn $policy_arn --version-id v1
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
```

Wenn Sie die Beispielrichtlinie in einem vorherigen Schritt erstellt haben, ist Ihre Ausgabe dieselbe. Wenn Sie eine andere Richtlinie erstellt haben, ist der *example* Inhalt anders.

6. Stellen Sie sicher, dass das Kubernetes-Servicekonto mit der Rolle annotiert ist.

```
kubectl describe serviceaccount my-service-account -n default
```

Eine Beispielausgabe sieht wie folgt aus.

```
Name:                my-service-account
Namespace:           default
Annotations:         eks.amazonaws.com/role-arn: arn:aws: iam::111122223333:role/my-
role
Image pull secrets:  <none>
Mountable secrets:   my-service-account-token-qqjfl
Tokens:              my-service-account-token-qqjfl
[...]
```

Nächste Schritte

- the section called “An Pod zuweisen”

Konfiguration von Pods für die Verwendung eines Kubernetes-Servicekontos

Wenn ein Pod auf AWS Dienste zugreifen muss, müssen Sie ihn für die Verwendung eines Kubernetes-Dienstkontos konfigurieren. Das Dienstkonto muss einer AWS Identity and Access Management (IAM) -Rolle zugeordnet sein, die über Berechtigungen für den Zugriff auf die AWS Dienste verfügt.

- Einen vorhandenen -Cluster. Wenn Sie keine Rolle haben, können Sie sie mithilfe einer der Anleitungen in Erste Schritte erstellen.
- Ein vorhandener Anbieter für IAM OpenID Connect (OIDC) für Ihren Cluster. Informationen zum Feststellen, ob Sie bereits einen haben oder wie Sie einen erstellen können, finden Sie unter the section called “IAM-OIDC-Anbieter”.
- Ein vorhandenes Kubernetes-Servicekonto, das einer IAM-Rolle zugeordnet ist. Das Servicekonto muss mit dem Amazon-Ressourcennamen (ARN) der IAM-Rolle versehen sein. Der Rolle muss eine IAM-Richtlinie zugeordnet sein, die die Berechtigungen enthält, die Ihre Pods für die Nutzung AWS von Diensten haben sollen. Weitere Informationen zu Rollen, ihren Vorteilen sowie zu ihrer Erstellung und Konfiguration finden Sie unter the section called “IAM-Rolle zuweisen”.

- Version `2.12.3` oder höher oder Version `1.27.160` oder höher der auf Ihrem Gerät installierten und konfigurierten AWS Befehlszeilenschnittstelle (AWS CLI) oder AWS CloudShell. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Paketmanager wie `yum` `apt-get`, oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der AWS CLI. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Die AWS CLI-Version, in der installiert ist, AWS CloudShell kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zur Aktualisierung finden Sie im AWS CloudShell Benutzerhandbuch unter AWS CLI in Ihrem Home-Verzeichnis installieren.
- Das `kubectl`-Befehlszeilen-Tool ist auf Ihrem Gerät oder in der AWS CloudShell installiert. Die Version kann mit der Kubernetes-Version Ihres Clusters identisch sein oder bis zu einer Nebenversion älter oder neuer sein. Wenn Ihre Clusterversion beispielsweise `1.29` ist, können Sie `kubectl`-Version `1.28`, `1.29`, oder `1.30` damit verwenden. Informationen zum Installieren oder Aktualisieren von `kubectl` finden Sie unter the section called “kubectl und eksctl einrichten”.
- Eine vorhandene `kubectl config`-Datei, die Ihre Clusterkonfiguration enthält. Informationen zum Erstellen einer `kubectl config`-Datei finden Sie unter the section called “Zugriff auf den Cluster mit kubectl”.

  1. Verwenden Sie den folgenden Befehl, um ein Bereitstellungs-Manifest zu erstellen, mit dem Sie einen Pod bereitstellen können, um die Konfiguration zu bestätigen. Ersetzen Sie die Beispielwerte durch eigene Werte.

```
cat >my-deployment.yaml <<EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      serviceAccountName: my-service-account
      containers:
      - name: my-app
```

```
        image: public.ecr.aws/nginx/nginx:X.XX
EOF
```

2. Stellen Sie das Manifest in Ihrem Cluster bereit.

```
kubectl apply -f my-deployment.yaml
```

3. Vergewissern Sie sich, dass die erforderlichen Umgebungsvariablen für Ihren Pod vorhanden sind.

   a. Zeigen Sie die Pods an, die im vorherigen Schritt bereitgestellt wurden.

   ```
   kubectl get pods | grep my-app
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   my-app-6f4dfff6cb-76cv9   1/1     Running   0          3m28s
   ```

   b. Zeigen Sie den ARN der IAM-Rolle an, den der Pod verwendet.

   ```
   kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep AWS_ROLE_ARN:
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   AWS_ROLE_ARN: arn:aws: iam::111122223333:role/my-role
   ```

   Der Rollen-ARN muss mit dem Rollen-ARN übereinstimmen, mit dem Sie das vorhandene Servicekonto mit Anmerkungen versehen haben. Weitere Informationen zum Annotieren des Servicekontos finden Sie unter the section called “IAM-Rolle zuweisen”.

   c. Bestätigen Sie, dass der Pod über einen Web-Identitätstoken-Datei-Mount verfügt.

   ```
   kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep AWS_WEB_IDENTITY_TOKEN_FILE:
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   AWS_WEB_IDENTITY_TOKEN_FILE:  /var/run/secrets/eks.amazonaws.com/serviceaccount/
   token
   ```

Das `kubelet` fordert das Token im Namen des Pods an und speichert es. Standardmäßig aktualisiert das `kubelet` das Token, wenn es älter als 80 Prozent seiner gesamten TTL ist oder wenn das Token älter als 24 Stunden ist. Sie können die Ablaufdauer für jedes Konto mit Ausnahme des Standardservicekontos mit den Einstellungen in Ihrer Pod-Spezifikation ändern. Weitere Informationen finden Sie unter Service Account Token Volume Projection in der Kubernetes-Dokumentation.

Der Pod-Identity-Webhook von Amazon EKS auf dem Cluster sucht nach Pods, die ein Servicekonto mit der folgenden Annotation verwenden:

```
eks.amazonaws.com/role-arn: arn:aws: iam::111122223333:role/my-role
```

Der Webhook wendet die vorherigen Umgebungsvariablen auf diese Pods an. Ihr Cluster muss nicht den Webhook verwenden, um die Umgebungsvariablen und die Token-Datei-Mounts zu konfigurieren. Sie können Pods manuell so konfigurieren, dass diese Umgebungsvariablen vorhanden sind. Die unterstützten Versionen des AWS SDK suchen zuerst im Credential Chain Provider nach diesen Umgebungsvariablen. Die Anmeldeinformationen der Rolle werden für Pods verwendet, die diese Kriterien erfüllen.

4. Vergewissern Sie sich, dass Ihre Pods mithilfe der Berechtigungen, die Sie in der Ihrer Rolle zugewiesenen IAM-Richtlinie zugewiesen haben, mit den AWS Diensten interagieren können.

> **Note**
>
> Wenn ein Pod AWS Anmeldeinformationen von einer IAM-Rolle verwendet, die mit einem Dienstkonto verknüpft ist, verwendet die AWS CLI oder andere SDKs in den Containern für diesen Pod die Anmeldeinformationen, die von dieser Rolle bereitgestellt werden. Der Pod hat weiterhin Zugriff auf die der Amazon-EKS-Knoten-IAM-Rolle bereitgestellten Anmeldeinformationen, es sei denn, Sie beschränken den Zugriff auf diese Anmeldeinformationen. Weitere Informationen finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.

Wenn Ihre Pods nicht wie erwartet mit den Services interagieren können, führen Sie die folgenden Schritte aus, um sicherzustellen, dass alles richtig konfiguriert ist.

a. Vergewissern Sie sich, dass Ihre Pods eine AWS SDK-Version verwenden, die die Übernahme einer IAM-Rolle über eine OpenID Connect-Webidentitätstokendatei unterstützt. Weitere Informationen finden Sie unter the section called “Unterstützte SDKs”.

b. Stellen Sie sicher, dass die Bereitstellung das Servicekonto verwendet.

```
kubectl describe deployment my-app | grep "Service Account"
```

Eine Beispielausgabe sieht wie folgt aus.

```
Service Account:  my-service-account
```

c. Wenn Ihre Pods immer noch nicht auf Services zugreifen können, überprüfen Sie die unter IAM-Rollen Kubernetes-Servicekonten zuweisen beschriebenen Schritte, um zu bestätigen, dass Ihre Rolle und Ihr Servicekonto richtig konfiguriert sind.

Konfiguration des Endpunkts des AWS-Sicherheitstoken-Service für ein Servicekonto

Wenn Sie ein Kubernetes-Servicekonto mit IAM-Rollen für Servicekonten verwenden, können Sie den Typ des AWS-Sicherheitstoken-Service-Endpunkts konfigurieren, der vom Servicekonto verwendet wird.

AWS empfiehlt die Verwendung der regionalen AWS-STS-Endpunkte statt des globalen Endpunkts. Dies reduziert die Latenz, bietet integrierte Redundanz und erhöht die Gültigkeit der Sitzungstoken. Der AWS-Sicherheitstoken-Service muss in der AWS-Region, in welcher der Pod ausgeführt wird, aktiv sein. Ihre Anwendung muss außerdem über integrierte Redundanz verfügen, um bei Ausfall des Service in der AWS-Region eine andere AWS-Region auszuwählen. Weitere Informationen finden Sie unter Verwaltung von AWS STS in einer AWS-Region im IAM-Benutzerhandbuch.

- Einen vorhandenen -Cluster. Wenn Sie keine Rolle haben, können Sie sie mithilfe einer der Anleitungen in Erste Schritte erstellen.
- Ein vorhandener IAM-OIDC-Anbieter für Ihren Cluster. Weitere Informationen finden Sie unter the section called “IAM-OIDC-Anbieter”.
- Ein vorhandenes Kubernetes-Servicekonto, das für die Verwendung mit der Funktion Amazon-EKS-IAM für Servicekonten.

Die folgenden Beispiele verwenden alle das vom Amazon-VPC-CNI-Plugin verwendete Kubernetes-Servicekonto aws-node. Sie können die *Beispielwerte* durch eigene Servicekonten, Pods, Namespaces und anderen Ressourcen ersetzen.

1. Wählen Sie einen Pod aus, der ein Servicekonto verwendet, für das Sie den Endpunkt ändern möchten. Bestimmen Sie, in welcher AWS-Region der Pod ausgeführt wird. Ersetzen Sie *aws-node-6mfgv* durch den Namen Ihres Pods und *kube-system* mit dem Namespace Ihres Pods.

   ```
   kubectl describe pod aws-node-6mfgv -n kube-system |grep Node:
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   ip-192-168-79-166.us-west-2/192.168.79.166
   ```

   In der vorherigen Ausgabe wird der Pod in einem Knoten in der AWS-Region us-west-2 ausgeführt.
2. Ermitteln Sie den Endpunkt-Typ, den das Servicekonto des Pods verwendet.

   ```
   kubectl describe pod aws-node-6mfgv -n kube-system |grep AWS_STS_REGIONAL_ENDPOINTS
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   AWS_STS_REGIONAL_ENDPOINTS: regional
   ```

   Wenn der aktuelle Endpunkt global ist, dann wird `global` in der Ausgabe zurückgegeben. Wenn keine Ausgabe zurückgegeben wird, wird der Standard-Endpunkttyp verwendet und wurde nicht überschrieben.
3. Wenn Ihre Cluster- oder Plattformversion dieselbe oder höher ist als die in der Tabelle aufgeführten, können Sie den von Ihrem Servicekonto verwendeten Endpunkttyp mit einem der folgenden Befehle vom Standardtyp in einen anderen Typ ändern. Ersetzen Sie *aws-node* durch den Namen Ihres Servicekontos und *kube-system* durch den Namespace Ihres Servicekontos.
   - Wenn Ihr Standard- oder aktueller Endpunkttyp global ist und Sie ihn in regional ändern möchten:

     ```
     kubectl annotate serviceaccount -n kube-system aws-node eks.amazonaws.com/sts-regional-endpoints=true
     ```

Wenn Sie IAM-Rollen für Servicekonten verwenden, um vorsignierte S3-URLs in Ihrer Anwendung zu generieren, die in Pods-Containern ausgeführt wird, ähnelt das Format der URL für regionale Endpunkte dem folgenden Beispiel:

```
https://bucket.s3.us-west-2.amazonaws.com/path?...&X-Amz-Credential=your-access-key-id/date/us-west-2/s3/aws4_request&...
```

- Wenn Ihr Standard- oder aktueller Endpunkttyp regional ist und Sie ihn in global ändern möchten:

```
kubectl annotate serviceaccount -n kube-system aws-node eks.amazonaws.com/sts-regional-endpoints=false
```

Wenn Ihre Anwendung explizit Anfragen an globale AWS-STS-Endpunkte stellt und Sie das Standardverhalten der Verwendung regionaler Endpunkte in Amazon-EKS-Clustern nicht außer Kraft setzen, werden Anfragen mit einem Fehler fehlschlagen. Weitere Informationen finden Sie unter the section called “Pod-Container erhalten folgenden Fehler: `An error occurred (SignatureDoesNotMatch) when calling the GetCallerIdentity operation: Credential should be scoped to a valid region`”.

Wenn Sie IAM-Rollen für Servicekonten verwenden, um vorsignierte S3-URLs in Ihrer Anwendung zu generieren, die in Pods-Containern ausgeführt wird, entspricht das Format der URL für globale Endpunkte in etwa dem folgenden Beispiel:

```
https://bucket.s3.amazonaws.com/path?...&X-Amz-Credential=your-access-key-id/date/us-west-2/s3/aws4_request&...
```

Wenn Sie über eine Automatisierung verfügen, welche die vorsignierte URL in einem bestimmten Format erwartet, oder wenn Ihre Anwendung oder Downstream-Abhängigkeiten, die vorsignierte URLs verwenden, eine bestimmte AWS-Region als Ziel erwarten, nehmen Sie die notwendigen Änderungen vor, um den entsprechenden AWS-sTS-Endpunkt zu verwenden.

4. Löschen Sie alle vorhandenen Pods, die dem Servicekonto zugeordnet sind, und erstellen Sie sie neu, um die Umgebungsvariablen für Anmeldeinformationen anzuwenden. Der mutierende Webhook wendet sie nicht auf Pods an, die bereits ausgeführt werden. Sie können *`Pods`*, *`kube-system`*, und*`-l k8s-app=aws-Knoten`* mit den Informationen für die Pods, für die Sie Ihre Annotation festlegen, ersetzen.

```
kubectl delete Pods -n kube-system -l k8s-app=aws-node
```

5. Bestätigen Sie, dass alle Pods neu gestartet wurden.

```
kubectl get Pods -n kube-system -l k8s-app=aws-node
```

6. Zeigen Sie die Umgebungsvariablen für einen der Pods an. Stellen Sie sicher, dass der `AWS_STS_REGIONAL_ENDPOINTS`-Wert der ist, den Sie im vorherigen Schritt festgelegt haben.

```
kubectl describe pod aws-node-kzbtr -n kube-system |grep AWS_STS_REGIONAL_ENDPOINTS
```

   Eine Beispielausgabe sieht wie folgt aus.

```
AWS_STS_REGIONAL_ENDPOINTS=regional
```

Authentifizierung bei einem anderen Konto mit IRSA

Sie können kontoübergreifende IAM-Berechtigungen konfigurieren, indem Sie entweder einen Identitätsanbieter aus dem Cluster eines anderen Kontos erstellen oder verkettete `AssumeRole`-Operationen verwenden. In den folgenden Beispielen besitzt Konto A einen Amazon-EKS-Cluster, der IAM-Rollen für Servicekonten unterstützt. Pods, die in diesem Cluster ausgeführt werden, müssen IAM-Berechtigungen von Konto B annehmen.

Example Identitätsanbieter aus dem Cluster eines anderen Kontos erstellen

Example

In diesem Beispiel stellt Konto A dem Konto B die OpenID Connect (OIDC)-Aussteller-URL aus seinem Cluster bereit. Konto B befolgt die Anweisungen unter Erstellen eines IAM-OIDC-Anbieters für Ihren Cluster und the section called “IAM-Rolle zuweisen” unter Verwendung der OIDC-Aussteller-URL aus dem Cluster von Konto A. Anschließend fügt ein Clusteradministrator dem Dienstkonto im Cluster von Konto A Anmerkungen hinzu, um die Rolle von Konto B () zu verwenden. *444455556666*

```
apiVersion: v1
kind: ServiceAccount
metadata:
  annotations:
```

```
    eks.amazonaws.com/role-arn: arn:aws: iam::444455556666:role/account-b-role
```

Example Verwenden von verketteten **`AssumeRole`**-Operationen

Example

In diesem Beispiel erstellt Konto B eine IAM-Richtlinie mit den Berechtigungen, die Pods im Cluster von Konto A gewährt werden sollen. Konto B (*444455556666*) ordnet diese Richtlinie einer IAM-Rolle mit einer Vertrauensstellung zu, die `AssumeRole` Berechtigungen für Konto A () gewährt. *111122223333*

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "sts:AssumeRole",
      "Condition": {}
    }
  ]
}
```

Konto A erstellt eine Rolle mit einer Vertrauensrichtlinie, die Anmeldeinformationen von dem Identitätsanbieter erhält, der mit der OIDC-Aussteller-Adresse des Clusters erstellt wurde.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/oidc.eks.us-
east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "oidc.eks.us-east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud":
 "sts.amazonaws.com"
```

```
        }
      }
    }
  ]
}
```

Konto A fügt dieser Rolle eine Richtlinie mit den folgenden Berechtigungen an, um die Rolle anzunehmen, die Konto B erstellt hat.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::444455556666:role/account-b-role"
        }
    ]
}
```

Der Anwendungscode für Pods, welche die Rolle von Konto B übernehmen sollen, verwendet zwei Profile: `account_b_role` und `account_a_role`. Das `account_b_role`-Profil verwendet das `account_a_role`-Profil als Quelle. Für die AWS CLI ähnelt die `~/.aws/config` Datei der folgenden.

```
[profile account_b_role]
source_profile = account_a_role
role_arn=arn:aws: iam::444455556666:role/account-b-role

[profile account_a_role]
web_identity_token_file = /var/run/secrets/eks.amazonaws.com/serviceaccount/token
role_arn=arn:aws: iam::111122223333:role/account-a-role
```

Informationen zur Angabe verketteter Profile für andere AWS SDKs finden Sie in der Dokumentation für das SDK, das Sie verwenden. Weitere Informationen finden Sie unter Tools, auf AWS denen Sie aufbauen können.

IRSA mit dem AWS-SDK verwenden

Verwenden von Anmeldeinformationen

Um die Anmeldeinformationen von IAM-Rollen für Servicekonten (IRSA) zu verwenden, kann Ihr Code ein beliebiges AWS-SDK verwenden, um einen Client für einen AWS-Service mit einem SDK zu erstellen. Standardmäßig sucht das SDK in einer Kette von Speicherorten nach zu verwendenden Anmeldeinformationen für AWS. Die IAM-Rollen für Anmeldeinformationen für Servicekonten werden verwendet, wenn Sie bei der Erstellung des Clients keinen Anmeldeinformationsanbieter angeben oder Sie das SDK anderweitig initialisiert haben.

Das funktioniert, weil IAM-Rollen für Servicekonten als Schritt in der standardmäßigen Anmeldeinformationskette hinzugefügt wurden. Wenn Ihre Workloads derzeit Anmeldeinformationen verwenden, die sich an früherer Stelle in der Anmeldeinformationskette befinden, werden diese Anmeldeinformationen auch dann weiterhin verwendet, wenn Sie IAM-Rollen für Servicekonten für dieselbe Workload konfigurieren.

Mithilfe der `AssumeRoleWithWebIdentity`-Aktion tauscht das SDK das OIDC-Token des Servicekontos automatisch gegen temporäre Anmeldeinformationen vom AWS-Sicherheits-Token-Service aus. Amazon EKS und diese SDK-Aktion rotieren weiterhin die temporären Anmeldeinformationen, indem sie erneuert werden, bevor sie ablaufen.

Wenn Sie IAM-Rollen für Servicekonten benutzen, müssen die Container in Ihren eine AWS-SDK-Version verwenden, welche die Annahme einer IAM-Rolle über eine Token-Datei für OpenID-Connect-Webidentitäts unterstützt. Stellen Sie sicher, dass Sie die folgenden Versionen oder höher für Ihr AWS-SDK verwenden:

- Java (Version 2) – 2.10.11
- Java – 1.12.782
- AWS SDK für Go v1 – 1.23.13
- AWS SDK für Go v2 – Support für alle Versionen
- Python (Boto3) — 1.9.220
- Python (botocore) —1.12.200
- AWS CLI – 1.16.232
- Knoten – 2.525.0 und 3.27.0
- Ruby – 3.58.0
- C++ – 1.7.174
- .NET – 3.3.659.1 – Sie müssen auch `AWSSDK.SecurityToken` angeben.
- PHP – 3.110.7

Viele gängige Kubernetes- Add-Ons, wie beispielsweise der Cluster Autoscaler, das Routing des Internet-Datenverkehrs mit AWS Load Balancer Controller und das Amazon-VPC-CNI-Plugin für Kubernetes, unterstützen IAM-Rollen für Servicekonten.

Um sicherzustellen, dass Sie ein unterstütztes SDK verwenden, befolgen Sie die Installationsanweisungen für Ihr bevorzugtes SDK unter Tools für die Entwickelung in AWS, wenn Sie Ihre Container entwickeln.

Überlegungen

Java

Wenn Sie Java verwenden, müssen Sie das `sts`-Modul in den Klassenpfad aufnehmen. Weitere Informationen finden Sie unter WebIdentityTokenFileCredentialsProvider in den Java SDK-Dokumenten.

Signaturschlüssel zur Validierung von OIDC-Token abrufen

Kubernetes vergibt einen `ProjectedServiceAccountToken` an jedes Kubernetes-Servicekonto. Dieser Token ist ein OIDC-Token, bei dem es wiederum um eine Art JSON-Web-Token (JWT) handelt. Amazon EKS Hostet für jeden Cluster einen öffentlichen OIDC-Endpunkt, der die Signaturschlüssel für das Token enthält, sodass externe Systeme es validieren können.

Um einen `ProjectedServiceAccountToken` zu validieren, müssen Sie die öffentlichen OIDC-Signaturschlüssel abrufen, die auch als JSON Web Key Set (JWKS) bezeichnet werden. Verwenden Sie diese Schlüssel in Ihrer Anwendung, um das Token zu validieren. Sie können beispielsweise die Python-Bibliothek PyJWT verwenden, um Token mit diesen Schlüsseln zu validieren. Weitere Informationen zu `ProjectedServiceAccountToken` finden Sie unter the section called “Hintergrundinformationen zu IAM, Kubernetes und OpenID Connect (OIDC)”.

Voraussetzungen

- Ein vorhandener Anbieter für AWS Identity and Access Management (IAM) OpenID Connect (OIDC) für Ihren Cluster. Informationen zum Feststellen, ob Sie bereits über einen verfügen oder einen erstellen müssen, finden Sie unter the section called “IAM-OIDC-Anbieter”.
- AWS-CLI – Ein Befehlszeilen-Tool für die Arbeit mit AWS-Services, einschließlich Amazon EKS. Weitere Informationen finden Sie unter Installation im Benutzerhandbuch der AWS-Befehlszeilenschnittstelle. Nach der Installation der AWS-CLI empfehlen wir, sie auch zu konfigurieren. Weitere Informationen finden Sie unter Schnellkonfiguration mit aws configure im Benutzerhandbuch zur AWS-Befehlszeilenschnittstelle.

Verfahren

1. Rufen Sie die OIDC-URL für Ihren Amazon-EKS-Cluster mithilfe der AWS-CLI ab.

```
$ aws eks describe-cluster --name my-cluster --query 'cluster.identity.oidc.issuer'
"https://oidc.eks.us-west-2.amazonaws.com/id/8EBDXXXX00BAE"
```

2. Rufen Sie den öffentlichen Signaturschlüssel mit curl oder einem ähnlichen Tool ab. Das Ergebnis ist ein JSON Web Key Set (JWKS).

> **Important**
>
> Amazon EKS drosselt das Aufrufen des OIDC-Endpunkts. Es wird empfohlen, den öffentlichen Signaturschlüssel zwischenzuspeichern. Beachten Sie die in der Antwort enthaltene `cache-control`-Kopfzeile.

> **Important**
>
> Amazon EKS rotiert den OIDC-Signaturschlüssel alle sieben Tage.

```
$ curl https://oidc.eks.us-west-2.amazonaws.com/id/8EBDXXXX00BAE/keys
{"keys":
[{"kty":"RSA","kid":"2284XXXX4a40","use":"sig","alg":"RS256","n":"wklbXXXXMVfQ","e":"AQAB"}]
```

Erfahren Sie, wie EKS Pod Identity Pods Zugriff auf AWS-Services gewährt

Die Anwendungen in den Containern eines Pods können ein AWS-SDK oder die AWS-CLI verwenden, um API-Anforderungen an AWS-Services mithilfe von AWS Identity and Access Management (IAM)-Berechtigungen zu senden. Die Anwendungen müssen ihre AWS-API-Anforderungen mit AWS-Anmeldeinformationen signieren.

EKS-Pod-Identitäten bieten die Möglichkeit, Anmeldeinformationen für Ihre Anwendungen zu verwalten, ähnlich wie Amazon-EC2-Instance-Profile Anmeldeinformationen für Amazon-EC2-Instances bereitstellen. Anstatt Ihre AWS-Anmeldeinformationen zu erstellen und an die Container zu verteilen oder die Rolle der Amazon-EC2-Instance zu verwenden, verknüpfen Sie eine IAM-Rolle

mit einem Kubernetes-Servicekonto und konfigurieren Ihre Pods so, dass sie das Servicekonto verwenden.

Jede EKS-Pod-Identity-Zuordnung ordnet einem Servicekonto in einem Namespace im angegebenen Cluster eine Rolle zu. Wenn Sie dieselbe Anwendung in mehreren Clustern haben, können Sie in jedem Cluster identische Zuordnungen vornehmen, ohne die Vertrauensrichtlinie der Rolle zu ändern.

Wenn ein Pod ein Servicekonto mit einer Zuordnung verwendet, legt Amazon EKS Umgebungsvariablen in den Containern des Pods fest. Die Umgebungsvariablen konfigurieren die AWS-SDKs, einschließlich der AWS-CLI, für die Verwendung der Anmeldeinformationen für EKS Pod Identity.

Vorteile von EKS-Pod-Identitäten

EKS-Pod-Identitäten bieten die folgenden Vorteile:

- Geringste Berechtigung – Sie können IAM-Berechtigungen auf ein Servicekonto beschränken, und nur Pods, die dieses Servicekonto verwenden, haben Zugriff auf diese Berechtigungen. Mit diesem Feature entfällt auch die Notwendigkeit von Drittanbieterlösungen wie `kiam` oder `kube2iam`.
- Isolierung von Anmeldeinformationen – Wenn der Zugriff auf den Amazon EC2 Instance Metadata Service (IMDS) eingeschränkt ist, können die Container eines Pods nur Anmeldeinformationen für die IAM-Rolle abrufen, die dem vom Container verwendeten Servicekonto zugeordnet ist. Ein Container hat nie Zugriff auf Anmeldeinformationen, die von anderen Containern in anderen Pods verwendet werden. Wenn IMDS nicht eingeschränkt ist, haben die Container des Pods auch Zugriff auf die IAM-Rolle des Amazon-EKS-Knotens und die Container können möglicherweise auf Anmeldeinformationen von IAM-Rollen anderer Pods auf demselben Knoten zugreifen. Weitere Informationen finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.

> **Note**
>
> Mit `hostNetwork: true` konfigurierte Pods haben immer IMDS-Zugriff, aber die AWS-SDKs und die CLI verwenden bei Aktivierung die Pod-Identity-Anmeldeinformationen.

- Überprüfbarkeit – Die Zugriffs- und Ereignisprotokollierung ist über AWS CloudTrail verfügbar, um eine nachträgliche Überprüfung zu vereinfachen.

> **Important**
>
> Container stellen keine Sicherheitsgrenze dar und die Verwendung von Pod Identity ändert daran nichts. Pods, die demselben Knoten zugewiesen sind, teilen sich einen Kernel und möglicherweise andere Ressourcen, abhängig von Ihrer Pod-Konfiguration. Während Pods, die auf separaten Knoten ausgeführt werden, auf der Rechenebene isoliert sind, gibt es Knotenanwendungen, die über zusätzliche Berechtigungen in der Kubernetes-API verfügen, die über den Umfang einer einzelnen Instance hinausgehen. Einige Beispiele sind `kubelet`, `kube-proxy`, CSI-Speichertreiber oder Ihre eigenen Kubernetes-Anwendungen.

EKS Pod Identity ist eine einfachere Methode als the section called “Anmeldeinformationen mit IRSA”, da diese keine OIDC-Identitätsanbieter nutzt. EKS Pod Identity bietet die folgenden Verbesserungen:

- Unabhängiger Betrieb – In vielen Organisationen liegt die Verantwortung für die Erstellung von OIDC-Identitätsanbietern bei anderen Teams als für die Verwaltung der Kubernetes-Cluster. EKS Pod Identity hat eine klare Aufgabentrennung: Die gesamte Konfiguration der EKS-Pod-Identity-Zuordnungen erfolgt in Amazon EKS und die gesamte Konfiguration der IAM-Berechtigungen erfolgt in IAM.
- Wiederverwendbarkeit – EKS Pod Identity verwendet einen einzigen IAM-Prinzipal anstelle der separaten Prinzipale für jeden Cluster, den IAM-Rollen für Servicekonten verwenden. Ihr IAM-Administrator fügt der Vertrauensrichtlinie jeder Rolle den folgenden Prinzipal hinzu, damit sie von EKS-Pod-Identitäten verwendet werden kann.

  ```
  "Principal": {
      "Service": "pods.eks.amazonaws.com"
  }
  ```

- Skalierbarkeit – Jeder Satz temporärer Anmeldeinformationen wird vom EKS-Service in EKS Pod Identity angenommen statt von jedem AWS-SDK, das Sie in jedem Pod ausführen. Anschließend stellt der Amazon EKS Pod Identity Agent, der auf jedem Knoten ausgeführt wird, die Anmeldeinformationen an die SDKs aus. Dadurch wird die Last für jeden Knoten reduziert und nicht in jedem Pod dupliziert. Weitere Details zu diesem Prozess finden Sie unter the section called “Funktionsweise”.

Weitere Informationen zum Vergleich der beiden Alternativen finden Sie unter the section called “Workload-Zugriff auf AWS ”.

Übersicht über die Einrichtung von EKS-Pod-Identitäten

Aktivieren Sie EKS-Pod-Identitäten, indem Sie die folgenden Verfahren ausführen:

1. the section called “Agent einrichten” – Sie führen diesen Vorgang für jeden Cluster nur einmal durch. Sie müssen diesen Schritt nicht ausführen, wenn EKS Auto Mode in Ihrem Cluster aktiviert ist.
2. the section called “IAM-Rolle zuweisen” – Führen Sie dieses Verfahren für jeden einzelnen Berechtigungssatz durch, den eine Anwendung haben soll.
3. the section called “Pod-Servicekonten” – Führen Sie dieses Verfahren für jeden Pod durch, der Zugriff auf AWS-Services benötigt.
4. the section called “Unterstützte SDKs” – Stellen Sie sicher, dass der Workload ein AWS-SDK einer unterstützten Version verwendet und dass der Workload die Kette standardmäßiger Anmeldeinformationen verwendet.

Grenzwerte

- Pro Cluster werden bis zu 5 000 EKS-Pod-Identity-Zuordnungen unterstützt, um IAM-Rollen Kubernetes-Servicekonten zuzuordnen.

Überlegungen

- IAM-Rollenzuordnung: Jedes Kubernetes-Servicekonto in einem Cluster kann einer IAM-Rolle aus demselben AWS-Konto wie das Cluster zugeordnet werden. Sie ändern die Rolle, indem Sie die EKS-Pod-Identity-Zuordnung bearbeiten. Für den kontenübergreifenden Zugriff delegieren Sie den Zugriff auf die Rolle mithilfe von IAM-Rollen. Weitere Informationen finden Sie unter Delegieren des Zugriffs über alle AWS-Konten hinweg mithilfe von IAM-Rollen im IAM-Benutzerhandbuch.
- EKS-Pod-Identity-Agent: Der Pod-Identity-Agent ist zur Verwendung von EKS Pod Identity erforderlich. Der Agent wird als Kubernetes `DaemonSet` in Cluster-Knoten ausgeführt und stellt Anmeldeinformationen nur für Pods auf demselben Knoten bereit. Er verwendet das `hostNetwork` des Knotens und belegt die Ports `80` und `2703` auf der link-lokalen Adresse (`169.254.170.23` für IPv4, `[fd00:ec2::23]` für IPv6). Wenn IPv6 in Ihrem Cluster deaktiviert ist, deaktivieren Sie IPv6

für den Pod-Identity-Agent. Weitere Informationen finden Sie unter IPv6 im EKS-Pod-Identity-Agent deaktivieren.

- Eventuelle Konsistenz: EKS-Pod-Identity-Zuordnungen sind letztendlich konsistent, mit möglichen Verzögerungen von mehreren Sekunden nach API-Aufrufen. Vermeiden Sie die Erstellung oder Aktualisierung von Verknüpfungen in kritischen Code-Pfaden mit hoher Verfügbarkeit. Führen Sie diese Aktionen stattdessen in separaten, weniger häufigen Initialisierungs- oder Einrichtungsroutinen durch. Weitere Informationen finden Sie unter Sicherheitsgruppen pro Pod im EKS-Leitfaden für bewährte Methoden.
- Überlegungen zu Proxy und Sicherheitsgruppe: Bei Pods, die einen Proxy verwenden, fügen Sie `169.254.170.23` (IPv4) und `[fd00:ec2::23]` (IPv6) zu den `no_proxy`/`NO_PROXY`-Umgebungsvariablen hinzu, um fehlgeschlagene Anfragen an den EKS-Pod-Identity-Agent zu vermeiden. Wenn Sie Sicherheitsgruppen für Pods mit AWS VPC CNI verwenden, setzen Sie das `ENABLE_POD_ENI`-Flag auf „true“ und das `POD_SECURITY_GROUP_ENFORCING_MODE`-Flag auf „standard“. Weitere Informationen finden Sie unter Zuweisen von Sicherheitsgruppen zu einzelnen Pods.

Cluster-Versionen von EKS Pod Identity

Um EKS Pod Identity nutzen zu können, muss der Cluster über eine Plattformversion verfügen, die mindestens der in der folgenden Tabelle aufgeführten Version entspricht, oder über eine Kubernetes-Version, die neuer ist als die in der Tabelle aufgeführten Versionen. Die empfohlene Version des Amazon-EKS-Pod-Identity-Agent für eine Kubernetes-Version finden Sie unter the section called “Kompatibilität überprüfen”.

Kubernetes-Version	Plattformversion
Nicht aufgeführte Kubernetes-Versionen	Alle Plattformversionen werden unterstützt
`1.28`	`eks.4`

Einschränkungen von EKS Pod Identity

EKS-Pod-Identitäten sind auf folgenden Plattformen verfügbar:

- Amazon-EKS-Cluster-Versionen, die im vorherigen Thema (the section called “Cluster-Versionen von EKS Pod Identity”) aufgeführt wurden.
- Worker-Knoten im Cluster, die Linux-Amazon-EC2-Instances sind.

EKS Pod Identities sind auf folgenden Plattformen nicht verfügbar:

- AWS Outposts
- Amazon EKS Anywhere.
- Kubernetes-Cluster, die Sie in Amazon EC2 erstellen und ausführen. Die EKS-Pod-Identity-Komponenten sind nur in Amazon EKS verfügbar.

Sie können EKS Pod Identities nicht verwenden mit:

- Pods, die nicht auf Linux-Amazon-EC2-Instances ausgeführt werden. Linux- und Windows-Pods, die in AWS Fargate (Fargate) ausgeführt werden, werden nicht unterstützt. In Windows-Amazon-EC2-Instances ausgeführte Pods werden nicht unterstützt.

Funktionsweise von EKS Pod Identity verstehen

Amazon-EKS-Pod-Identity-Zuordnungen bieten die Möglichkeit, Anmeldeinformationen für Ihre Anwendungen zu verwalten, ähnlich wie Amazon-EC2-Instance-Profile Anmeldeinformationen für Amazon-EC2-Instances bereitstellen.

Amazon EKS Pod Identity bietet Anmeldeinformationen für Ihre Workloads mit einer zusätzlichen EKS-Auth-API und einem Agent-Pod, der auf jedem Knoten ausgeführt wird.

In Ihren Add-ons, z. B. Amazon-EKS-Add-ons und Add-ons für selbstverwaltete Controller und Operatoren, muss der Autor seine Software aktualisieren, um die neuesten AWS-SDKs verwenden zu können. Die Liste der Kompatibilität zwischen EKS Pod Identity und den von Amazon EKS produzierten Add-ons finden Sie im vorherigen Abschnitt (the section called “Einschränkungen von EKS Pod Identity”).

Verwenden von EKS-Pod-Identitäten in Ihrem Code

In Ihrem Code können Sie die AWS-SDKs verwenden, um auf AWS-Services zuzugreifen. Sie schreiben Code, um einen Client für einen AWS-Service mit einem SDK zu erstellen. Standardmäßig sucht das SDK in einer Kette von Speicherorten nach zu verwendenden Anmeldeinformationen für AWS-Identitäts- und Zugriffsverwaltung. Nachdem gültige Anmeldeinformationen gefunden wurden, wird die Suche beendet. Weitere Informationen zu den verwendeten Standardspeicherorten finden Sie unter Anbieterkette von Anmeldeinformationen im Referenzhandbuch für AWS-SDKs und -Tools.

EKS-Pod-Identitäten wurden dem Anbieter für Container-Anmeldeinformationen hinzugefügt, der in einem Schritt in der standardmäßigen Anmeldeinformationskette durchsucht wird. Wenn Ihre

Workloads derzeit Anmeldeinformationen verwenden, die in der Kette der Anmeldeinformationen weiter vorn stehen, werden diese Anmeldeinformationen auch dann weiter verwendet, wenn Sie eine EKS-Pod-Identity-Zuordnung für dieselbe Workload konfigurieren. Auf diese Weise können Sie sicher von anderen Anmeldeinformationstypen migrieren, indem Sie die Zuordnung erstellen, bevor Sie die alten Anmeldeinformationen entfernen.

Der Anbieter für Container-Anmeldeinformationen stellt temporäre Anmeldeinformationen über einen Agent bereit, der auf jedem Knoten ausgeführt wird. In Amazon EKS ist dies der Amazon EKS Pod Identity Agent und bei Amazon Elastic Container Service ist es der `amazon-ecs-agent`. Die SDKs verwenden Umgebungsvariablen, um den Agent zu finden, zu dem eine Verbindung hergestellt werden soll.

Im Gegensatz dazu stellen IAM-Rollen für Servicekonten ein Token für Web-Identität bereit, welches das AWS-SDK mithilfe von `AssumeRoleWithWebIdentity` mit dem AWS-Sicherheitstoken-Service austauschen muss.

Funktionsweise von EKS-Pod-Identity-Agent mit einem Pod

1. Wenn Amazon EKS einen neuen Pod startet, der ein Servicekonto mit einer EKS-Pod-Identity-Zuordnung verwendet, fügt der Cluster dem Pod-Manifest den folgenden Inhalt hinzu:

```
    env:
    - name: AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE
      value: "/var/run/secrets/pods.eks.amazonaws.com/serviceaccount/eks-pod-
identity-token"
    - name: AWS_CONTAINER_CREDENTIALS_FULL_URI
      value: "http://169.254.170.23/v1/credentials"
    volumeMounts:
    - mountPath: "/var/run/secrets/pods.eks.amazonaws.com/serviceaccount/"
      name: eks-pod-identity-token
  volumes:
  - name: eks-pod-identity-token
    projected:
      defaultMode: 420
      sources:
      - serviceAccountToken:
          audience: pods.eks.amazonaws.com
          expirationSeconds: 86400 # 24 hours
          path: eks-pod-identity-token
```

2. Kubernetes wählt aus, auf welchem Knoten der Pod ausgeführt werden soll. Anschließend verwendet der Amazon EKS Pod Identity Agent im Knoten die Aktion AssumeRoleForPodIdentity, um von der EKS-Auth-API temporäre Anmeldeinformationen abzurufen.
3. Der EKS Pod Identity Agent stellt diese Anmeldeinformationen für die AWS-SDKs zur Verfügung, die Sie in Ihren Containern ausführen.
4. Sie nutzen das SDK in Ihrer Anwendung, ohne einen Anmeldeinformationsanbieter für die Verwendung der standardmäßigen Anmeldeinformationskette anzugeben. Oder Sie geben den Anbieter für Container-Anmeldeinformationen an. Weitere Informationen zu den verwendeten Standardspeicherorten finden Sie unter Anbieterkette von Anmeldeinformationen im Referenzhandbuch für AWS-SDKs und -Tools.
5. Das SDK verwendet die Umgebungsvariablen, um eine Verbindung zum EKS Pod Identity Agent herzustellen und die Anmeldeinformationen abzurufen.

> **Note**
>
> Wenn Ihre Workloads derzeit Anmeldeinformationen verwenden, die in der Kette der Anmeldeinformationen weiter vorn stehen, werden diese Anmeldeinformationen auch dann weiter verwendet, wenn Sie eine EKS-Pod-Identity-Zuordnung für dieselbe Workload konfigurieren.

Einrichtung des Amazon-EKS-Pod-Identity-Agenten

Amazon EKS Pod Identity-Zuordnungen bieten die Möglichkeit, Anmeldeinformationen für Ihre Anwendungen zu verwalten, ähnlich wie EC2 Amazon-Instance-Profile Anmeldeinformationen für EC2 Amazon-Instances bereitstellen.

Amazon EKS Pod Identity bietet Anmeldeinformationen für Ihre Workloads mit einer zusätzlichen EKS-Auth-API und einem Agent-Pod, der auf jedem Knoten ausgeführt wird.

> **Tip**
>
> Es ist nicht erforderlich, den EKS-Pod-Identity-Agent auf EKS-Auto-Mode-Clustern zu installieren. Diese Funktion ist in EKS Auto Mode integriert.

Überlegungen

- Standardmäßig ist der EKS-Pod-Identity-Agent auf EKS-Auto-Mode-Clustern vorinstalliert. Weitere Informationen hierzu finden Sie unter EKS Auto Mode.
- Standardmäßig überwacht der EKS-Pod-Identity-Agent eine `IPv4`- und `IPv6`-Adresse auf Pods, die Anmeldeinformationen anfordern. Der Agent verwendet die Loopback-IP-Adresse (localhost) `169.254.170.23` für `IPv4` und die localhost-IP-Adresse `[fd00:ec2::23]` für `IPv6`.
- Wenn Sie `IPv6`-Adressen deaktivieren oder localhost-IP-Adressen `IPv6` anderweitig verhindern, kann der Agent nicht gestartet werden. Um den Agenten auf Knoten zu starten, die `IPv6` nicht verwenden können, befolgen Sie die Schritte unter the section called “IPv6 deaktivieren”, um die `IPv6`-Konfiguration zu deaktivieren.

Erstellen des Amazon EKS Pod Identity Agent

Voraussetzungen für den Agent

- Ein vorhandener Amazon-EKS-Cluster. Informationen zum Bereitstellen finden Sie unter Erste Schritte. Die Cluster-Version und die Plattformversion müssen mit den in den EKS-Pod-Identity-Cluster-Versionen aufgeführten Versionen übereinstimmen oder höher sein.
- Die Knotenrolle verfügt über Berechtigungen für den Agent, um die `AssumeRoleForPodIdentity`-Aktion in der EKS-Auth-API auszuführen. Sie können die AWS verwaltete Richtlinie verwenden: Amazon EKSWorker NodePolicy oder eine benutzerdefinierte Richtlinie hinzufügen, die der folgenden ähnelt:

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks-auth:AssumeRoleForPodIdentity"
            ],
            "Resource": "*"
        }
    ]
}
```

Diese Aktion kann durch Tags beschränkt werden, um einzuschränken, welche Rollen von Pods übernommen werden können, die den Agent verwenden.

- Die Knoten können auf Amazon ECR zugreifen und Images von Amazon ECR herunterladen. Das Container-Image für das Add-On befindet sich in den Registies, die unter Amazon-Container-Image-Registries für Amazon-EKS-Add-Ons anzeigen aufgeführt sind.

  Beachten Sie, dass Sie den Speicherort des Images ändern und EKS-Add-Ons in den optionalen Konfigurationseinstellungen in der AWS-Managementkonsole und `--configuration-values` in der AWS CLI bereitstellen `imagePullSecrets` können.
- Die Knoten können die Amazon-EKS-Auth-API erreichen. Für private Cluster ist der `eks-auth` AWS PrivateLink Endpunkt-Eingang erforderlich.

Agent mit AWS Konsole einrichten

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich Cluster aus. Wählen Sie anschließend den Namen des Clusters aus, für den Sie das Add-on „EKS Pod Identity Agent“ konfigurieren möchten.
3. Wählen Sie die Registerkarte Add-ons.
4. Wählen Sie Weitere Add-Ons erhalten.
5. Wählen Sie das Kästchen oben rechts im Add-on-Bereich für EKS Pod Identity Agent aus und gehen Sie dann auf Bearbeiten.
6. Wählen Sie auf der Seite Konfigurieren ausgewählter Add-ons-Einstellungen eine beliebige Version in der Dropdown-Liste Version aus.
7. (Optional) Erweitern Sie Optionale Konfigurationseinstellungen, um eine zusätzliche Konfiguration einzugeben. Sie können beispielsweise einen alternativen Speicherort für das Container-Image und `ImagePullSecrets` angeben. Das JSON-Schema mit den akzeptierten Schlüsseln wird im Add-On-Konfigurationsschema angezeigt.

   Geben Sie die Konfigurationsschlüssel und Werte in das Feld Konfigurationswerte ein
8. Wählen Sie Weiter aus.
9. Vergewissern Sie sich, dass die EKS-Pod-Identity-Pods auf Ihrem Cluster ausgeführt werden.

```
kubectl get pods -n kube-system | grep 'eks-pod-identity-agent'
```

Eine Beispielausgabe sieht wie folgt aus.

```
eks-pod-identity-agent-gmqp7                                          1/1     Running
   1 (24h ago)   24h
eks-pod-identity-agent-prnsh                                          1/1     Running
   1 (24h ago)   24h
```

Sie können jetzt EKS-Pod-Identity-Zuordnungen in Ihrem Cluster verwenden. Weitere Informationen finden Sie unter the section called “IAM-Rolle zuweisen”.

Agent mit AWS CLI einrichten

1. Führen Sie den folgenden AWS CLI-Befehl aus. Ersetzen Sie `my-cluster` mit dem Namen Ihres Clusters.

```
aws eks create-addon --cluster-name my-cluster --addon-name eks-pod-identity-agent --
addon-version v1.0.0-eksbuild.1
```

> **Note**
>
> Der EKS-Pod-Identity-Agent verwendet den `service-account-role-arn` nicht für IAM-Rollen für Servicekonten. Sie müssen dem EKS Pod Identity Agent Berechtigungen für die Knotenrolle gewähren.

2. Vergewissern Sie sich, dass die EKS-Pod-Identity-Pods auf Ihrem Cluster ausgeführt werden.

```
kubectl get pods -n kube-system | grep 'eks-pod-identity-agent'
```

Eine Beispielausgabe sieht wie folgt aus.

```
eks-pod-identity-agent-gmqp7                                          1/1     Running
   1 (24h ago)   24h
eks-pod-identity-agent-prnsh                                          1/1     Running
   1 (24h ago)   24h
```

Sie können jetzt EKS-Pod-Identity-Zuordnungen in Ihrem Cluster verwenden. Weitere Informationen finden Sie unter the section called “IAM-Rolle zuweisen”.

Zuordnung einer IAM-Rolle einem Kubernetes-Servicekonto

In diesem Thema wird beschrieben, wie Sie ein Kubernetes-Dienstkonto so konfigurieren, dass es eine AWS Identity and Access Management (IAM) -Rolle mit EKS Pod Identity annimmt. Alle Pods, die für die Verwendung des Dienstkontos konfiguriert sind, können dann auf jeden AWS Dienst zugreifen, für den die Rolle über Zugriffsberechtigungen verfügt.

Um eine EKS Pod Identity-Zuordnung zu erstellen, gibt es nur einen einzigen Schritt: Sie erstellen die Zuordnung in EKS über die AWS CLI AWS-Managementkonsole AWS SDKs, AWS CloudFormation und andere Tools. Es gibt keine Daten oder Metadaten zu den Zuordnungen innerhalb des Clusters in Kubernetes-Objekten, und Sie fügen den Servicekonten keine Annotationen hinzu

Voraussetzungen

- Einen vorhandenen -Cluster. Falls Sie noch kein Service-Konto haben, können Sie eines erstellen, indem Sie einer der Anleitungen in Erste Schritte folgen.
- Der IAM-Prinzipal, der die Zuordnung erstellt, muss `iam:PassRole` haben.
- Die neueste Version der AWS CLI, die auf Ihrem Gerät installiert und konfiguriert ist, oder AWS CloudShell. Sie können Ihre aktuelle Version mit `aws --version | cut -d / -f2 | cut -d ' ' -f1` überprüfen. Paketmanager wie `yum`apt-get, oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der AWS CLI. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Die in der installierte AWS CLI-Version AWS CloudShell kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zur Aktualisierung finden Sie im AWS CloudShell Benutzerhandbuch unter AWS CLI in Ihrem Home-Verzeichnis installieren.
- Das `kubectl`-Befehlszeilen-Tool ist auf Ihrem Gerät oder in der AWS CloudShell installiert. Die Version kann mit der Kubernetes-Version Ihres Clusters identisch sein oder bis zu einer Nebenversion älter oder neuer sein. Wenn Ihre Clusterversion beispielsweise `1.29` ist, können Sie `kubectl`-Version `1.28`, `1.29`, oder `1.30` damit verwenden. Informationen zum Installieren oder Aktualisieren von `kubectl` finden Sie unter the section called “kubectl und eksctl einrichten”.
- Eine vorhandene `kubectl config`-Datei, die Ihre Clusterkonfiguration enthält. Informationen zum Erstellen einer `kubectl config`-Datei finden Sie unter the section called “Zugriff auf den Cluster mit kubectl”.

Erstellen Sie eine Pod Identity-Zuordnung (AWS Konsole)

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich Cluster aus. Wählen Sie anschließend den Namen des Clusters aus, für den Sie das Add-on „EKS Pod Identity Agent“ konfigurieren möchten.
3. Wählen Sie die Registerkarte Zugriff aus.
4. Wählen Sie in den Pod-Identity-Zuordnungen die Option Erstellen aus.
5. Wählen Sie als IAM-Rolle die IAM-Rolle mit den Berechtigungen aus, die die Workload haben soll.

> **Note**
>
> Die Liste enthält nur Rollen mit der folgenden Vertrauensrichtlinie, die EKS Pod Identity ermöglicht, diese Rollen zu verwenden.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEksAuthToAssumeRoleForPodIdentity",
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
```

`sts:AssumeRole` – EKS Pod Identity nutzt `AssumeRole`, um die IAM-Rolle anzunehmen, bevor es die temporären Anmeldeinformationen an Ihre Pods übergibt.

`sts:TagSession`— EKS Pod Identity verwendet`TagSession`, um Sitzungs-Tags in die Anfragen an AWS STS aufzunehmen.

Sie können diese Tags in den Bedingungsschlüsseln der Vertrauensrichtlinie verwenden, um einzuschränken, welche Servicekonten, Namespaces und Cluster diese Rolle verwenden dürfen.

Eine Liste der Amazon EKS-Bedingungsschlüssel finden Sie unter Bedingungsschlüssel für Amazon Elastic Kubernetes Service in der Service-Autorisierungsreferenz. Um zu erfahren, mit welchen Aktionen und Ressourcen Sie einen Bedingungsschlüssel verwenden können, lesen Sie von Amazon Elastic Kubernetes Service definierte Aktionen.

6. Wählen Sie als Kubernetes-Namespace den Kubernetes-Namespace aus, der das Servicekonto und die Workload enthält. Optional können Sie den Namen eines Namespace angeben, der nicht im Cluster vorhanden ist.
7. Wählen Sie für das Kubernetes-Servicekonto das zu verwendende Kubernetes-Servicekonto aus. Das Manifest für Ihre Kubernetes-Workload muss dieses Servicekonto angeben. Optional können Sie den Namen eines Servicekontos angeben, das nicht im Cluster vorhanden ist.
8. (Optional) Wählen Sie für die Tags die Option Tag hinzufügen aus, um Metadaten in einem Schlüssel-Wert-Paar hinzuzufügen. Diese Tags werden auf die Zuordnung angewendet und können in IAM-Richtlinien verwendet werden.

   Sie können diesen Schritt wiederholen, um mehrere Regionen hinzuzufügen.
9. Wählen Sie Erstellen aus.

Erstellen Sie eine Pod Identity-Zuordnung (AWS CLI)

1. Wenn Sie Ihrer IAM-Rolle eine vorhandene IAM-Richtlinie zuordnen möchten, fahren Sie mit dem nächsten Schritt fort.

   Erstellen Sie eine IAM-Richtlinie. Sie können Ihre eigene Richtlinie erstellen oder eine AWS verwaltete Richtlinie kopieren, die bereits einige der benötigten Berechtigungen gewährt, und sie an Ihre spezifischen Anforderungen anpassen. Weitere Informationen finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.

   a. Erstellen Sie eine Datei, die die Berechtigungen für die AWS Dienste enthält, auf die Ihre Pods zugreifen sollen. Eine Liste aller Aktionen für alle AWS Dienste finden Sie in der Referenz zur Dienstautorisierung.

      Sie können den folgenden Befehl ausführen, um eine Beispiel-Richtliniendatei zu erstellen, die schreibgeschützten Zugriff auf einen Amazon-S3-Bucket gewährt. Sie können optional Konfigurationsinformationen oder ein Bootstrap-Skript in diesem Bucket speichern und die

Container in Ihrem Pod können die Datei aus dem Bucket lesen und in Ihre Anwendung laden. Wenn Sie diese Beispielrichtlinie erstellen möchten, kopieren Sie den folgenden Inhalt auf Ihr Gerät. *my-pod-secrets-bucket*Ersetzen Sie es durch Ihren Bucket-Namen und führen Sie den Befehl aus.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
```

b. Erstellen Sie die IAM-Richtlinie.

```
aws iam create-policy --policy-name my-policy --policy-document file://my-
policy.json
```

2. Erstellen Sie eine IAM-Rolle und ordnen Sie diese einem Kubernetes- Servicekonto zu.

a. Wenn Sie ein vorhandenes Kubernetes-Servicekonto haben, das eine IAM-Rolle annehmen soll, können Sie diesen Schritt überspringen.

Erstellen Sie ein Kubernetes-Servicekonto. Kopieren Sie den folgenden Inhalt auf Ihr Gerät. *my-service-account*Ersetzen Sie ihn durch Ihren gewünschten Namen und gegebenenfalls *default* durch einen anderen Namespace. Wenn Sie eine Änderung vornehmen*default*, muss der Namespace bereits existieren.

```
cat >my-service-account.yaml <<EOF
apiVersion: v1
kind: ServiceAccount
metadata:
  name: my-service-account
  namespace: default
EOF
kubectl apply -f my-service-account.yaml
```

Führen Sie den folgenden Befehl aus.

```
kubectl apply -f my-service-account.yaml
```

b. Führen Sie den folgenden Befehl aus, um eine Vertrauensrichtlinie für die IAM-Rolle zu erstellen.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEksAuthToAssumeRoleForPodIdentity",
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
```

c. Erstellen Sie die Rolle. Ersetzen Sie *my-role* durch einen Namen für Ihre IAM-Rolle und *my-role-description* durch eine Beschreibung für Ihre Rolle.

```
aws iam create-role --role-name my-role --assume-role-policy-document file://
trust-relationship.json --description "my-role-description"
```

d. Hängen Sie eine IAM-Richtlinie an Ihre Rolle an. Ersetzen Sie *my-role* durch den Namen Ihrer IAM-Rolle und *my-policy* durch den Namen einer vorhandenen Richtlinie, die Sie erstellt haben.

```
aws iam attach-role-policy --role-name my-role --policy-arn=arn:aws:
 iam::111122223333:policy/my-policy
```

> **Note**
>
> Im Gegensatz zu IAM-Rollen für Servicekonten verwendet EKS Pod Identity keine Annotation im Servicekonto.

e. Führen Sie den folgenden Befehl aus, um die Zuordnung zu erstellen. `my-cluster`Ersetzen Sie es durch den Namen des Clusters, *`my-service-account`* ersetzen Sie es durch Ihren gewünschten Namen und *`default`* gegebenenfalls durch einen anderen Namespace.

```
aws eks create-pod-identity-association --cluster-name my-cluster --role-arn
 arn:aws: iam::111122223333:role/my-role --namespace default --service-account my-
service-account
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "association": {
        "clusterName": "my-cluster",
        "namespace": "default",
        "serviceAccount": "my-service-account",
        "roleArn": "arn:aws: iam::111122223333:role/my-role",
        "associationArn": "arn:aws::111122223333:podidentityassociation/my-
cluster/a-abcdefghijklmnop1",
        "associationId": "a-abcdefghijklmnop1",
        "tags": {},
        "createdAt": 1700862734.922,
        "modifiedAt": 1700862734.922
    }
}
```

> **Note**
>
> Sie können den Namen eines Namespace und eines Servicekontos angeben, der bzw. das nicht im Cluster vorhanden ist. Sie müssen den Namespace, das Servicekonto und die Workload erstellen, die das Servicekonto verwendet, damit die EKS-Pod-Identity-Zuordnung funktioniert.

Konfiguration bestätigen

1. Stellen Sie sicher, dass die Vertrauensrichtlinie der IAM-Rolle korrekt konfiguriert ist.

```
aws iam get-role --role-name my-role --query Role.AssumeRolePolicyDocument
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Allow EKS Auth service to assume this role for Pod Identities",
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
```

2. Stellen Sie sicher, dass die Richtlinie, die Sie in einem vorherigen Schritt an Ihre Rolle angehängt haben, an die Rolle angehängt ist.

```
aws iam list-attached-role-policies --role-name my-role --query
 'AttachedPolicies[].PolicyArn' --output text
```

Eine Beispielausgabe sieht wie folgt aus.

```
 arn:aws: iam::111122223333:policy/my-policy
```

3. Legen Sie eine Variable fest, um den Amazon-Ressourcennamen (ARN) der Richtlinie zu speichern, die Sie verwenden möchten. *my-policy*Ersetzen Sie es durch den Namen der Richtlinie, für die Sie die Berechtigungen bestätigen möchten.

```
export policy_arn=arn:aws: iam::111122223333:policy/my-policy
```

4. Zeigen Sie die Standardversion der Richtlinie an.

```
aws iam get-policy --policy-arn $policy_arn
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "EXAMPLEBIOWGLDEXAMPLE",
        "Arn": "arn:aws: iam::111122223333:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        [...]
    }
}
```

5. Überprüfen Sie den Inhalt der Richtlinie, um sicherzustellen, dass sie alle Berechtigungen enthält, die Ihr Pod benötigt. Falls erforderlich, ersetzen Sie *1* den folgenden Befehl durch die Version, die in der vorherigen Ausgabe zurückgegeben wurde.

   ```
   aws iam get-policy-version --policy-arn $policy_arn --version-id v1
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   {
       "Version":"2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": "s3:GetObject",
               "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
           }
       ]
   }
   ```

   Wenn Sie die Beispielrichtlinie in einem vorherigen Schritt erstellt haben, ist Ihre Ausgabe dieselbe. Wenn Sie eine andere Richtlinie erstellt haben, ist der *example* Inhalt anders.

Nächste Schritte

the section called “Pod-Servicekonten”

Zugreifen auf AWS Ressourcen mithilfe von EKS Pod Identity Target IAM-Rollen

Wenn Sie Anwendungen auf Amazon Elastic Kubernetes Service (Amazon EKS) ausführen, müssen Sie möglicherweise auf AWS Ressourcen zugreifen, die in verschiedenen AWS Konten vorhanden sind. In dieser Anleitung erfahren Sie, wie Sie den kontoübergreifenden Zugriff mithilfe von EKS Pod Identity einrichten, sodass Ihre Kubernetes-Pods mithilfe von Zielrollen auf andere AWS Ressourcen zugreifen können.

Voraussetzungen

Stellen Sie vor dem Beginn sicher, dass Sie die folgenden Schritte abgeschlossen haben:

- Einrichtung des Amazon-EKS-Pod-Identity-Agenten
- Erstellung der EKS-Pod-Identity-Rolle

So funktioniert's

Mit Pod Identity können Anwendungen in Ihrem EKS-Cluster über einen Prozess, der als Rollenverkettung bezeichnet wird, kontenübergreifend auf AWS Ressourcen zugreifen.

Wenn Sie eine Pod Identity-Zuordnung erstellen, können Sie zwei IAM-Rollen angeben: eine EKS-Pod-Identity-Rolle in demselben Konto wie Ihr EKS-Cluster und eine Ziel-IAM-Rolle aus dem Konto, das Ihre AWS Ressourcen enthält, auf die Sie zugreifen möchten (wie S3-Buckets oder RDS-Datenbanken). Die EKS-Pod-Identity-Rolle muss sich aufgrund der PassRoleIAM-Anforderungen im Konto Ihres EKS-Clusters befinden, während sich die Target-IAM-Rolle in jedem Konto befinden kann. AWS PassRole ermöglicht es einer AWS Entität, die Rollenübernahme an einen anderen Dienst zu delegieren. EKS Pod Identity verwendet PassRole , um eine Rolle mit einem Kubernetes-Dienstkonto zu verbinden. Dabei müssen sich sowohl die Rolle als auch die Identität, die sie übergibt, in demselben AWS Konto wie der EKS-Cluster befinden. Wenn Ihr Anwendungs-Pod auf AWS Ressourcen zugreifen muss, fordert er Anmeldeinformationen von Pod Identity an. Pod Identity führt dann automatisch zwei Rollenübernahmen nacheinander durch: Zunächst übernimmt es die EKS-Pod-Identity-Rolle und verwendet dann diese Anmeldeinformationen, um die Ziel-IAM-Rolle zu übernehmen. Durch diesen Vorgang erhält Ihr Pod temporäre Anmeldeinformationen, für die die in der Zielrolle definierten Berechtigungen gelten, sodass Sie sicher auf Ressourcen in anderen AWS Konten zugreifen können.

Überlegungen zum Caching

Aufgrund von Caching-Mechanismen werden Aktualisierungen einer IAM-Rolle in einer vorhandenen Pod-Identity-Zuordnung möglicherweise nicht sofort in den Pods wirksam, die auf Ihrem EKS-Cluster ausgeführt werden. Der Pod-Identity-Agent speichert IAM-Anmeldedaten basierend auf der Konfiguration der Zuordnung zum Zeitpunkt des Abrufs der Anmeldedaten zwischen. Wenn die Zuordnung nur eine EKS-Pod-Identity-Rolle und keine Ziel-IAM-Rolle enthält, bleiben die im Cache gespeicherten Anmeldedaten 6 Stunden lang gültig. Wenn die Zuordnung sowohl die EKS-Pod-Identity-Rolle ARN als auch eine Ziel-IAM-Rolle enthält, bleiben die im Cache gespeicherten Anmeldeinformationen 59 Minuten lang gültig. Durch das Ändern einer vorhandenen Zuordnung, z. B. durch Aktualisieren der ARN der EKS-Pod-Identity-Rolle oder Hinzufügen einer Ziel-IAM-Rolle, wird der vorhandene Cache nicht zurückgesetzt. Daher erkennt der Agent Aktualisierungen erst, wenn die zwischengespeicherten Anmeldedaten aktualisiert werden. Um Änderungen schneller anzuwenden, können Sie die vorhandenen Pods neu erstellen. Andernfalls müssen Sie warten, bis der Cache abgelaufen ist.

Schritt 1: Erstellen und Zuordnen einer Ziel-IAM-Rolle

In diesem Schritt erstellen Sie eine sichere Vertrauenskette, indem Sie eine Ziel-IAM-Rolle erstellen und konfigurieren. Zur Veranschaulichung erstellen wir eine neue Target-IAM-Rolle, um eine Vertrauenskette zwischen zwei AWS Konten einzurichten: Die EKS-Pod-Identity-Rolle (z. B.`eks-pod-identity-primary-role`) im AWS Konto des EKS-Clusters erhält die Erlaubnis, die Target-IAM-Rolle (z. B.`eks-pod-identity-aws-resources`) in Ihrem Zielkonto zu übernehmen, wodurch der Zugriff auf AWS Ressourcen wie Amazon S3 S3-Buckets ermöglicht wird.

Ziel-IAM-Rolle erstellen

1. Öffnen Sie die Amazon IAM-Konsole.
2. Vergewissern Sie sich in der oberen Navigationsleiste, dass Sie bei dem Konto angemeldet sind, das die AWS Ressourcen (wie S3-Buckets oder DynamoDB-Tabellen) für Ihre Target IAM-Rolle enthält.
3. Wählen Sie im linken Navigationsbereich Roles aus.
4. Wählen Sie die Schaltfläche „Rolle erstellen“ und anschließend „ AWS Konto“ unter „Vertrauenswürdiger Entitätstyp“.
5. Wählen Sie Anderes AWS Konto, geben Sie Ihre AWS Kontonummer ein (das Konto, für das Ihre EKS Pod Identity-Rolle existiert) und wählen Sie dann Weiter.
6. Fügen Sie die Berechtigungsrichtlinien hinzu, die Sie der Rolle zuordnen möchten (z. B. AmazonS3FullAccess), und wählen Sie dann Weiter.

7. Geben Sie einen Rollennamen ein, z. B. `MyCustomIAMTargetRole`, und wählen Sie dann Rolle erstellen aus.

Aktualisierung der Vertrauensrichtlinie für die Ziel-IAM-Rolle

1. Nach der Erstellung der Rolle werden Sie zur Liste Rollen zurückgeleitet. Suchen und wählen Sie die neue Rolle aus, die Sie im vorherigen Schritt erstellt haben (z. B. `MyCustomIAMTargetRole`).
2. Wählen Sie den Tab Vertraunsbeziehungen.
3. Klicken Sie auf der rechten Seite auf Vertrauensrichtlinie bearbeiten.
4. Ersetzen Sie im Richtlinien-Editor die Standard-JSON-Datei durch Ihre Vertrauensrichtlinie. Ersetzen Sie die Platzhalterwerte für den Rollennamen und `111122223333` im ARN der IAM-Rolle durch die AWS Konto-ID, die Ihren EKS-Cluster hostet. Sie können PrincipalTags in der Rollenvertrauensrichtlinie optional auch festlegen, dass nur bestimmte Dienstkonten aus einem bestimmten Cluster und Namespace Ihre Zielrolle übernehmen. Beispiel:

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ],
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/eks-cluster-arn": "arn:aws:eks:us-
east-1:111122223333:cluster/example-cluster",
          "aws:RequestTag/kubernetes-namespace": "ExampleNameSpace",
          "aws:RequestTag/kubernetes-service-account": "ExampleServiceAccountName"
        },
        "ArnEquals": {
          "aws:PrincipalARN": "arn:aws:iam::111122223333:role/eks-pod-identity-primary-
role"
        }
      }
```

```
    }
  ]
}
```

Mit der obigen Richtlinienrichtlinie kann die Rolle `eks-pod-identeity-primary-role` aus dem AWS Konto 111122223333 mit den entsprechenden EKS Pod Identity-Sitzungstags diese Rolle übernehmen.

Wenn Sie Sitzungs-Tags in Ihrer EKS-Pod-Identität deaktiviert haben, legt die EKS Pod Identity beim Übernehmen einer Ziel-Rolle auch die `sts:ExternalId` mit Informationen zum Cluster, Namespace und Servicekonto eines Pods fest.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "sts:ExternalId": "region/111122223333/cluster-name/namespace/service-
account-name"
        },
        "ArnEquals": {
          "aws:PrincipalARN": "arn:aws:iam::111122223333:role/eks-pod-identity-primary-
role"
        }
      }
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "sts:TagSession"
    }
  ]
}
```

Die oben genannte Richtlinie trägt dazu bei, dass nur der erwartete Cluster, Namespace und das Servicekonto die Zielrolle übernehmen können.

Aktualisierung der Berechtigungsrichtlinie für die EKS-Pod-Identity-Rolle

In diesem Schritt aktualisieren Sie die Berechtigungsrichtlinie der EKS-Pod-Identity-Rolle, die Ihrem Amazon-EKS-Cluster zugeordnet ist, indem Sie die Ziel-IAM-Rolle ARN als Ressource hinzufügen.

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich Cluster und anschließend den Namen des EKS-Clusters, den Sie ändern möchten.
3. Wählen Sie die Registerkarte Zugriff aus.
4. Wählen Sie unter Pod Identity-Zuordnungen Ihre EKS-Pod-Identity-Rolle aus.
5. Wählen Sie Berechtigungen, Berechtigungen hinzufügen und anschließend Inline-Richtlinie erstellen aus.
6. Wählen Sie auf der rechten Seite JSON aus.
7. Ersetzen Sie im Richtlinien-Editor die Standard-JSON-Datei durch Ihre Berechtigungsrichtlinie. Ersetzen Sie den Platzhalterwert für den Rollennamen und `222233334444` in der IAM-Rolle ARN durch Ihre Ziel-IAM-Rolle. Beispiel:

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ],
            "Resource": "arn:aws:iam::222233334444:role/eks-pod-identity-aws-resources"
        }
    ]
}
```

Schritt 2: Zuordnung der Ziel-IAM-Rolle zu einem Kubernetes-Servicekonto

In diesem Schritt erstellen Sie eine Zuordnung zwischen der Ziel-IAM-Rolle und dem Kubernetes-Servicekonto in Ihrem EKS-Cluster.

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich Cluster und anschließend den Namen des Clusters aus, zu dem Sie die Zuordnung hinzufügen möchten.
3. Wählen Sie die Registerkarte Zugriff aus.
4. Wählen Sie in den Pod-Identity-Zuordnungen die Option Erstellen aus.
5. Wählen Sie die EKS-Pod-Identity-Rolle in der IAM-Rolle für Ihre Workloads aus.
6. Wählen Sie die Ziel-IAM-Rolle in der Ziel-IAM-Rolle aus, die von der EKS-Pod-Identity-Rolle übernommen werden soll.
7. Geben Sie im Feld Kubernetes-Namespace den Namen des Namespace ein, in dem Sie die Zuordnung erstellen möchten (z. B. `my-app-namespace`). Dies definiert, wo sich das Servicekonto befindet.
8. Geben Sie im Feld Kubernetes-Servicekonto den Namen des Servicekontos ein (z. B. `my-service-account`), das die IAM-Anmeldeinformationen verwenden soll. Dadurch wird die IAM-Rolle dem Servicekonto zugeordnet.
9. Wählen Sie Erstellen aus, um die Zuordnung zu erstellen.

Pods für den Zugriff auf AWS Dienste mit Dienstkonten konfigurieren

Wenn ein Pod auf AWS Dienste zugreifen muss, müssen Sie ihn für die Verwendung eines Kubernetes-Dienstkontos konfigurieren. Das Dienstkonto muss einer AWS Identity and Access Management (IAM) -Rolle zugeordnet sein, die über Berechtigungen für den Zugriff auf die AWS Dienste verfügt.

- Einen vorhandenen -Cluster. Wenn Sie keine Rolle haben, können Sie sie mithilfe einer der Anleitungen in Erste Schritte erstellen.
- Ein vorhandenes Kubernetes-Servicekonto und eine EKS-Pod-Identity-Zuordnung, die das Servicekonto einer IAM-Rolle zuordnet. Der Rolle muss eine IAM-Richtlinie zugeordnet sein, die die Berechtigungen enthält, die Ihre Pods für die Nutzung AWS von Diensten haben sollen. Weitere Informationen zu Rollen, ihren Vorteilen sowie zu ihrer Erstellung und Konfiguration finden Sie unter the section called “IAM-Rolle zuweisen”.
- Die neueste Version der AWS CLI, die auf Ihrem Gerät installiert und konfiguriert ist, oder AWS CloudShell. Sie können Ihre aktuelle Version mit `aws --version | cut -d / -f2 | cut -d ' ' -f1` überprüfen. Paketmanager wie `yum` `apt-get`, oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der AWS CLI. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im

Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Die in der installierte AWS CLI-Version AWS CloudShell kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zur Aktualisierung finden Sie im AWS CloudShell Benutzerhandbuch unter AWS CLI in Ihrem Home-Verzeichnis installieren.

- Das `kubectl`-Befehlszeilen-Tool ist auf Ihrem Gerät oder in der AWS CloudShell installiert. Die Version kann mit der Kubernetes-Version Ihres Clusters identisch sein oder bis zu einer Nebenversion älter oder neuer sein. Wenn Ihre Clusterversion beispielsweise `1.29` ist, können Sie `kubectl`-Version `1.28`, `1.29`, oder `1.30` damit verwenden. Informationen zum Installieren oder Aktualisieren von `kubectl` finden Sie unter the section called “kubectl und eksctl einrichten”.
- Eine vorhandene `kubectl config`-Datei, die Ihre Clusterkonfiguration enthält. Informationen zum Erstellen einer `kubectl config`-Datei finden Sie unter the section called “Zugriff auf den Cluster mit kubectl”.

1. Verwenden Sie den folgenden Befehl, um ein Bereitstellungs-Manifest zu erstellen, mit dem Sie einen Pod bereitstellen können, um die Konfiguration zu bestätigen. Ersetzen Sie die Beispielwerte durch eigene Werte.

```
cat >my-deployment.yaml <<EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      serviceAccountName: my-service-account
      containers:
      - name: my-app
        image: public.ecr.aws/nginx/nginx:X.XX
EOF
```

2. Stellen Sie das Manifest in Ihrem Cluster bereit.

```
kubectl apply -f my-deployment.yaml
```

3. Vergewissern Sie sich, dass die erforderlichen Umgebungsvariablen für Ihren Pod vorhanden sind.

   a. Zeigen Sie die Pods an, die im vorherigen Schritt bereitgestellt wurden.

   ```
   kubectl get pods | grep my-app
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   my-app-6f4dfff6cb-76cv9   1/1     Running   0          3m28s
   ```

   b. Vergewissern Sie sich, dass der Pod eine Token-Datei-Einbindung für ein Servicekonto hat.

   ```
   kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep
    AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE:
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE:  /var/run/secrets/
   pods.eks.amazonaws.com/serviceaccount/eks-pod-identity-token
   ```

4. Vergewissern Sie sich, dass Ihre Pods mithilfe der Berechtigungen, die Sie in der Ihrer Rolle beigefügten IAM-Richtlinie zugewiesen haben, mit den AWS Diensten interagieren können.

   > **Note**
   >
   > Wenn ein Pod AWS Anmeldeinformationen von einer IAM-Rolle verwendet, die mit einem Dienstkonto verknüpft ist, verwendet die AWS CLI oder andere SDKs in den Containern für diesen Pod die Anmeldeinformationen, die von dieser Rolle bereitgestellt werden. Der Pod hat weiterhin Zugriff auf die der Amazon-EKS-Knoten-IAM-Rolle bereitgestellten Anmeldeinformationen, es sei denn, Sie beschränken den Zugriff auf diese Anmeldeinformationen. Weitere Informationen finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.

   Wenn Ihre Pods nicht wie erwartet mit den Services interagieren können, führen Sie die folgenden Schritte aus, um sicherzustellen, dass alles richtig konfiguriert ist.

a. Vergewissern Sie sich, dass Ihre Pods eine AWS SDK-Version verwenden, die die Übernahme einer IAM-Rolle über eine EKS-Pod-Identity-Zuordnung unterstützt. Weitere Informationen finden Sie unter the section called “Unterstützte SDKs”.

b. Stellen Sie sicher, dass die Bereitstellung das Servicekonto verwendet.

```
kubectl describe deployment my-app | grep "Service Account"
```

Eine Beispielausgabe sieht wie folgt aus.

```
Service Account:  my-service-account
```

Pods Zugriff auf AWS Ressourcen basierend auf Tags gewähren

Die attributbasierte Zugriffskontrolle (ABAC) gewährt Benutzern Rechte anhand von Richtlinien, die Attribute miteinander kombinieren. EKS Pod Identity fügt den temporären Anmeldeinformationen für jeden Pod Tags mit Attributen wie Cluster-Name, Namespace und Servicekonto-Name hinzu. Diese Rollensitzungs-Tags ermöglichen es Administratoren, eine einzelne Rolle zu erstellen, die für alle Dienstkonten verwendet werden kann, indem sie den Zugriff auf AWS Ressourcen auf der Grundlage übereinstimmender Tags ermöglicht. Durch den zusätzlichen Support für Rollensitzungs-Tags können Kunden engere Sicherheitsgrenzen zwischen Clustern und Workloads innerhalb von Clustern durchsetzen und gleichzeitig dieselben IAM-Rollen und IAM-Richtlinien wiederverwenden.

Beispiel-Richtlinie mit Tags

Nachfolgend finden Sie ein Beispiel für eine IAM-Richtlinie, die `s3:GetObject`-Berechtigungen gewährt, wenn das entsprechende Objekt mit dem EKS-Clusternamen gekennzeichnet ist.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
```

```
            "Action": [
                "s3:GetObject",
                "s3:GetObjectTagging"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:ExistingObjectTag/eks-cluster-name": "${aws:PrincipalTag/eks-
cluster-name}"
                }
            }
        }
    ]
}
```

Aktivieren oder Deaktivieren von Sitzungs-Tags

EKS Pod Identity fügt einen vordefinierten Satz von Sitzungs-Tags hinzu, wenn es die Rolle übernimmt. Diese Sitzungs-Tags ermöglichen es Administratoren, eine einzelne Rolle zu erstellen, die ressourcenübergreifend verwendet werden kann, indem sie den Zugriff auf AWS Ressourcen auf der Grundlage übereinstimmender Tags ermöglicht.

Aktivieren von Sitzungs-Tags

Sitzungstags werden automatisch mit EKS Pod Identity aktiviert – Ihrerseits sind keine Maßnahmen erforderlich. Standardmäßig fügt EKS Pod Identity Ihrer Sitzung eine Reihe vordefinierter Tags hinzu. Um in Richtlinien auf diese Tags zu verweisen, verwenden Sie die Syntax `${aws:PrincipalTag/` gefolgt vom Tag-Schlüssel. Beispiel, `${aws:PrincipalTag/kubernetes-namespace}`.

- `eks-cluster-arn`
- `eks-cluster-name`
- `kubernetes-namespace`
- `kubernetes-service-account`
- `kubernetes-pod-name`
- `kubernetes-pod-uid`

Deaktivieren von Sitzungs-Tags

AWS komprimiert Inline-Sitzungsrichtlinien, verwaltete Richtlinien ARNs und Sitzungs-Tags in ein gepacktes Binärformat mit separatem Limit. Wenn Sie einen `PackedPolicyTooLarge`-Fehler

erhalten, die darauf hinweist, dass das gepackte Binärformat die Größenbeschränkung überschritten hat, können Sie diese Größe reduzieren, indem Sie die von EKS Pod Identity hinzugefügten Sitzungs-Tags deaktivieren. Um diese Sitzungs-Tags zu deaktivieren, führen Sie die folgenden Schritte aus:

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich Clusters und anschließend den Namen des Clusters, den Sie ändern möchten.
3. Wählen Sie die Registerkarte Zugriff aus.
4. Wählen Sie unter Pod-Identity-Zuordnungen die Zuordnungs-ID, die Sie ändern möchten, und klicken Sie dann auf Bearbeiten.
5. Wählen Sie unter Sitzungs-Tags die Option Sitzungs-Tags deaktivieren.
6. Wählen Sie Änderungen speichern aus.

Kontoübergreifende Tags

Alle Sitzungs-Tags, die von EKS Pod Identity hinzugefügt werden, sind transitiv. Die Tag-Schlüssel und -Werte werden an alle `AssumeRole`-Aktionen weitergegeben, die Ihre Workloads verwenden, um Rollen auf ein anderes Konto zu wechseln. Sie können diese Tags in Richtlinien in anderen Konten verwenden, um den Zugriff in kontoübergreifenden Szenarien einzuschränken. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Verketten von Rollen mit Sitzungs-Tags.

Benutzerdefinierte Tags

EKS Pod Identity kann der `AssumeRole`-Aktion, die es ausführt, keine zusätzlichen benutzerdefinierten Tags hinzufügen. Tags, die Sie auf die IAM-Rolle anwenden, sind jedoch immer im gleichen Format verfügbar: `${aws:PrincipalTag/` gefolgt vom Schlüssel, zum Beispiel `${aws:PrincipalTag/MyCustomTag}`.

> **Note**
>
> Tags, die der Sitzung über die `sts:AssumeRole`-Anfrage hinzugefügt wurden, haben im Konfliktfall Vorrang. Nehmen wir zum Beispiel an:
>
> - Amazon EKS fügt der Sitzung einen Schlüssel `eks-cluster-name` und einen Wert `my-cluster` hinzu, wenn EKS die Kundenrolle übernimmt und

- Sie fügen der IAM-Rolle ein `eks-cluster-name`-Tag mit dem Wert `my-own-cluster` hinzu.

In diesem Fall hat Ersteres Vorrang und der Wert für das `eks-cluster-name`-Tag ist `my-cluster`.

Verwendung der Pod Identity mit dem AWS-SDK

Verwenden von EKS-Pod-Identity-Anmeldeinformationen

Um die Anmeldeinformationen von einer EKS-Pod-Identity-Zuordnung zu nutzen, kann Ihr Code ein beliebiges AWS-SDK verwenden, um einen Client für einen AWS-Service mit einem SDK zu erstellen. Standardmäßig sucht das SDK in einer Kette von Speicherorten nach zu verwendenden Anmeldeinformationen für die AWS-Identitäts- und Zugriffsverwaltung. Die Anmeldeinformationen für EKS Pod Identity werden verwendet, wenn Sie bei der Erstellung des Clients keinen Anmeldeinformationsanbieter angeben oder Sie das SDK anderweitig initialisiert haben.

Das funktioniert, weil EKS-Pod-Identitäten dem Anbieter für Container-Anmeldeinformationen hinzugefügt wurden, der in einem Schritt in der standardmäßigen Anmeldeinformationskette durchsucht wird. Wenn Ihre Workloads derzeit Anmeldeinformationen verwenden, die in der Kette der Anmeldeinformationen weiter vorn stehen, werden diese Anmeldeinformationen auch dann weiter verwendet, wenn Sie eine EKS-Pod-Identity-Zuordnung für dieselbe Workload konfigurieren.

Weitere Informationen zur Funktionsweise von EKS-Pod-Identitäten finden Sie unter the section called “Funktionsweise”.

Bei der Verwendung von Erfahren Sie, wie EKS Pod Identity Pods Zugriff auf AWS-Services gewährt müssen die Container in Ihren Pods eine AWS-SDK-Version verwenden, welche die Übernahme einer IAM-Rolle vom EKS-Pod-Identity-Agent unterstützt. Stellen Sie sicher, dass Sie die folgenden Versionen oder höher für Ihr AWS-SDK verwenden:

- Java (Version 2) – 2.21.30
- Java – 1.12.746
- Go v1 – v1.47.11
- Go v2 – release-2023-11-14

- Python (Boto3) – 1.34.41
- Python (botocore) – 1.34.41
- AWS-CLI – 1.30.0

  AWS-CLI – 2.15.0
- JavaScript v2 – 2.1550.0
- JavaScript v3 – v3.458.0
- Kotlin – v1.0.1
- Ruby – 3.188.0
- Rust – release-2024-03-13
- C++ – 1.11.263
- .NET – 3.7.734.0
- PowerShell – 4.1.502
- PHP – 3.289.0

Um sicherzustellen, dass Sie ein unterstütztes SDK verwenden, befolgen Sie die Installationsanweisungen für Ihr bevorzugtes SDK unter Tools für die Entwickelung in AWS, wenn Sie Ihre Container entwickeln.

Eine Liste der Add-Ons, die EKS Pod Identity unterstützen, finden Sie unter the section called "Unterstützungsreferenz für Pod Identity".

Deaktivierung von **`IPv6`** im EKS-Pod-Identity-Agent

AWS-Managementkonsole

1. Um `IPv6` im EKS-Pod-Identity-Agent zu deaktivieren, fügen Sie die folgende Konfiguration zu den Optionalen Konfigurationseinstellungen des EKS- Add-Ons hinzu.
   a. Öffnen Sie die Amazon-EKS-Konsole.
   b. Wählen Sie im linken Navigationsbereich Clusters (Cluster) aus. Wählen Sie anschließend den Namen des Clusters aus, für den Sie das Add-On konfigurieren möchten.
   c. Wählen Sie die Registerkarte Add-ons.
   d. Aktivieren Sie das Kontrollkästchen oben rechts im Feld „EKS-Pod-Identity-Agent-Add-on“ und wählen Sie anschließend Bearbeiten.
   e. Auf der Seite EKS-Pod-Identity-Agent konfigurieren:

i. Wählen Sie die Version aus, die Sie verwenden möchten. Wir empfehlen Ihnen, die gleiche Version wie im vorherigen Schritt beizubehalten und die Version und Konfiguration in separaten Schritten zu aktualisieren.

ii. Erweitern Sie Optionale Konfigurationseinstellungen.

iii. Geben Sie den JSON-Schlüssel `"agent":` und den Wert eines verschachtelten JSON-Objekts mit einem Schlüssel `"additionalArgs":` in den Konfigurationswerten ein. Der resultierende Text muss ein gültiges JSON-Objekt sein. Wenn dieser Schlüssel und dieser Wert die einzigen Daten im Textfeld sind, setzen Sie den Schlüssel und den Wert in geschweifte Klammern `{ }`. Das folgende Beispiel veranschaulicht, dass die Netzwerkrichtlinie aktiviert ist:

```
{
    "agent": {
        "additionalArgs": {
            "-b": "169.254.170.23"
        }
    }
}
```

Diese Konfiguration legt die `IPv4`-Adresse als einzige Adresse fest, die vom Agenten verwendet wird.

f. Um die neue Konfiguration durch Ersetzen des Agent-Pods von EKS Pod Identity anzuwenden, wählen Sie Änderungen speichern.

Amazon EKS wendet Änderungen an den EKS-Add-Ons an, indem es einen Einführung von Kubernetes `DaemonSet` für EKS Pod Identity Agent durchführt. Sie können den Status der Einführung im Aktualisierungsverlauf des Add-Ons in AWS-Managementkonsole und mit `kubectl rollout status daemonset/eks-pod-identity-agent --namespace kube-system` verfolgen.

`kubectl rollout` verfügt über die folgenden Befehle:

```
$ kubectl rollout

history  -- View rollout history
pause    -- Mark the provided resource as paused
restart  -- Restart a resource
resume   -- Resume a paused resource
```

```
status   -- Show the status of the rollout
undo     -- Undo a previous rollout
```

Wenn die Einführung zu lange dauert, macht Amazon EKS die Einführung rückgängig und eine Nachricht mit dem Typ der Add-On-Aktualisierung und dem Status Fehlgeschlagen dem Aktualisierungsverlauf des Add-Ons hinzugefügt. Um etwaige Probleme zu untersuchen, beginnen Sie mit dem Verlauf der Einführung und führen Sie `kubectl logs` auf einem Agent-Pod von EKS Pod Identity aus, um die Protokolle des EKS-Pod-Identity-Agent anzuzeigen.

2. Wenn der neue Eintrag im Aktualisierungsverlauf den Status Erfolgreich aufweist, ist die Einführung abgeschlossen und das Add-On verwendet die neue Konfiguration in allen Pods von EKS Pod Identity Agent.

AWS-CLI

1. Um `IPv6` im EKS-Pod-Identity-Agent zu deaktivieren, fügen Sie die folgende Konfiguration zu den Konfigurationswerten des EKS-Add-Ons hinzu.

   Führen Sie den folgenden AWS-CLI-Befehl aus. Ersetzen Sie `my-cluster` durch den Namen Ihres Clusters und den IAM-Rollen-ARN durch die Rolle, die Sie verwenden.

   ```
   aws eks update-addon --cluster-name my-cluster --addon-name eks-pod-identity-agent \
       --resolve-conflicts PRESERVE --configuration-values '{"agent":{"additionalArgs":
    { "-b": "169.254.170.23"}}}'
   ```

   Diese Konfiguration legt die `IPv4`-Adresse als einzige Adresse fest, die vom Agenten verwendet wird.

   Amazon EKS wendet Änderungen an den EKS-Add-Ons an, indem es eine Einführung von Kubernetes DaemonSet für EKS-Pod-Identity-Agent durchführt. Sie können den Status der Einführung im Aktualisierungsverlauf des Add-Ons in AWS-Managementkonsole und mit `kubectl rollout status daemonset/eks-pod-identity-agent --namespace kube-system` verfolgen.

   `kubectl rollout` verfügt über die folgenden Befehle:

   ```
   kubectl rollout

   history  -- View rollout history
   ```

```
pause    -- Mark the provided resource as paused
restart  -- Restart a resource
resume   -- Resume a paused resource
status   -- Show the status of the rollout
undo     -- Undo a previous rollout
```

Wenn die Einführung zu lange dauert, macht Amazon EKS die Einführung rückgängig und eine Nachricht mit dem Typ der Add-On-Aktualisierung und dem Status Fehlgeschlagen dem Aktualisierungsverlauf des Add-Ons hinzugefügt. Um etwaige Probleme zu untersuchen, beginnen Sie mit dem Verlauf der Einführung und führen Sie `kubectl logs` auf einem Agent-Pod von EKS Pod Identity aus, um die Protokolle des EKS-Pod-Identity-Agent anzuzeigen.

Erstellen einer IAM-Rolle mit der von EKS Pod Identity geforderten Vertrauensrichtlinie

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEksAuthToAssumeRoleForPodIdentity",
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
```

`sts:AssumeRole`

EKS Pod Identity nutzt `AssumeRole`, um die IAM-Rolle anzunehmen, bevor es die temporären Anmeldeinformationen an Ihre Pods übergibt.

`sts:TagSession`

EKS Pod Identity verwendet`TagSession`, um Sitzungs-Tags in die Anfragen an STS aufzunehmen. AWS

Einstellungsbedingungen

Sie können diese Tags in den Bedingungsschlüsseln der Vertrauensrichtlinie verwenden, um einzuschränken, welche Servicekonten, Namespaces und Cluster diese Rolle verwenden dürfen. Die Liste der von Pod Identity hinzugefügten Anforderungs-Tags finden Sie unter the section called “Aktivieren oder Deaktivieren von Sitzungs-Tags”.

Sie können beispielsweise mit der folgenden Vertrauensrichtlinie mit dem hinzugefügten `Condition` einschränken, welche Pods die Rolle einer Pod-Identity-IAM-Rolle auf ein bestimmtes `ServiceAccount` und `Namespace` übernehmen können:

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEksAuthToAssumeRoleForPodIdentity",
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/kubernetes-namespace": [
                        "Namespace"
                    ],
                    "aws:RequestTag/kubernetes-service-account": [
                        "ServiceAccount"
                    ]
                }
            }
        }
    ]
}
```

Eine Liste der Amazon EKS-Bedingungsschlüssel finden Sie unter Bedingungsschlüssel für Amazon Elastic Kubernetes Service in der Service-Autorisierungsreferenz. Um zu erfahren, mit welchen

Aktionen und Ressourcen Sie einen Bedingungsschlüssel verwenden können, lesen Sie von Amazon Elastic Kubernetes Service definierte Aktionen.

Verwaltung von Datenverarbeitungsressourcen mithilfe von Knoten

Ein Kubernetes-Knoten ist ein Rechner, auf dem containerisierte Anwendungen ausgeführt werden. Jeder Knoten umfasst die folgenden Komponenten:

- Container-Laufzeit – Software, die für den Betrieb der Container verantwortlich ist.
- kubelet – Stellt sicher, dass die Container fehlerfrei sind und innerhalb ihres zugeordneten Pods ausgeführt werden.
- kube-proxy – Behält die Netzwerkregeln bei, die die Kommunikation mit Ihren Pods ermöglichen.

Weitere Informationen finden Sie unter Nodes in der Kubernetes-Dokumentation.

Ihr Amazon-EKS-Cluster kann Pods auf einer beliebigen Kombination aus im EKS Auto Mode verwalteten Knoten, selbstverwalteten Knoten, von Amazon EKS verwalteten Knotengruppen, AWS Fargate und Amazon EKS Hybrid Nodes planen. Weitere Informationen zu Knoten, die in Ihrem Cluster bereitgestellt werden, finden Sie unter the section called “Zugriff auf Cluster-Ressourcen”.

> **Note**
>
> Mit Ausnahme von Hybridknoten müssen sich die Knoten in derselben VPC befinden wie die Subnetze, die Sie beim Erstellen des Clusters ausgewählt haben. Die Knoten müssen sich jedoch nicht in denselben Subnetzen befinden.

Vergleich von Rechenoptionen

Die folgende Tabelle enthält mehrere Kriterien, die bei der Entscheidung ausgewertet werden müssen, welche Optionen Ihren Anforderungen am besten entsprechen. Selbstverwaltete Knoten sind eine weitere Option, die alle aufgeführten Kriterien erfüllen, jedoch erfordern sie einen deutlich höheren manuellen Wartungsaufwand. Weitere Informationen finden Sie unter the section called “Selbstverwaltete Knoten”.

> **Note**
>
> Bottlerocket weist einige spezifische Unterschiede zu den allgemeinen Informationen in dieser Tabelle auf. Weitere Informationen finden Sie in der Bottlerocket-Dokumentation auf GitHub.

Kriterien	EKS-verwaltete Knotengruppen	EKS Auto Mode	Amazon EKS Hybrid Nodes
Kann in AWS Outposts bereitgestellt werden	Nein	Nein	Nein
Kann auf eine AWS Lokale Zone bereitgestellt werden	Ja	Nein	Nein
Kann Container ausführen, die Windows benötigen	Ja	Nein	Nein
Kann Container ausführen, die Linux benötigen	Ja	Ja	Ja
Kann Workloads ausführen, die den Inferentia-Chip benötigen	Ja – nur Amazon-Linux-Knoten	Ja	Nein
Kann Workloads ausführen, die eine GPU benötigen	Ja – nur Amazon-Linux-Knoten	Ja	Ja
Kann Workloads ausführen, die	Ja	Ja	Ja

Kriterien	EKS-verwaltete Knotengruppen	EKS Auto Mode	Amazon EKS Hybrid Nodes
Arm-Prozessoren benötigen			
Kann ausgeführt werdenAWS Bottlerocket	Ja	Ja	Nein
Pods teilen CPU, Arbeitsspeicher, Speicher und Netzwerkressourcen mit anderen Pods.	Ja	Ja	Ja
Bereitstellen und Verwalten von Amazon-EC2-Instances	Ja	Nein – Erfahren Sie mehr über verwaltete EC2-Instances	Ja – die physischen oder virtuellen Maschinen On-Premises werden von Ihnen mit den Tools Ihrer Wahl verwaltet.
Das Betriebssystem von Amazon-EC2-Instances muss gesichert, gewartet und gepatcht werden	Ja	Nein	Ja – das auf Ihren physischen oder virtuellen Maschinen ausgeführte Betriebssystem wird von Ihnen mit den Tools Ihrer Wahl verwaltet.
Kann Bootstrap-Argumente bei der Bereitstellung eines Knotens bereitstellen, z. B. zusätzliche kubelet-Argumente.	Ja – mithilfe von `eksctl` oder einer Startvorlage mit einem benutzerdefinierten AMI.	Nein – Verwenden Sie zur Konfiguration der Knoten eine `NodeClass`	Ja – Sie können Bootstrap-Argumente mit nodeadm anpassen. Siehe the section called “nodeadm für Hybridknoten”.

Kriterien	EKS-verwaltete Knotengruppen	EKS Auto Mode	Amazon EKS Hybrid Nodes
IP-Adressen können Pods aus einem anderen CIDR-Block als der dem Knoten zugewiesenen IP-Adresse zuweisen.	Ja – Verwenden einer Startvorlage mit einem benutzerdefinierten AMI. Weitere Informationen finden Sie unter the section called “Startvorlagen”.	Nein	Ja – siehe the section called “CNI konfigurieren”.
SSH im Knoten nicht möglich	Ja	Nein – Erfahren Sie, wie Sie Fehler an Knoten beheben	Ja
Kann Ihr eigenes benutzerdefiniertes AMI auf Knoten bereitstellen	Ja – Verwenden eines Startvorlage	Nein	Ja
Kann Ihr eigenes benutzerdefiniertes CNI auf Knoten bereitstellen	Ja – Verwenden eines Startvorlage mit einem benutzerdefinierten AMI	Nein	Ja

Kriterien	EKS-verwaltete Knotengruppen	EKS Auto Mode	Amazon EKS Hybrid Nodes
Knoten-AMI muss selbst aktualisiert werden	Ja – Wenn Sie ein für Amazon-EKS-optimiertes AMI bereitgestellt haben, werden Sie in der Amazon-EKS-Konsole benachrichtigt, wenn Aktualisierungen verfügbar sind. Sie können die Aktualisierung mit einem Klick in der Konsole durchführen. Wenn Sie ein benutzerdefiniertes AMI bereitgestellt haben, werden Sie in der Amazon-EKS-Konsole nicht benachrichtigt, wenn Aktualisierungen verfügbar sind. Sie müssen die Aktualisierung selbst durchführen.	Nein	Ja – das auf Ihren physischen oder virtuellen Maschinen ausgeführte Betriebssystem wird von Ihnen mit den Tools Ihrer Wahl verwaltet. Siehe the section called “Betriebssystem vorbereiten”.

Kriterien	EKS-verwaltete Knotengruppen	EKS Auto Mode	Amazon EKS Hybrid Nodes
Muss die Kubernetes-Version des Knotens selbst aktualisieren	Ja – Wenn Sie ein für Amazon-EKS-optimiertes AMI bereitgestellt haben, werden Sie in der Amazon-EKS-Konsole benachrichtigt, wenn Aktualisierungen verfügbar sind. Sie können die Aktualisierung mit einem Klick in der Konsole durchführen. Wenn Sie ein benutzerdefiniertes AMI bereitgestellt haben, werden Sie in der Amazon-EKS-Konsole nicht benachrichtigt, wenn Aktualisierungen verfügbar sind. Sie müssen die Aktualisierung selbst durchführen.	Nein	Ja – Sie verwalten Upgrades von Hybridknoten mit den Tools Ihrer Wahl oder mit `nodeadm`. Siehe the section called “Hybridknoten aktualisieren”.
Kann Amazon-EBS-Speicher mit Pods verwenden	Ja	Ja, als integrierte Funktion. Erfahren Sie, wie Sie eine Speicherklasse erstellen.	Nein
Kann Amazon-EFS-Speicher mit Pods verwenden	Ja	Ja	Nein

Kriterien	EKS-verwaltete Knotengruppen	EKS Auto Mode	Amazon EKS Hybrid Nodes
Kann Amazon FSx für Lustre Speicher mit Pods verwenden	Ja	Ja	Nein
Kann Network Load Balancer für Services verwenden	Ja	Ja	Ja – Zieltyp `ip` muss verwendet werden.
Pods können in einem öffentlichen Subnetz ausgeführt werden	Ja	Ja	Nein – Pods werden in einer On-Premises-Umgebung ausgeführt.
Kann einzelnen Pods verschiedene VPC-Sicherheitsgruppen zuweisen	Ja – Nur Linux-Knoten	Nein	Nein
Kann Kubernetes DaemonSets ausführen	Ja	Ja	Ja
Unterstützung für `HostPort` und `HostNetwork` im Pod-Manifest	Ja	Ja	Ja
AWS-Verfügbarkeit in Regionen	Alle von Amazon EKS unterstützten Regionen	Alle von Amazon EKS unterstützten Regionen	Alle von Amazon EKS unterstützten Regionen mit Ausnahme der AWS GovCloud (US)-Regionen und der China-Regionen.

Kriterien	EKS-verwaltete Knotengruppen	EKS Auto Mode	Amazon EKS Hybrid Nodes
Kann Container auf Amazon-EC2-Dedicated-Hosts ausführen	Ja	Nein	Nein
Preise	Kosten einer Amazon-EC2-Instance, die mehrere Pods ausführt. Weitere Informationen dazu finden Sie unter Amazon EC2 – Preise.	Wenn EKS Auto Mode in Ihrem Cluster aktiviert ist, zahlen Sie zusätzlich zu den Standardgebühren für EC2-Instances eine separate Gebühr für die Instances, die mit der Rechenleistung des Automatikmodus gestartet werden. Der Betrag hängt vom gestarteten Instance-Typ und der AWS-Region ab, in der sich Ihr Cluster befindet. Weitere Informationen finden Sie unter Amazon EKS – Preise.	Kosten für Hybridknoten-vCPU pro Stunde. Weitere Informationen finden Sie unter Amazon EKS – Preise.

Vereinfachung des Knotenlebenszyklus mit verwalteten Knotengruppen

Von Amazon EKS verwaltete Knotengruppen automatisieren die Bereitstellung und das Lebenszyklusmanagement von Knoten (EC2 Amazon-Instances) für Amazon EKS Kubernetes-Cluster.

Mit Amazon EKS-verwalteten Knotengruppen müssen Sie die EC2 Amazon-Instances, die Rechenkapazität für die Ausführung Ihrer Kubernetes-Anwendungen bereitstellen, nicht separat bereitstellen oder registrieren. Sie können Knoten für Ihren Cluster mit einem einzigen Vorgang erstellen, aktualisieren oder beenden. Knotenaktualisierungen und -beendigungen entleeren Knoten automatisch, um sicherzustellen, dass Ihre Anwendungen verfügbar bleiben.

Jeder verwaltete Knoten wird als Teil einer Amazon EC2 Auto Scaling Scaling-Gruppe bereitgestellt, die von Amazon EKS für Sie verwaltet wird. Jede Ressource, einschließlich der Instances und Auto Scaling Scaling-Gruppen, wird in Ihrem AWS Konto ausgeführt. Jede Knotengruppe wird in mehreren Availability Zones ausgeführt, die Sie definieren.

Verwaltete Knotengruppen können optional auch die automatische Knotenreparatur nutzen, die den Zustand der Knoten kontinuierlich überwacht. Sie reagiert automatisch auf erkannte Probleme und ersetzt Knoten, wenn möglich. Dies unterstützt die Gesamtverfügbarkeit des Clusters mit minimalen manuellen Eingriffen. Weitere Informationen finden Sie unter the section called “Knotenintegrität”.

Sie können mithilfe der Amazon EKS-Konsole, AWS CLI, `eksctl` AWS API oder Infrastruktur als Code-Tools eine verwaltete Knotengruppe zu neuen oder vorhandenen Clustern hinzufügen, einschließlich AWS CloudFormation. Knoten, die als Teil einer verwalteten Knotengruppe gestartet werden, werden vom Kubernetes Cluster Autoscaler automatisch für die automatische Erkennung mit Tags versehen. Sie können die Knotengruppe verwenden, um Kubernetes-Labels auf Knoten anzuwenden und sie jederzeit zu aktualisieren.

Für die Nutzung von Amazon EKS-verwalteten Knotengruppen fallen keine zusätzlichen Kosten an. Sie zahlen nur für die AWS Ressourcen, die Sie bereitstellen. Dazu gehören EC2 Amazon-Instances, Amazon EBS-Volumes, Amazon EKS-Cluster-Stunden und jede andere AWS Infrastruktur. Es fallen keine Mindestgebühren oder Vorauszahlungen an.

Weitere Informationen zu den ersten Schritten mit einem neuen Amazon-EKS-Cluster und einer verwalteten Knotengruppe finden Sie unter the section called “Cluster erstellen (Konsole und CLI)”.

Informationen zum Hinzufügen einer verwalteten Knotengruppe zu einem bestehenden Cluster finden Sie unter the section called “Create”.

Konzepte für verwaltete Knotengruppen

- Von Amazon EKS verwaltete Knotengruppen erstellen und verwalten EC2 Amazon-Instances für Sie.
- Jeder verwaltete Knoten wird als Teil einer Amazon EC2 Auto Scaling Scaling-Gruppe bereitgestellt, die von Amazon EKS für Sie verwaltet wird. Darüber hinaus werden alle Ressourcen,

einschließlich EC2 Amazon-Instances und Auto Scaling Scaling-Gruppen, in Ihrem AWS Konto ausgeführt.

- Die Auto-Scaling-Gruppe einer verwalteten Knotengruppe umfasst alle Subnetze, die Sie beim Erstellen der Gruppe angeben.
- Amazon EKS kennzeichnet verwaltete Knotengruppenressourcen, sodass sie für die Verwendung des Kubernetes Cluster Autoscaler konfiguriert sind.

> **Important**
>
> Wenn Sie eine zustandsbehaftete Anwendung über mehrere Availability Zones hinweg ausführen, die von Amazon-EBS-Volumes gesichert wird und den Kubernetes Cluster Autoscaler verwendet, sollten Sie mehrere Knotengruppen konfigurieren, die jeweils für eine einzelne Availability Zone gelten. Außerdem sollten Sie das `--balance-similar-node-groups`-Feature aktivieren.

- Sie können eine benutzerdefinierte Startvorlage für ein höheres Maß an Flexibilität und Anpassung bei der Bereitstellung verwalteter Knoten verwenden. Zum Beispiel können Sie zusätzliche `kubelet`-Argumente angeben und ein benutzerdefiniertes AMI verwenden. Weitere Informationen finden Sie unter the section called “Startvorlagen”. Wenn Sie beim ersten Erstellen einer verwalteten Knotengruppe keine benutzerdefinierte Startvorlage verwenden, gibt es eine automatisch generierte Startvorlage. Ändern Sie diese automatisch generierte Vorlage nicht manuell sonst treten Fehler auf.
- Amazon EKS folgt dem Modell der gemeinsamen Verantwortung für CVEs und Sicherheitspatches auf verwalteten Knotengruppen. Wenn verwaltete Knoten ein für Amazon EKS optimiertes AMI ausführen, ist Amazon EKS für die Erstellung von Patch-Versionen des AMI verantwortlich, wenn Fehler oder Probleme gemeldet werden. Wir können eine Korrektur veröffentlichen. Sie sind jedoch dafür verantwortlich, diese Patch-AMI-Versionen für Ihre verwalteten Knotengruppen bereitzustellen. Wenn verwaltete Knoten ein benutzerdefiniertes AMI ausführen, sind Sie für die Erstellung von Patch-Versionen des AMI verantwortlich, wenn Fehler oder Probleme gemeldet werden, und für die anschließende Bereitstellung des AMI. Weitere Informationen finden Sie unter the section called “Aktualisierung”.
- Von Amazon EKS verwaltete Knotengruppen können sowohl in öffentlichen als auch privaten Subnetzen gestartet werden. Wenn Sie eine verwaltete Knotengruppe am oder nach dem 22. April 2020 in einem öffentlichen Subnetz starten, muss `MapPublicIpOnLaunch` für das Subnetz auf „Wahr“ gesetzt sein, damit die Instances einem Cluster hinzugefügt werden können. Wenn das öffentliche Subnetz am `eksctl` oder nach dem 26. März 2020 mithilfe der von Amazon EKS

bereitgestellten AWS CloudFormation Vorlagen erstellt wurde, ist diese Einstellung bereits auf true gesetzt. Wenn die öffentlichen Subnetze vor dem 26. März 2020 erstellt wurden, müssen Sie die Einstellung manuell ändern. Weitere Informationen finden Sie unter Ändern des Attributs für die öffentliche IPv4 Adressierung für Ihr Subnetz.

- Wenn Sie eine verwaltete Knotengruppe in privaten Subnetzen bereitstellen, müssen Sie sicherstellen, dass diese auf Amazon ECR zugreifen kann, um Container-Images abzurufen. Sie können dies tun, indem Sie ein NAT-Gateway mit der Routing-Tabelle des Subnetzes verbinden oder indem Sie die folgenden AWS PrivateLink -VPC-Endpunkte hinzufügen:
  - Amazon-ECR-API-Endpunktschnittstelle – `com.amazonaws.region-code.ecr.api`
  - API-Endpunktschnittstelle der Amazon-ECR-Docker-Registrierung – `com.amazonaws.region-code.ecr.dkr`
  - Amazon-S3-Gateway-Endpunkt – `com.amazonaws.region-code.s3`

  Weitere häufig verwendete Services und Endpunkte finden Sie unter the section called “Private Cluster”.
- Verwaltete Knotengruppen können nicht in AWS Outposts oder in AWS Wavelength bereitgestellt werden. Verwaltete Knotengruppen können AWS Local Zones erstellt werden. Weitere Informationen finden Sie unter the section called “ AWS Local Zones”.
- Sie können mehrere verwaltete Knotengruppen innerhalb eines einzelnen Clusters erstellen. So können Sie beispielsweise eine Knotengruppe mit dem standardmäßig für Amazon EKS optimierten Amazon Linux AMI für einige Workloads und eine andere mit der GPU-Variante für Workloads erstellen, die GPU-Unterstützung erfordern.
- Wenn Ihre verwaltete Knotengruppe auf einen Fehler bei der Überprüfung des EC2 Amazon-Instance-Status stößt, gibt Amazon EKS einen Fehlercode zurück, der Ihnen bei der Diagnose des Problems hilft. Weitere Informationen finden Sie unter the section called “Fehlercodes bei verwalteten Knotengruppen”.
- Amazon EKS fügt Kubernetes-Labels zu verwalteten Knotengruppen-Instances hinzu. Diese von Amazon EKS bereitgestellten Labels sind mit dem Präfix `eks.amazonaws.com` versehen.
- Amazon EKS leert Knoten automatisch mit der Kubernetes-API während Beendigungs- oder Aktualisierungsvorgängen.
- Die Budgets für Pod-Unterbrechung werden nicht eingehalten, wenn ein Knoten mit `AZRebalance` beendet oder die Anzahl der gewünschten Knoten reduziert wird. Diese Aktionen versuchen, Pods auf dem Knoten zu entfernen. Wenn dies jedoch länger als 15 Minuten dauert, wird der Knoten beendet, unabhängig davon, ob alle Pods auf dem Knoten beendet sind. Um den Zeitraum bis zum Beenden des Knotens zu verlängern, fügen Sie der Auto-Scaling-Gruppe einen Lebenszyklus-

Hook hinzu. Weitere Informationen finden Sie unter Hinzufügen von Lifecycle-Hooks im Amazon EC2 Auto Scaling Scaling-Benutzerhandbuch.

- Damit der Draining-Prozess nach Erhalt einer Benachrichtigung über eine Spot-Unterbrechung oder eine Kapazitätsumverteilung korrekt ausgeführt werden kann, muss `CapacityRebalance` auf `true` eingestellt sein.
- Aktualisierungen verwalteter Knotengruppen respektieren die Budgets für Pod-Unterbrechung, die Sie für Ihre Pods festgelegt haben. Weitere Informationen finden Sie unter the section called “Verhaltensdetails aktualisieren”.
- Für die Verwendung verwalteter Amazon-EKS-Knotengruppen fallen keine zusätzlichen Kosten an. Sie zahlen nur für die AWS Ressourcen, die Sie bereitstellen.
- Wenn Sie Amazon-EBS-Volumes für Ihre Knoten verschlüsseln möchten, können Sie die Knoten mithilfe einer Startvorlage bereitstellen. Um verwaltete Knoten mit verschlüsselten Amazon-EBS-Volumes ohne Verwendung einer Startvorlage bereitzustellen, verschlüsseln Sie alle neuen Amazon-EBS-Volumes, die in Ihrem Konto erstellt wurden. Weitere Informationen finden Sie unter Standardverschlüsselung im EC2 Amazon-Benutzerhandbuch.

Kapazitätstypen für verwaltete Knotengruppen

Beim Erstellen einer verwalteten Knotengruppe können Sie entweder den Kapazitätstyp On-Demand oder Spot-Kapazität auswählen. Amazon EKS stellt eine verwaltete Knotengruppe mit einer Amazon EC2 Auto Scaling Scaling-Gruppe bereit, die entweder nur On-Demand-Instances oder nur Amazon EC2 Spot-Instances enthält. Sie können Pods für Fehlertoleranz-Anwendungen für verwaltete Spot-Knotengruppen und fehlerintolerante Anwendungen für On-Demand-Knotengruppen in einem einzelnen Kubernetes-Cluster planen. Standardmäßig stellt eine verwaltete Knotengruppe EC2 On-Demand-Amazon-Instances bereit.

On-Demand

Mit On-Demand-Instances zahlen Sie für Rechenkapazität bis zur zweiten Stunde ohne langfristige Verpflichtungen.

Standardmäßig: Wenn Sie keinen Capacity Type festlegen, wird die verwaltete Knotengruppe mit On-Demand-Instances bereitgestellt. Eine verwaltete Knotengruppe konfiguriert in Ihrem Namen eine Amazon EC2 Auto Scaling Scaling-Gruppe, wobei die folgenden Einstellungen angewendet werden:

- Die Allokationsstrategie zur Bereitstellung von On-Demand-Kapazität ist auf `prioritized` eingestellt. Verwaltete Knotengruppen verwenden die Reihenfolge der in der API übergebenen

Instance-Typen, um zu bestimmen, welcher Instance-Typ bei der Erfüllung der On-Demand-Kapazität zuerst verwendet werden soll. Sie können beispielsweise drei Instance-Typen in der folgenden Reihenfolge angeben: `c5.large`, `c4.large`, und `c3.large`. Wenn Ihre On-Demand-Instances gestartet werden, erfüllt die verwaltete Knotengruppe On-Demand-Kapazität beginnend mit `c5.large`, dann `c4.large`, und dann `c3.large`. Weitere Informationen finden Sie unter Amazon EC2 Auto Scaling Scaling-Gruppe im Amazon EC2 Auto Scaling Scaling-Benutzerhandbuch.

- Amazon EKS fügt allen Knoten in Ihrer verwalteten Knotengruppe die folgende Kubernetes-Bezeichnung hinzu, die den Kapazitätstyp angibt: `eks.amazonaws.com/capacityType: ON_DEMAND`. Sie können dieses Label verwenden, um statusbehaftete oder fehlerintolerante Anwendungen auf On-Demand-Knoten zu planen.

Spot-Instances

Amazon EC2 Spot-Instances sind freie EC2 Amazon-Kapazitäten, die erhebliche Rabatte gegenüber den On-Demand-Preisen bieten. Amazon EC2 Spot-Instances können mit einer zweiminütigen Unterbrechungsbenachrichtigung unterbrochen werden, wenn die Kapazität wieder EC2 benötigt wird. Weitere Informationen finden Sie unter Spot-Instances im EC2 Amazon-Benutzerhandbuch. Sie können eine verwaltete Knotengruppe mit Amazon EC2 Spot-Instances konfigurieren, um die Kosten für die Rechenknoten zu optimieren, die in Ihrem Amazon EKS-Cluster ausgeführt werden.

Um Spot-Instances in einer verwalteten Knotengruppe zu verwenden, müssen Sie eine verwaltete Knotengruppe erstellen, indem Sie den Kapazitätstyp als `spot` einstellen. Eine verwaltete Knotengruppe konfiguriert in Ihrem Namen eine Amazon EC2 Auto Scaling Scaling-Gruppe, wobei die folgenden bewährten Spot-Methoden angewendet werden:

- Um sicherzustellen, dass Ihre Spot-Knoten in den optimalen Spot-Kapazitätspools bereitgestellt werden, ist die Zuweisungsstrategie auf eine der folgenden Optionen eingestellt:
  - `price-capacity-optimized` (PCO) – Beim Erstellen neuer Knotengruppen in einem Cluster mit Kubernetes-Version `1.28` oder höher ist die Zuweisungsstrategie auf `price-capacity-optimized` eingestellt. Allerdings wird die Zuweisungsstrategie für Knotengruppen nicht geändert, die bereits mit `capacity-optimized` erstellt wurden, bevor von Amazon EKS verwaltete Knotengruppen PCO unterstützt hatten.
  - `capacity-optimized` (CO) – Beim Erstellen neuer Knotengruppen in einem Cluster mit Kubernetes-Version `1.27` oder niedriger ist die Zuweisungsstrategie auf `capacity-optimized` eingestellt.

Um die Anzahl der Spot-Kapazitätspools zu erhöhen, die für die Zuweisung von Kapazität verfügbar sind, konfigurieren Sie eine verwaltete Knotengruppe für die Verwendung mehrerer Instance-Typen.

- Amazon EC2 Spot Capacity Rebalancing ist aktiviert, sodass Amazon EKS Ihre Spot-Knoten problemlos entleeren und neu verteilen kann, um Anwendungsunterbrechungen zu minimieren, wenn ein Spot-Knoten einem erhöhten Unterbrechungsrisiko ausgesetzt ist. Weitere Informationen finden Sie unter Amazon EC2 Auto Scaling Capacity Rebalancing im Amazon EC2 Auto Scaling Scaling-Benutzerhandbuch.
  - Wenn ein Spot-Knoten eine Neuausgleichsempfehlung erhält, versucht Amazon EKS automatisch, einen neuen Ersatz-Spot-Knoten zu starten.
  - Wenn ein Spot-Unterbrechungsnachweis in zwei Minuten eintrifft, bevor sich der Ersatz-Spot-Knoten in einem`Ready`, beginnt Amazon EKS mit dem Entleeren des Spot-Knotens, der die Neuausgleichsempfehlung erhalten hat. Amazon EKS entleert den Knoten nach bestem Wissen. Daher gibt es keine Garantie dafür, dass Amazon EKS wartet, bis der Ersatzknoten dem Cluster beitritt, bevor der vorhandene Knoten entleert wird.
  - Wenn ein Ersatz-Spot-Knoten gestartet wird und in dem `Ready` Zustand auf Kubernetes ist, sperrt Amazon EKS den Spot-Knoten, der die Neuausgleichs-Empfehlung erhalten hat, und entleert ihn. Durch das Cordoning des Spot-Knotens wird sichergestellt, dass der Service-Controller keine neuen Anforderungen an diesen Spot-Knoten sendet. Es entfernt sie auch aus der Liste der gesunden, aktiven Spot-Knoten. Durch das Entleeren des Spot-Knotens wird sichergestellt, dass ausgeführte Pods reibungslos entsorgt werden.
- Amazon EKS fügt allen Knoten in Ihrer verwalteten Knotengruppe die folgende Kubernetes-Bezeichnung hinzu, die den Kapazitätstyp angibt: `eks.amazonaws.com/capacityType: SPOT`. Sie können dieses Label verwenden, um fehlertolerante Anwendungen auf Spot-Knoten zu planen.

> **Important**
>
> EC2 gibt zwei Minuten vor dem Beenden Ihrer Spot-Instance eine Benachrichtigung zur Spot-Unterbrechung aus. Pods auf Spot-Knoten erhalten jedoch möglicherweise nicht das volle 2-minütige Zeitfenster für ein ordnungsgemäßes Herunterfahren. Wenn Amazon EC2 EKS die Benachrichtigung ausgibt, gibt es eine Verzögerung, bis Amazon EKS mit der Räumung von Pods beginnt. Räumungen erfolgen sequenziell, um den Kubernetes-API-Server zu schützen. Bei mehreren gleichzeitigen Spot-Rückforderungen kann es also vorkommen, dass einige Pods verspätete Räumungsbenachrichtigungen erhalten. Pods können ohne Empfang von Kündigungssignalen gewaltsam beendet werden,

insbesondere auf Knoten mit hoher Pod-Dichte, bei gleichzeitigen Rückforderungen oder bei Verwendung langer Kündigungsfristen. Für Spot-Workloads empfehlen wir, Anwendungen unterbrechungstolerant zu gestalten, Übergangsfristen von 30 Sekunden oder weniger zu verwenden, lang andauernde PreStop-Hooks zu vermeiden und die Pod-Räumungsmetriken zu überwachen, um die tatsächlichen Kulanzfristen in Ihren Clustern zu ermitteln. Für Workloads, die eine garantierte ordentliche Kündigung erfordern, empfehlen wir, stattdessen On-Demand-Kapazität zu verwenden.

Bei der Entscheidung, ob eine Knotengruppe mit On-Demand- oder Spot-Kapazität bereitgestellt werden soll, sollten Sie die folgenden Bedingungen berücksichtigen:

- Spot-Instances eignen sich gut für zustandslose, fehlertolerante, flexible Anwendungen. Dazu gehören Workloads für Batch- und Machine-Learning-Schulungen, Big Data ETLs wie Apache Spark, Anwendungen zur Verarbeitung von Warteschlangen und statusfreie API-Endpunkte. Da es sich bei Spot um freie EC2 Amazon-Kapazität handelt, die sich im Laufe der Zeit ändern kann, empfehlen wir, Spot-Kapazität für unterbrechungstolerante Workloads zu verwenden. Insbesondere eignet sich die Spot-Kapazität für Workloads, die Zeiten tolerieren können, in denen die erforderliche Kapazität nicht verfügbar ist.
- Es wird empfohlen, On-Demand für Anwendungen zu verwenden, die fehlerintolerant sind. Dazu gehören Cluster-Verwaltungs-Tools wie Überwachungs- und Betriebstools, Bereitstellungen, die `StatefulSets` erfordern, und zustandsbehaftete Anwendungen wie Datenbanken.
- Um die Verfügbarkeit Ihrer Anwendungen bei der Verwendung von Spot-Instances zu maximieren, wird empfohlen, eine verwaltete Spot-Knotengruppe für die Verwendung mehrerer Instance-Typen zu konfigurieren. Es wird empfohlen, die folgenden Regeln anzuwenden, wenn mehrere Instance-Typen verwendet werden:
  - Wenn Sie innerhalb einer verwalteten Knotengruppe den Cluster Autoscaler verwenden, wird empfohlen, einen flexiblen Satz von Instance-Typen mit der gleichen Menge an vCPU und Arbeitsspeicherressourcen zu verwenden. Dies soll sicherstellen, dass die Knoten in Ihrem Cluster wie erwartet skalieren. Wenn Sie beispielsweise vier V CPUs - und acht GiB Arbeitsspeicher benötigen, verwenden Sie`c3.xlarge`,`c4.xlarge`,`c5.xlarge`,`c5d.xlarge`, `c5a.xlargec5n.xlarge`, oder andere ähnliche Instance-Typen.
  - Um die Anwendungsverfügbarkeit zu verbessern, empfehlen wir Ihnen, mehrere Spot-verwaltete Knotengruppen bereitzustellen. Dafür sollte jede Gruppe einen flexiblen Satz von Instance-Typen verwenden, die über die gleichen vCPU- und Speicherressourcen verfügen. Wenn Sie beispielsweise 4 V CPUs und 8 GiB Arbeitsspeicher benötigen, empfehlen wir Ihnen,

eine verwaltete Knotengruppe mit`c3.xlarge`,,,`c4.xlarge`, `c5.xlarge` `c5d.xlarge` `c5a.xlarge`c5n.xlarge, oder anderen ähnlichen Instanztypen und eine zweite verwaltete Knotengruppe mit`m3.xlarge`,,,`m4.xlarge`, `m5.xlarge` `m5d.xlarge`m5a.xlarge, `m5n.xlarge` oder anderen ähnlichen Instanztypen zu erstellen.

- Wenn Sie Ihre Knotengruppe mit dem Spot-Kapazitätstyp bereitstellen, der eine benutzerdefinierte Startvorlage verwendet, verwenden Sie die API, um mehrere Instance-Typen zu übergeben. Übergeben Sie keinen einzelnen Instance-Typ durch die Startvorlage. Weitere Informationen zum Bereitstellen einer Knotengruppe mithilfe einer Startvorlage finden Sie unterthe section called “Startvorlagen”aus.

Eine verwaltete Knotengruppe für Ihren Cluster erstellen

Dieses Thema beschreibt, wie Sie von Amazon EKS verwaltete Knotengruppen von Knoten starten können, die sich bei Ihrem Amazon-EKS-Cluster registrieren. Nachdem die Knoten dem Cluster beigetreten sind, können Sie Kubernetes-Anwendungen darin bereitstellen.

Wenn Sie zum ersten Mal eine von Amazon EKS verwaltete Knotengruppe starten, empfehlen wir Ihnen, stattdessen eine unserer Anleitungen unter Erste Schritte zu befolgen. Diese Anleitungen bieten schrittweise Anleitungen zum Erstellen eines Amazon-EKS-Clusters mit Knoten.

⚠ Important

- Amazon EKS-Knoten sind EC2 Standard-Amazon-Instances. Ihnen wird auf der Grundlage der normalen EC2 Amazon-Preise in Rechnung gestellt. Weitere Informationen finden Sie unter EC2 Amazon-Preise.
- Sie können keine verwalteten Knoten in einer AWS Region erstellen, in der AWS Outposts oder AWS Wavelength aktiviert sind. Sie könnenSelbstverwaltetes-Knoten. Weitere Informationen finden Sie unter the section called “Amazon Linux”, the section called “Windows” und the section called “Flaschenrakete”. Sie können auch eine selbstverwaltete Amazon-Linux-Knotengruppe auf einem Outpost erstellen. Weitere Informationen finden Sie unter the section called “Knoten”.
- Wenn Sie keine AMI-ID für die `bootstrap.sh`-Datei angeben, die in Amazon-EKS-optimiertem Linux oder Bottlerocket enthalten ist, erzwingen verwaltete Knotengruppen eine maximale Anzahl für den Wert von `maxPods`. Für Instances mit weniger als 30 V CPUs ist `110` die maximale Anzahl. Bei Instances mit mehr als 30 V CPUs springt die maximale Anzahl auf`250`. Diese Zahlen basieren auf Kubernetes-

Skalierbarkeitsschwellenwerten und empfohlenen Einstellungen, die nach internen Tests des Amazon-EKS-Skalierbarkeitsteams empfohlen werden. Weitere Informationen finden Sie im Blogbeitrag Amazon VPC CNI plugin increases pods per node limits (VPC-CNI-Plugin von Amazon erhöht Grenzwerte für Pods pro Knoten).

- Ein vorhandener Amazon-EKS-Cluster. Informationen zum Bereitstellen finden Sie unter the section called “Erstellen eines Clusters”.
- Eine vorhandene IAM-Rolle, die von den Knoten verwendet werden soll. Informationen zum Erstellen finden Sie unter the section called “Knoten-IAM-Rolle”. Wenn diese Rolle keine der beiden Richtlinien für die VPC CNI enthält, ist die folgende separate Rolle für die VPC-CNI-Pods erforderlich.
- (Optional, aber empfohlen) Das Amazon-VPC-CNI-Plugin für Kubernetes-Add-On, konfiguriert mit einer eigenen IAM-Rolle, der die erforderliche IAM-Richtlinie zugeordnet ist. Weitere Informationen finden Sie unter the section called “Konfiguration für IRSA”.
- Vertrautheit mit den unter Wählen Sie einen optimalen EC2 Amazon-Node-Instance-Typ aufgeführten Überlegungen. Je nachdem, welchen Instance-Typ Sie wählen, kann es zusätzliche Voraussetzungen für Ihren Cluster und Ihre VPC geben.
- Um eine von Windows verwaltete Knotengruppe hinzuzufügen, müssen Sie zunächst den Windows-Support für Ihren Cluster aktivieren. Weitere Informationen finden Sie unter the section called “Windows-Support aktivieren”.

Sie können eine verwaltete Knotengruppe mit einer der folgenden Optionen erstellen:

- the section called “`eksctl`”
- the section called “AWS-Managementkonsole”

`eksctl`

Verwaltete Knotengruppe mit eksctl erstellen

Für diesen Vorgang ist `eksctl` Version `0.215.0` oder höher erforderlich. Sie können Ihre -Version mit dem folgenden Befehl überprüfen:

```
eksctl version
```

Eine Installations- und Upgrade-Anleitung für `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.

1. (Optional) Wenn die verwaltete IAM-Richtlinie AmazonEKS_CNI_Policy Ihrer IAM-Rolle des Amazon-EKS-Knoten zugeordnet ist, empfehlen wir, sie stattdessen einer IAM-Rolle zuzuweisen, die Sie dem Kubernetes-Servicekonto `aws-node` zuordnen. Weitere Informationen finden Sie unter the section called “Konfiguration für IRSA”.
2. Erstellen Sie eine verwaltete Knotengruppe mit oder ohne Verwendung einer benutzerdefinierten Startvorlage. Das manuelle Angeben einer Startvorlage ermöglicht eine bessere Anpassung einer Knotengruppe. Zum Beispiel kann es die Bereitstellung eines benutzerdefinierten AMI oder die Bereitstellung von Argumenten für das `boostrap.sh`-Skript in einem für Amazon EKS optimierten AMI ermöglichen. Geben Sie den folgenden Befehl ein, um eine vollständige Liste aller verfügbaren Optionen und Standardwerte anzuzeigen.

   ```
   eksctl create nodegroup --help
   ```

   Ersetzen Sie im folgenden Befehl *`my-cluster`* durch den Namen Ihres Clusters und ersetzen Sie *`my-mng`* durch den Namen Ihrer Knotengruppe. Der Name der Knotengruppe darf nicht länger als 63 Zeichen sein. Er muss mit einem Buchstaben oder einer Ziffer beginnen, kann danach aber auch Bindestriche und Unterstriche enthalten.

   > **Important**
   >
   > Wenn Sie beim erstmaligen Erstellen einer verwalteten Knotengruppe keine benutzerdefinierte Startvorlage verwenden, sollten Sie später auch keine für die Knotengruppe verwenden. Wenn Sie keine benutzerdefinierte Startvorlage angegeben haben, generiert das System automatisch eine Startvorlage, die Sie nicht manuell ändern können. Das manuelle Ändern dieser automatisch generierten Startvorlage kann zu Fehlern führen.

Ohne Startvorlage

`eksctl`erstellt eine standardmäßige EC2 Amazon-Startvorlage in Ihrem Konto und stellt die Knotengruppe mithilfe einer Startvorlage bereit, die auf der Grundlage der von Ihnen angegebenen Optionen erstellt wird. Bevor Sie einen Wert für `--node-type` festlegen, siehe the section called “Amazon EC2-Instance-Typen”.

Ersetzen Sie *ami-family* durch ein zulässiges Schlüsselwort. Weitere Informationen finden Sie unter Einrichtung der Knoten-AMI-Familie in der `eksctl`-Dokumentation. *my-key*Ersetzen Sie es durch den Namen Ihres EC2 Amazon-Schlüsselpaars oder öffentlichen Schlüssels. Dieser Schlüssel wird für den SSH-Zugriff zu Ihren Knoten verwendet, nachdem diese gestartet wurden.

> **Note**
>
> Für Windows aktiviert dieser Befehl SSH nicht. Stattdessen verknüpft es Ihr EC2 Amazon-Schlüsselpaar mit der Instance und ermöglicht Ihnen, per RDP auf die Instance zuzugreifen.

Wenn Sie noch kein EC2 Amazon-Schlüsselpaar haben, können Sie eines in der erstellen AWS-Managementkonsole. Informationen zu Linux finden Sie unter EC2 Amazon-Schlüsselpaare und Linux-Instances im EC2 Amazon-Benutzerhandbuch. Informationen zu Windows finden Sie unter EC2 Amazon-Schlüsselpaare und Windows-Instances im EC2 Amazon-Benutzerhandbuch.

Wir empfehlen, den Pod-Zugriff auf das IMDS zu blockieren, wenn die folgenden Bedingungen erfüllt sind:

- Sie planen, allen Ihren Kubernetes-Servicekonten IAM-Rollen zuzuweisen, damit Pods nur die Mindestberechtigungen haben, die sie benötigen.
- Keine Pods im Cluster benötigen aus anderen Gründen Zugriff auf den Amazon EC2 Instance Metadata Service (IMDS), z. B. zum Abrufen der aktuellen AWS Region.

Weitere Informationen finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.

Wenn Sie den Pod-Zugriff auf IMDS blockieren möchten, fügen Sie die `--disable-pod-imds`-Option dem folgenden Befehl hinzu.

```
eksctl create nodegroup \
  --cluster my-cluster \
  --region region-code \
  --name my-mng \
  --node-ami-family ami-family \
  --node-type m5.large \
  --nodes 3 \
  --nodes-min 2 \
  --nodes-max 4 \
  --ssh-access \
```

```
  --ssh-public-key my-key
```

Ihre Instances können Pods optional eine deutlich höhere Anzahl von IP-Adressen zuweisen, Pods aus einem anderen CIDR-Block als der Instance IP-Adressen zuweisen und in einem Cluster ohne Internetzugang bereitgestellt werden. Weitere Informationen dazu finden Sie unter the section called “IP-Adressen erhöhen”, the section called “Benutzerdefinierte Netzwerke” und the section called “Private Cluster” für weitere Optionen, die dem vorherigen Befehl hinzugefügt werden.

Verwaltete Knotengruppen berechnen und wenden einen einzelnen Wert für die maximale Anzahl von Pods an, die auf jedem Knoten Ihrer Knotengruppe ausgeführt werden können, basierend auf dem Instance-Typ. Wenn Sie eine Knotengruppe mit unterschiedlichen Instance-Typen erstellen, wird der kleinste Wert, der für alle Instance-Typen berechnet wird, als die maximale Anzahl von Pods angewendet, die für jeden Instance-Typ in der Knotengruppe ausgeführt werden können. Managed Node Groups berechnet den Wert anhand des Skripts, auf das in der von Amazon EKS empfohlenen maximalen Pods für jeden EC2 Amazon-Instance-Typ verwiesen wird.

Mit einer Startvorlage

Die Startvorlage muss bereits vorhanden sein und die in den Grundlagen der Startvorlagenkonfiguration angegebenen Anforderungen erfüllen. Wir empfehlen, den Pod-Zugriff auf das IMDS zu blockieren, wenn die folgenden Bedingungen erfüllt sind:

- Sie planen, allen Ihren Kubernetes-Servicekonten IAM-Rollen zuzuweisen, damit Pods nur die Mindestberechtigungen haben, die sie benötigen.
- Keine Pods im Cluster benötigen aus anderen Gründen Zugriff auf den Amazon EC2 Instance Metadata Service (IMDS), z. B. zum Abrufen der aktuellen AWS Region.

Weitere Informationen finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.

Wenn Sie den Pod-Zugriff auf IMDS blockieren möchten, geben Sie die erforderlichen Einstellungen in der Startvorlage an.

a. Kopieren Sie den folgenden Inhalt auf Ihr Gerät. Ersetzen Sie die Beispielwerte und führen Sie dann den modifizierten Befehl aus, um die `eks-nodegroup.yaml` Datei zu erstellen. Mehrere Einstellungen, die Sie bei der Bereitstellung ohne Startvorlage angeben, werden in die Startvorlage verschoben. Wenn Sie keine `version` angeben, wird die Standardversion verwendet.

```
cat >eks-nodegroup.yaml <<EOF
```

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: my-cluster
  region: region-code
managedNodeGroups:
- name: my-mng
  launchTemplate:
    id: lt-id
    version: "1"
EOF
```

Eine vollständige Liste der `eksctl` Konfigurationsdateieinstellungen finden Sie unter Konfigurationsdateischema in der `eksctl` Dokumentation. Ihre Instances können Pods optional eine deutlich höhere Anzahl von IP-Adressen zuweisen, Pods IP-Adressen aus einem anderen CIDR-Block als dem der Instance zuweisen und in einem Cluster ohne ausgehenden Internetzugang bereitgestellt werden. Weitere Informationen dazu finden Sie unter the section called “IP-Adressen erhöhen”, the section called “Benutzerdefinierte Netzwerke” und the section called “Private Cluster” für zusätzliche Optionen, die der Konfigurationsdatei hinzugefügt werden.

Wenn Sie in Ihrer Startvorlage keine AMI-ID angegeben haben, berechnet „verwaltete Knotengruppen“ einen einzelnen Wert für die maximale Anzahl von Pods, die auf jedem Knoten Ihrer Knotengruppe ausgeführt werden können, basierend auf Instance-Typ. Wenn Sie eine Knotengruppe mit unterschiedlichen Instance-Typen erstellen, wird der kleinste Wert, der für alle Instance-Typen berechnet wird, als die maximale Anzahl von Pods angewendet, die für jeden Instance-Typ in der Knotengruppe ausgeführt werden können. Managed Node Groups berechnet den Wert anhand des Skripts, auf das in der von Amazon EKS empfohlenen maximalen Pods für jeden EC2 Amazon-Instance-Typ verwiesen wird.

Wenn Sie eine AMI-ID in Ihrer Startvorlage angegeben haben, geben Sie die maximale Anzahl von Pods an, die auf jedem Knoten Ihrer Knotengruppe ausgeführt werden können, wenn Sie Benutzerdefinierte Netzwerke verwenden oder die Anzahl der Ihrer Instance zugewiesenen IP-Adressen erhöhen möchten. Weitere Informationen finden Sie unter the section called “Von Amazon EKS empfohlene maximale Pods für jeden Amazon-EC2-Instance-Typ”.

b. Stellen Sie die Knotengruppe mit dem folgenden Befehl bereit.

```
eksctl create nodegroup --config-file eks-nodegroup.yaml
```

AWS-Managementkonsole

Verwaltete Knotengruppe mithilfe von AWS-Managementkonsole erstellen

1. Warten Sie, bis der Status des Clusters als `ACTIVE` angezeigt wird. Sie können keine verwaltete Knotengruppe für einen Cluster erstellen, der noch nicht `ACTIVE` ist.
2. Öffnen Sie die Amazon-EKS-Konsole.
3. Wählen Sie den Namen des Clusters aus, in dem Sie eine verwaltete Knotengruppe erstellen möchten.
4. Wählen Sie die Registerkarte Compute (Datenverarbeitung) aus.
5. Wählen Sie Add Node Group (Knotengruppe hinzufügen) aus.
6. Geben Sie auf der Seite Configure node group (Knotengruppe konfigurieren) die Parameter entsprechend aus, und wählen Sie dann Next (Weiter).
   - Name – Geben Sie einen eindeutigen Namen für die verwaltete Knotengruppe ein. Der Name der Knotengruppe darf nicht länger als 63 Zeichen sein. Er muss mit einem Buchstaben oder einer Ziffer beginnen, kann danach aber auch Bindestriche und Unterstriche enthalten.
   - Node IAM role (Knoten-IAM-Rolle) – Wählen Sie die Knoten-Instance-Rolle aus, die mit Ihrer Knotengruppe verwendet werden soll. Weitere Informationen finden Sie unter the section called “Knoten-IAM-Rolle”.

     > ⚠ Important
     >
     > - Sie können nicht dieselbe Rolle verwenden, die zum Erstellen von Clustern verwendet wurde.
     > - Es wird empfohlen, eine Rolle zu verwenden, die derzeit nicht von einer selbstverwalteten Knotengruppe genutzt wird. Andernfalls planen Sie die Verwendung mit einer neuen selbstverwalteten Knotengruppe. Weitere Informationen finden Sie unter the section called “Delete”.

   - Startvorlage verwenden – (Optional) Wählen Sie aus, ob Sie eine bestehende Startvorlage verwenden möchten. Wählen Sie einen Namen für die Startvorlage aus. Wählen Sie dann eine Launch template version (Startvorlagenversion) aus. Wenn Sie keine Version auswählen, verwendet Amazon EKS die Standardversion der Vorlage. Startvorlagen ermöglichen eine stärkere Anpassung Ihrer Knotengruppe, z. B. die Bereitstellung eines benutzerdefinierten AMI, die Zuordnung einer deutlich höheren Anzahl von IP-Adressen zu Pods, die Zuordnung von IP-

Adressen zu Pods aus einem anderen CIDR-Block als dem der Instance und die Bereitstellung von Knoten in einem Cluster ohne ausgehenden Internetzugang. Weitere Informationen finden Sie unter the section called “IP-Adressen erhöhen”, the section called “Benutzerdefinierte Netzwerke” und the section called “Private Cluster”.

Die Startvorlage muss die Anforderungen unter Verwaltete Knoten mit Startvorlagen anpassen erfüllen Wenn Sie keine eigene Startvorlage verwenden, erstellt die Amazon EKS-API eine standardmäßige EC2 Amazon-Startvorlage in Ihrem Konto und stellt die Knotengruppe mithilfe der Standard-Startvorlage bereit.

Wenn Sie IAM-Rollen für Dienstkonten implementieren, jedem Pod, der Zugriff auf AWS Dienste benötigt, die erforderlichen Berechtigungen direkt zuweisen und keine Pods in Ihrem Cluster aus anderen Gründen Zugriff auf IMDS benötigen, z. B. zum Abrufen der aktuellen AWS Region, können Sie den Zugriff auf IMDS auch für Pods deaktivieren, die kein Host-Netzwerk in einer Startvorlage verwenden. Weitere Informationen finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.

- Kubernetes labels – (Optional) Sie können Kubernetes-Labels auf die Knoten in Ihrer verwalteten Knotengruppe anwenden.
- Kubernetes-Taints – (Optional) Sie können Kubernetes-Taints auf die Knoten in Ihrer verwalteten Knotengruppe anwenden. Die verfügbaren Optionen im Menü Effect (Effekt) sind **`NoSchedule`** , **`NoExecute`** und **`PreferNoSchedule`** . Weitere Informationen finden Sie unter the section called “Pods von der Planung auf bestimmten Knoten ausschließen”.
- Tags – (Optional) Sie können wählen, ob Sie Ihre mit Amazon EKS verwaltete Knotengruppe taggen möchten. Diese Tags werden nicht an andere Ressourcen in der Knotengruppe, z. B. Auto-Scaling-Gruppen oder Instances, bekannt gegeben. Weitere Informationen finden Sie unter the section called “Markieren Ihrer -Ressourcen”.

7. Geben Sie auf der Seite Set compute configuration (Datenverarbeitung-Konfiguration einrichten) die Parameter entsprechend ein, und wählen Sie dann Next (Weiter).
   - AMI-Typ – Wählen Sie einen AMI-Typ aus. Wenn Sie Arm-Instances bereitstellen, sollten Sie AMIs vor der Bereitstellung unbedingt die Überlegungen unter Amazon EKS-optimiertes Arm Amazon Linux lesen.

     Wenn Sie auf der vorherigen Seite eine Startvorlage angegeben haben und in der Startvorlage ein AMI angegeben haben, können Sie keinen Wert auswählen. Der Wert aus der Vorlage wird angezeigt. Das in der Vorlage angegebene AMI muss die Anforderungen unter Angabe eines AMI erfüllen.

- Capacity type (Kapazitätstyp) — Wählen Sie einen Kapazitätstyp aus. Weitere Informationen zur Auswahl eines Kapazitätstyps finden Sie unter the section called “Kapazitätstypen für verwaltete Knotengruppen”. Es ist nicht möglich, verschiedene Kapazitätstypen innerhalb derselben Knotengruppe zu mischen. Wenn Sie beide Kapazitätstypen verwenden möchten, erstellen Sie separate Knotengruppen mit jeweils eigenen Kapazitäts- und Instance-Typen. Informationen GPUs zur Bereitstellung und Skalierung von GPU-beschleunigten Worker-Knoten finden Sie unter Für verwaltete Knotengruppen reservieren.
- Instance-Typen- Standardmäßig wird ein oder mehrere Instance-Typen angegeben. Um einen Standard-Instance-Typ zu entfernen, wählen Sie X auf der rechten Seite des Instance-Typs. Wählen Sie den Instance-Typ aus, der in der verwalteten Knotengruppe verwendet werden soll. Weitere Informationen finden Sie unter the section called “Amazon EC2-Instance-Typen”.

  Die Konsole zeigt eine Reihe von häufig verwendeten Instance-Typen an. Wenn Sie eine verwaltete Knotengruppe mit einem Instanztyp erstellen müssen, der nicht angezeigt wird`eksctl`, verwenden Sie die AWS CLI oder ein SDK AWS CloudFormation, um die Knotengruppe zu erstellen. Wenn Sie auf der vorherigen Seite eine Startvorlage angegeben haben, können Sie keinen Wert auswählen, da der Instance-Typ in der Startvorlage angegeben werden muss. Der Wert aus der Startvorlage wird angezeigt. Wenn Sie die OptionSpot-InstancesfürCapacity type (Kapazitätstyp)Wir empfehlen Ihnen, mehrere Instance-Typen anzugeben, um die Verfügbarkeit zu verbessern.
- Datenträgergröße – Geben Sie die Datenträgergröße (in GiB) ein, die für das Root-Volume des Knotens verwendet werden soll.

  Wenn Sie auf der vorherigen Seite eine Startvorlage angegeben haben, können Sie keinen Wert auswählen, da dieser in der Startvorlage angegeben werden muss.
- Desired size (Gewünschte Größe) – Geben Sie die aktuelle Anzahl von Knoten an, die die verwaltete Knotengruppe beim Start beibehalten soll.

  > **Note**
  >
  > Amazon EKS skaliert Ihre Knotengruppe nicht automatisch nach oben oder unten. Sie können jedoch den Kubernetes Cluster Autoscaler so konfigurieren, dass er dies für Sie übernimmt. Weitere Informationen finden Sie unter Cluster Autoscaler on. AWS
- Minimum size (Mindestgröße) – Geben Sie die Mindestanzahl von Knoten an, auf die die verwaltete Knotengruppe skaliert werden kann.

- Maximum size (Maximale Größe) – Geben Sie die maximale Anzahl von Knoten an, auf die die verwaltete Knotengruppe skaliert werden kann.
- Konfiguration der Knotengruppe aktualisieren— (Optional) Sie können die Anzahl oder den Prozentsatz der Knoten auswählen, die parallel aktualisiert werden sollen. Diese Knoten werden während der Aktualisierung nicht verfügbar sein. FürMaximal nicht verfügbarWählen Sie eine der folgenden Optionen und geben Sie einenValue:
  - Zahl – Wählen und geben Sie die Anzahl der Knoten in Ihrer Knotengruppe an, die parallel aktualisiert werden können.
  - Prozentsatz – Wählen und geben Sie den Prozentsatz der Knoten in der Knotengruppe an, die parallel aktualisiert werden können. Dies ist nützlich, wenn Sie eine große Anzahl von Knoten in Ihrer Knotengruppe haben.
- Konfiguration der automatischen Knotenreparatur – (Optional) Wenn Sie das Kontrollkästchen Automatische Knotenreparatur aktivieren aktivieren, ersetzt Amazon EKS automatisch Knoten, wenn erkannte Probleme auftreten. Weitere Informationen finden Sie unter the section called “Knotenintegrität”.

8. Füllen Sie auf der Seite Specify Details (Details angeben) die Parameter entsprechend aus, und klicken Sie dann auf Next.
   - Subnets (Subnetze) – Wählen Sie die Subnetze aus, in die die verwalteten Knoten gestartet werden sollen.

> **Important**
>
> Wenn Sie eine zustandsbehaftete Anwendung über mehrere Availability Zones hinweg ausführen, die von Amazon-EBS-Volumes gesichert wird und den Kubernetes Cluster Autoscaler verwendet, sollten Sie mehrere Knotengruppen konfigurieren, die jeweils für eine einzelne Availability Zone gelten. Außerdem sollten Sie das `--balance-similar-node-groups`-Feature aktivieren.

> **Important**
>
> - Wenn Sie ein öffentliches Subnetz wählen und in Ihrem Cluster nur der öffentliche API-Server-Endpunkt aktiviert ist, muss das Subnetz `MapPublicIPOnLaunch` auf `true` eingestellt sein, damit die Instances erfolgreich einem Cluster zugeordnet werden können. Wenn das Subnetz am `eksctl` oder nach dem 26. März 2020

mithilfe der von Amazon EKS bereitgestellten AWS CloudFormation Vorlagen erstellt wurde, ist diese Einstellung bereits auf gesetzt. `true` Wenn die Subnetze mit `eksctl` oder den AWS CloudFormation Vorlagen vor dem 26. März 2020 erstellt wurden, müssen Sie die Einstellung manuell ändern. Weitere Informationen finden Sie unter Ändern des Attributs für die öffentliche IPv4 Adressierung für Ihr Subnetz.

- Wenn Sie eine Startvorlage verwenden und mehrere Netzwerkschnittstellen angeben, weist Amazon EC2 nicht automatisch eine öffentliche `IPv4` Adresse zu, auch wenn sie auf `true` eingestellt `MapPublicIpOnLaunch` ist. Damit Knoten dem Cluster in diesem Szenario beitreten können, müssen Sie entweder den privaten API-Serverendpunkt des Clusters aktivieren oder Knoten in einem privaten Subnetz mit ausgehendem Internetzugriff starten, der über eine alternative Methode wie ein NAT-Gateway bereitgestellt wird. Weitere Informationen finden Sie unter EC2 Amazon-Instance-IP-Adressierung im EC2 Amazon-Benutzerhandbuch.

- Konfigurieren des SSH-Zugriffs auf Knoten (Optional). Mit der Aktivierung von SSH können Sie eine Verbindung zu Ihren Instances herstellen und Diagnoseinformationen erfassen, wenn Probleme auftreten. Wir empfehlen dringend, den Fernzugriff zu aktivieren, sobald Sie eine Knotengruppe erstellen. Sie können den Fernzugriff nicht mehr aktivieren, nachdem die Knotengruppe erstellt wurde.

  Wenn Sie eine Startvorlage verwenden möchten, wird diese Option nicht angezeigt. Um den Remotezugriff auf Ihre Knoten zu aktivieren, geben Sie in der Startvorlage ein Schlüsselpaar an und stellen Sie sicher, dass die richtige Schnittstelle für die Knoten in den Sicherheitsgruppen geöffnet ist, die Sie in der Startvorlage angeben. Weitere Informationen finden Sie unter the section called “Benutzerdefinierte Sicherheitsgruppen”.

  > **Note**
  >
  > Für Windows aktiviert dieser Befehl SSH nicht. Stattdessen verknüpft es Ihr EC2 Amazon-Schlüsselpaar mit der Instance und ermöglicht Ihnen, per RDP auf die Instance zuzugreifen.

- Wählen Sie für das SSH-Schlüsselpaar (optional) einen Amazon EC2 SSH-Schlüssel aus, den Sie verwenden möchten. Informationen zu Linux finden Sie unter EC2 Amazon-Schlüsselpaare und Linux-Instances im EC2 Amazon-Benutzerhandbuch. Informationen zu Windows finden Sie unter EC2 Amazon-Schlüsselpaare und Windows-Instances im EC2 Amazon-Benutzerhandbuch. Wenn Sie eine Startvorlage verwenden möchten, können Sie

keine auswählen. Wenn ein Amazon EC2 SSH-Schlüssel für Knotengruppen bereitgestellt wird, die Bottlerocket verwenden AMIs, ist der administrative Container ebenfalls aktiviert. Weitere Informationen finden Sie unter Administrator-Container auf GitHub.

- Für Zulassen des SSH-Remote-Zugriffs von, wenn Sie den Zugriff auf bestimmte Instances beschränken möchten, wählen Sie die Sicherheitsgruppen aus, die diesen Instances zugeordnet sind. Wenn Sie keine bestimmten Sicherheitsgruppen auswählen, ist SSH-Zugriff von überall im Internet erlaubt (`0.0.0.0/0`).

9. Überprüfen Sie auf der Seite Review and create (Überprüfen und Erstellen) die Konfiguration der verwalteten Knoten, und wählen Sie Create (Erstellen).

   Sollten Knoten nicht in den Cluster aufgenommen werden können, finden Sie weitere Informationen unter the section called “Knoten können nicht mit dem Cluster verknüpft werden” im Kapitel zur Fehlerbehebung.

10. Sehen Sie sich den Status Ihrer Knoten an und warten Sie, bis diese in den `Ready`-Status eintreten.

```
kubectl get nodes --watch
```

11. (Nur GPU-Knoten) Wenn Sie einen GPU-Instance-Typ und ein für Amazon EKS optimiertes beschleunigtes AMI ausgewählt haben, müssen Sie das NVIDIA-Geräte-Plug-In für Kubernetes als DaemonSet auf Ihrem Cluster anwenden. Ersetzen Sie es *`vX.X.X`* durch die gewünschte s-device-pluginNVIDIA/K8-Version, bevor Sie den folgenden Befehl ausführen.

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/
deployments/static/nvidia-device-plugin.yml
```

Kubernetes-Add-Ons installieren

Nachdem Sie nun über einen funktionierenden Amazon-EKS-Cluster mit Worker-Knoten verfügen, können Sie mit der Installation von Kubernetes-Add-Ons und -Anwendungen auf Ihrem Cluster beginnen. Die folgenden Dokumentationsthemen helfen Ihnen bei der Erweiterung der Funktionalität Ihres Clusters.

- Der IAM-Prinzipal, der den Cluster erstellt hat, ist der einzige Prinzipal, der Aufrufe an den Kubernetes-API-Server mit `kubectl` oder AWS-Managementkonsole tätigen kann. Wenn Sie möchten, dass andere IAM-Prinzipale Zugriff auf Ihren Cluster haben, müssen Sie sie hinzufügen.

Weitere Informationen erhalten Sie unter the section called “Kubernetes-API-Zugriff” und the section called “Erforderliche Berechtigungen”.

- Wir empfehlen, den Pod-Zugriff auf das IMDS zu blockieren, wenn die folgenden Bedingungen erfüllt sind:
  - Sie planen, allen Ihren Kubernetes-Servicekonten IAM-Rollen zuzuweisen, damit Pods nur die Mindestberechtigungen haben, die sie benötigen.
  - Keine Pods im Cluster benötigen aus anderen Gründen Zugriff auf den Amazon EC2 Instance Metadata Service (IMDS), z. B. zum Abrufen der aktuellen AWS Region.

  Weitere Informationen finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.
- Konfigurieren Sie den Kubernetes Cluster Autoscaler, um die Anzahl der Knoten in Ihren Knotengruppen automatisch anzupassen.
- Stellen Sie eine Beispielanwendung für Ihr Cluster bereit.
- Organisieren und überwachen Sie Cluster-Ressourcen mit wichtigen Tools für die Verwaltung Ihres Clusters.

Eine verwaltete Knotengruppe für Ihren Cluster aktualisieren

Wenn Sie eine Aktualisierung der verwalteten Knotengruppe initiieren, aktualisiert Amazon EKS Ihre Knoten automatisch für Sie. Führen Sie die Schritte aus, die unter Jede Phase der Knotenaktualisierungen verstehen aufgelistet sind. Wenn Sie ein Amazon-EKS-optimiertes AMI verwenden, wendet Amazon EKS automatisch die neuesten Sicherheits-Patches und Betriebssystem-Aktualisierungen auf Ihre Knoten als Teil der neuesten AMI-Versionsversion an.

Es gibt mehrere Szenarien, in denen Sie die Version oder Konfiguration Ihrer verwalteten Amazon-EKS-Knotengruppe aktualisieren:

- Sie haben die Kubernetes-Version für Ihren Amazon-EKS-Cluster aktualisiert und möchten Ihre Arbeitsknoten so aktualisieren, dass sie dieselbe Kubernetes-Version verwenden.
- Eine neue AMI-Version ist für Ihre verwaltete Knotengruppe verfügbar. Weitere Informationen zu AMI-Versionen finden Sie unter diesen Abschnitten:
  - the section called “Versionsinformationen abrufen”
  - the section called “Bottlerocket”
  - the section called “Versionsinformationen abrufen”

- Sie möchten die minimale, maximale oder gewünschte Anzahl der Instances in Ihrer verwalteten Knotengruppe anpassen.
- Sie möchten Kubernetes-Labels hinzufügen oder aus den Instances in Ihrer verwalteten Knotengruppe entfernen.
- Sie möchten AWS-Tags Ihrer verwalteten Knotengruppe hinzufügen oder daraus entfernen.
- Sie müssen eine neue Version einer Startvorlage mit Konfigurationsänderungen bereitstellen, z. B. ein aktualisiertes benutzerdefiniertes AMI.
- Sie haben Version `1.9.0` oder höher des Amazon-VPC-CNI-Add-Ons bereitgestellt, das Add-On für die Präfix-Delegierung aktiviert und möchten neue AWS-Nitro-System-Instances in einer Knotengruppe, um eine deutlich erhöhte Anzahl von Pods zu unterstützen. Weitere Informationen finden Sie unter the section called “IP-Adressen erhöhen”.
- Sie haben die IP-Präfix-Delegierung für Windows-Knoten aktiviert und möchten, dass neue AWS-Nitro-System-Instances in einer Knotengruppe eine deutlich erhöhte Anzahl von Pods unterstützen. Weitere Informationen finden Sie unter the section called “IP-Adressen erhöhen”.

Wenn es eine neuere AMI-Version für die Kubernetes-Version Ihrer verwalteten Knotengruppe gibt als die, welche Ihre Knotengruppe derzeit ausführt, können Sie sie aktualisieren, um diese neue AMI-Version zu verwenden. Wenn Ihr Cluster eine neuere Kubernetes-Version als Ihre Knotengruppe ausführt, können Sie die Knotengruppe so aktualisieren, dass sie die neueste AMI-Version verwendet, welche der Kubernetes-Version Ihres Clusters entspricht.

Wenn ein Knoten in einer verwalteten Knotengruppe aufgrund eines Skalierungsvorgangs oder Aktualisierung beendet wird, werden die Pods in diesem Knoten zuerst geleert. Weitere Informationen finden Sie unter the section called “Verhaltensdetails aktualisieren”.

Aktualisieren einer Knotengruppenversion

Sie können eine Knotengruppen-Version mit einer der folgenden Methoden aktualisieren:

- the section called “`eksctl`”
- the section called “AWS-Managementkonsole”

Die Version, auf die Sie aktualisieren, darf nicht höher sein als die Version der Steuerebene.

`eksctl`

Aktualisierung einer verwalteten Knotengruppe mithilfe von **`eksctl`**

Aktualisieren Sie eine verwaltete Knotengruppe mit dem folgenden Befehl auf die neueste AMI-Version derselben Kubernetes-Version, die derzeit auf den Knoten bereitgestellt wird. Ersetzen Sie jeden *Beispielwert* durch Ihre eigenen Werte.

```
eksctl upgrade nodegroup \
  --name=node-group-name \
  --cluster=my-cluster \
  --region=region-code
```

> **Note**
>
> Wenn Sie eine Knotengruppe, die mit einer Startvorlage bereitgestellt wurde, auf eine neue Version der Startvorlage aktualisieren, fügen Sie `--launch-template-version` *version-number* dem vorangehenden Befehl hinzu. Die Startvorlage muss die unter Anpassen verwalteter Knoten mit Startvorlagen beschriebenen Anforderungen erfüllen. Wenn die Startvorlage eine benutzerdefinierte AMI enthält, muss die AMI die Anforderungen unter Angabe einer AMI erfüllen. Wenn Sie Ihre Knotengruppe auf eine neuere Version Ihrer Startvorlage aktualisieren, wird jeder Knoten wiederverwertet, um mit der neuen Konfiguration der angegebenen Startvorlagenversion übereinzustimmen.

Sie können eine Knotengruppe, die ohne eine Startvorlage bereitgestellt wird, nicht direkt auf eine neue Startvorlagenversion aktualisieren. Stattdessen müssen Sie mithilfe der Startvorlage eine neue Knotengruppe bereitstellen, um die Knotengruppe auf eine neue Version der Startvorlage zu aktualisieren.

Sie können eine Knotengruppe auf dieselbe Version aktualisieren wie die Kubernetes-Version der Steuerebene. Wenn Sie beispielsweise über einen Cluster mit Kubernetes `1.33` verfügen, können Sie Knoten, auf denen derzeit Kubernetes `1.32` ausgeführt wird, mit dem folgenden Befehl auf Version `1.33` aktualisieren.

```
eksctl upgrade nodegroup \
  --name=node-group-name \
  --cluster=my-cluster \
  --region=region-code \
  --kubernetes-version=1.33
```

AWS-Managementkonsole

Aktualisierung einer verwalteten Knotengruppe mithilfe von AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie den Cluster aus, der die zu aktualisierende Knotengruppe enthält.
3. Wenn mindestens eine Knotengruppe über ein verfügbares Update verfügt, wird oben auf der Seite ein Feld angezeigt, in dem Sie über das verfügbare Update informiert werden. Wenn Sie die Registerkarte Datenverarbeitung wählen, finden Sie den Eintrag Jetzt aktualisieren in der Spalte AMI-Vorversion in der Tabelle Knotengruppen für die Knotengruppe, für die eine Aktualisierung verfügbar ist. Um die Knotengruppe zu aktualisieren, wählen Sie Update now (Jetzt aktualisieren).

   Es wird keine Benachrichtigung für Knotengruppen angezeigt, die mit einem benutzerdefinierten AMI bereitgestellt wurden. Wenn Ihre Knoten mit einem benutzerdefinierten AMI bereitgestellt werden, führen Sie die folgenden Schritte aus, um ein neues aktualisiertes benutzerdefiniertes AMI bereitzustellen.

   a. Erstellen Sie eine neue Version Ihres AMI.

   b. Erstellen Sie eine neue Version der Startvorlage mit der neuen AMI-ID.

   c. Aktualisieren Sie die Knoten auf die neue Version der Startvorlage.
4. Aktivieren oder deaktivieren Sie im Dialogfeld Update node group version (Knotengruppenversion aktualisieren) die folgenden Optionen:
   - Update node group version (Knotengruppenversion aktualisieren) – Diese Option ist nicht verfügbar, wenn Sie ein benutzerdefiniertes AMI bereitgestellt haben oder Ihr Amazon EKS-optimiertes AMI derzeit über die neueste Version für Ihren Cluster verfügt.
   - Change launch template version (Startvorlagenversion ändern) – Diese Option ist nicht verfügbar, wenn die Knotengruppe ohne eine benutzerdefinierte Startvorlage bereitgestellt wird. Sie können die Version der Startvorlage nur für eine Knotengruppe aktualisieren, die mit einer benutzerdefinierten Startvorlage bereitgestellt wurde. Wählen Sie die Launch template version (Startvorlagenversion) aus, auf die Sie die Knotengruppe aktualisieren möchten. Wenn Ihre Knotengruppe mit einem benutzerdefinierten AMI konfiguriert ist, muss die ausgewählte Version auch ein AMI angeben. Wenn Sie ein Upgrade auf eine neuere Version Ihrer Startvorlage durchführen, wird jeder Knoten wiederverwertet, damit er der neuen Konfiguration der angegebenen Startvorlagenversion entspricht.
5. Wählen Sie für Update strategy (Aktualisierungsstrategie) eine der folgenden Optionen aus:

- Fortlaufende Aktualisierung – Diese Option berücksichtigt die Budgets für die Pod-Unterbrechung Ihres Clusters. Aktualisierungen schlagen fehl, wenn ein Problem mit einem Budget für Pod-Unterbrechung auftritt, das dazu führt, dass Amazon EKS die Pods, die auf dieser Knotengruppe ausgeführt werden, nicht ordnungsgemäß entleert werden kann.
- Aktualisierung erzwingen – Bei dieser Option werden Budgets für Pod-Unterbrechungen nicht berücksichtigt. Aktualisierungen erfolgen unabhängig von Problemen mit Budgets für Pod-Unterbrechung, indem Knoten-Neustarts erzwungen werden.

6. Wählen Sie Aktualisieren.

Bearbeiten einer Knotengruppenkonfiguration

Sie können einige Konfigurationen einer verwalteten Knotengruppe ändern.

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie den Cluster aus, der die zu bearbeitende Knotengruppe enthält.
3. Wählen Sie die Registerkarte Compute (Datenverarbeitung) aus.
4. Wählen Sie die zu bearbeitende Knotengruppe aus und wählen Sie dann Edit (Bearbeiten).
5. (Optional) Gehen Sie auf der Seite Edit node group (Knotengruppe bearbeiten) wie folgt vor:
   a. Bearbeiten Sie die Node group scaling configuration (Skalierungskonfiguration der Knotengruppe).
      - Desired size (Gewünschte Größe) – Geben Sie die aktuelle Anzahl von Knoten an, die die verwaltete Knotengruppe beibehalten soll.
      - Minimum size (Mindestgröße) – Geben Sie die Mindestanzahl von Knoten an, auf die die verwaltete Knotengruppe skaliert werden kann.
      - Maximum size (Maximale Größe) – Geben Sie die maximale Anzahl von Knoten an, auf die die verwaltete Knotengruppe skaliert werden kann. Die maximale Anzahl von Knoten, die in einer Knotengruppe unterstützt werden, finden Sie unter the section called “Servicekontingente”.
   b. (Optional) Fügen Sie den Knoten in Ihrer Knotengruppe Kubernetes-Labels hinzu oder entfernen Sie sie. Die hier gezeigten Labels sind nur die Labels, die Sie mit Amazon EKS angewendet haben. Andere Labels können auf Ihren Knoten vorhanden sein, die hier nicht angezeigt werden.
   c. (Optional) Fügen Sie den Knoten in Ihrer Knotengruppe Kubernetes-Taints hinzu oder entfernen Sie sie. Hinzugefügte Färbungen können die Wirkung von **`NoSchedule`**, **`NoExecute`**, oder

`PreferNoSchedule` haben. Weitere Informationen finden Sie unter the section called “Pods von der Planung auf bestimmten Knoten ausschließen”.

d. (Optional) Fügen Sie Tags zu Ihrer Knotengruppenressource hinzu oder entfernen Sie sie. Diese Markierungen werden nur auf die Amazon-EKS-Knotengruppe angewendet. Knotengruppen-Tags werden nicht auf andere Ressourcen übertragen, die der Knotengruppe zugeordnet sind, z. B. die Amazon-EC2-Instances oder Subnetze.

e. (Optional) Bearbeiten Sie Konfiguration der Knotengruppe aktualisieren. Wählen Sie entweder Zahl oder Prozentanteil aus.

- Zahl – Wählen und geben Sie die Anzahl der Knoten in Ihrer Knotengruppe an, die parallel aktualisiert werden können. Diese Knoten sind während der Aktualisierung nicht verfügbar.
- Prozentsatz – Wählen und geben Sie den Prozentsatz der Knoten in der Knotengruppe an, die parallel aktualisiert werden können. Diese Knoten sind während der Aktualisierung nicht verfügbar. Dies ist nützlich, wenn Sie viele Knoten in Ihrer Knotengruppe haben.

f. Wenn Sie mit der Bearbeitung fertig sind, wählen Sie Änderungen speichern aus.

> **Important**
>
> Bei der Aktualisierung der Knotengruppen-Konfiguration werden durch die Änderung von `NodegroupScalingConfig` die Budgets für Pod-Unterbrechung (PDBs) nicht berücksichtigt. Im Gegensatz zum Aktualisierungsprozess für Knotengruppen (bei dem Knoten während der Aktualisierungsphase entleert und PDBs berücksichtigt werden) führt die Aktualisierung der Skalierungskonfiguration dazu, dass Knoten sofort über einen Aufruf zur Verkleinerung der Auto-Scaling-Gruppe (ASG) beendet werden. Dies geschieht ohne Berücksichtigung von PDBs, unabhängig von der Zielgröße, auf die Sie skalieren. Das bedeutet, dass bei einer Reduzierung von `desiredSize` einer von Amazon EKS verwalteten Knotengruppe die Pods sofort nach Beendigung der Knoten entfernt werden, ohne dass PDBs berücksichtigt werden.

Alle Phasen der Knotenaktualisierung verstehen

Die Upgrade-Strategie für verwaltete Amazon-EKS-Worker-Knoten umfasst vier verschiedene Phasen, die in den folgenden Abschnitten beschrieben werden.

Einrichtungsphase

Die Einrichtungsphase umfasst folgende Schritte:

1. Erstellt eine neue Amazon-EC2-Startvorlage für die Auto-Scaling-Gruppe, die Ihrer Knotengruppe zugeordnet ist. Die neue Version der Startvorlage verwendet das Ziel-AMI oder die vom Kunden bereitgestellte Startvorlagenversion für die Aktualisierung.
2. Die Auto-Scaling-Gruppe wird so aktualisiert, dass sie die neueste Startvorlage verwendet.
3. Es bestimmt die maximale Anzahl der parallel zu aktualisierenden Knoten mithilfe der `updateConfig`-Eigenschaft für die Knotengruppe. Der maximal nicht verfügbare Wert hat ein Kontingent von 100 Knoten. Der Standardwert beträgt einen Knoten. Weitere Informationen dazu finden Sie unter der Eigenschaft updateConfig in der Amazon-EKS-API-Referenz.

Aufskalierungsphase

Beim Upgrade der Knoten in einer verwalteten Knotengruppe werden die aktualisierten Knoten in derselben Availability Zone gestartet wie diejenigen, die aktualisiert werden. Um diese Platzierung zu garantieren, verwenden wir das Availability Zone Rebalancing von Amazon EC2. Weitere Informationen dazu finden Sie unter Availability-Zone-Rebalancing im Amazon-EC2-Auto-Scaling-Benutzerhandbuch. Um diese Anforderung zu erfüllen, ist es möglich, dass wir bis zu zwei Instances pro Availability Zone in Ihrer verwalteten Knotengruppe starten.

Die Aufskalierungsphase umfasst folgende Schritte:

1. Es erhöht die maximale Größe und die gewünschte Größe der Auto-Scaling-Gruppe um den jeweils größeren Wert:
   - Bis zu doppelt so viele Availability Zones, in denen die Auto-Scaling-Gruppe bereitgestellt wird.
   - Die maximale Nichtverfügbarkeit der Aktualisierung.

     Wenn Ihre Knotengruppe beispielsweise über fünf Availability Zones und `maxUnavailable` als eine verfügt, kann der Upgrade-Prozess maximal zehn Knoten starten. Wenn jedoch `maxUnavailable` 20 ist (oder etwas größer als 10, würde der Prozess 20 neue Knoten starten).
2. Nach dem Skalieren der Auto-Scaling-Gruppe wird überprüft, ob die Knoten, die die neueste Konfiguration verwenden, in der Knotengruppe vorhanden sind. Dieser Schritt ist nur erfolgreich, wenn er diese Kriterien erfüllt:
   - Mindestens ein neuer Knoten wird in jeder Availability Zone gestartet, in der der Knoten existiert.

- Jeder neue Knoten sollte im `Ready`-Zustand sein.
- Neue Knoten sollten Amazon-EKS-Labels angewandt haben.

  Dies sind die von Amazon EKS auf die Worker-Knoten in einer regulären Knotengruppe angewandten Labels:

  - `eks.amazonaws.com/nodegroup-image=$amiName`
  - `eks.amazonaws.com/nodegroup=$nodeGroupName`

  Dies sind die von Amazon EKS angewendeten Labels auf den Worker-Knoten in einer benutzerdefinierten Startvorlage oder AMI-Knotengruppe:

  - `eks.amazonaws.com/nodegroup-image=$amiName`
  - `eks.amazonaws.com/nodegroup=$nodeGroupName`
  - `eks.amazonaws.com/sourceLaunchTemplateId=$launchTemplateId`
  - `eks.amazonaws.com/sourceLaunchTemplateVersion=$launchTemplateVersion`

3. Es markiert Knoten als nicht planbar, um die Planung neuer Pods zu vermeiden. Es kennzeichnet Knoten auch mit `node.kubernetes.io/exclude-from-external-load-balancers=true`, um die Knoten aus den Load Balancern zu entfernen, bevor die Knoten beendet werden.

Die folgenden sind bekannte Gründe, die zu einem `NodeCreationFailure`-Fehler in dieser Phase führen:

Nicht genügend Kapazität in der Availability Zone

Es besteht die Möglichkeit, dass die Availability Zone nicht über die Kapazität der angeforderten Instance-Typen verfügt. Es wird empfohlen, beim Erstellen einer verwalteten Knotengruppe mehrere Instance-Typen zu konfigurieren.

EC2-Instance-Limits in Ihrem Konto

Möglicherweise müssen Sie die Anzahl der Amazon-EC2-Instances erhöhen, die Ihr Konto gleichzeitig mit Service Quotas ausführen kann. Weitere Informationen dazu finden Sie unter EC2-Service-Quotas im Amazon-Elastic-Compute-Cloud-Benutzerhandbuch für Linux-Instances.

Anwenderbezogene Benutzerdaten

Anwenderbezogene Benutzerdaten können manchmal den Bootstrap-Prozess stören. Dieses Szenario kann dazu führen, dass `kubelet` nicht auf dem Knoten startet oder Knoten nicht die

erwarteten Amazon-EKS-Labels auf ihnen erhalten. Weitere Informationen finden Sie unter the section called “Angeben eines AMI”.

Alle Änderungen, die dazu führen, dass ein Knoten fehlerhaft ist oder nicht bereit ist

Knotenfestplattendruck, Speicherdruck und ähnliche Bedingungen können dazu führen, dass ein Knoten nicht in den `Ready`-Zustand wechselt.

Jeder Knoten wird in der Regel innerhalb von 15 Minuten gebootet

Wenn ein Knoten länger als 15 Minuten zum Bootstrapping und Beitritt zum Cluster benötigt, kommt es zu einer Zeitüberschreitung des Upgrades. Dies ist die Gesamtlaufzeit für das Booten eines neuen Knotens, gemessen von der Zeit, in welcher der neue Knoten benötigt wird, bis zu seinem Beitritt zum Cluster. Bei der Aktualisierung einer verwalteten Knotengruppe beginnt der Zeitzähler zu laufen, sobald die Größe der Auto-Scaling-Gruppe zunimmt.

Aktualisierungs-Phase

Die Aktualisierungsphase verläuft je nach Aktualisierungsstrategie auf zwei unterschiedliche Arten. Es gibt zwei Aktualisierungsstrategien: Standard und Minimal.

Wir empfehlen in den meisten Fällen die Standard-Strategie. Diese erstellt neue Knoten, bevor die alten beendet werden, sodass die verfügbare Kapazität während der Aktualisierungsphase erhalten bleibt. Diese Minimal-Strategie ist in Szenarien sinnvoll, in denen Sie hinsichtlich Ressourcen oder Kosten eingeschränkt sind, beispielsweise bei Hardware-Beschleunigern wie GPUs. Es werden die alten Knoten beendet, bevor neue erstellt werden, sodass die Gesamtkapazität niemals über die von Ihnen konfigurierte Menge hinausgeht.

Die Standard-Aktualisierungsstrategie umfasst folgende Schritte:

1. Die Anzahl der Knoten (gewünschte Anzahl) in der Auto-Scaling-Gruppe wird erhöht, wodurch die Knotengruppe zusätzliche Knoten erstellt.
2. Es wählt nach dem Zufallsprinzip einen Knoten aus, der aktualisiert werden muss, bis zum Maximum der für die Knotengruppe nicht verfügbar ist.
3. Es entleert die Pods vom Knoten. Wenn die Pods den Knoten nicht innerhalb von 15 Minuten verlassen und es kein Erzwingungs-Flag gibt, scheitert die Aktualisierungsphase mit einem `PodEvictionFailure`-Fehler. Für dieses Szenario können Sie das Erzwingungs-Flag mit der `update-nodegroup-version`-Anforderung anwenden, um die Pods zu löschen.

4. Der Knoten wird nach dem Entfernen jedes Pods gesperrt und wartet 60 Sekunden lang. Dies geschieht, damit der Service-Controller keine neuen Anfragen an diesen Knoten sendet und diesen Knoten aus seiner Liste der aktiven Knoten entfernt.
5. Es sendet eine Beendigungsanforderung an die Auto-Scaling-Gruppe für den abgesperrten Knoten.
6. Es wiederholt die vorherigen Aktualisierungsschritte, bis keine Knoten in der Knotengruppe mehr vorhanden sind, die mit der früheren Version der Startvorlage bereitgestellt wurden.

Die Minimal-Aktualisierungsstrategie umfasst folgende Schritte:

1. Zunächst werden alle Knoten der Knotengruppe gesperrt, sodass der Service-Controller keine neuen Anfragen an diese Knoten sendet.
2. Es wählt nach dem Zufallsprinzip einen Knoten aus, der aktualisiert werden muss, bis zum Maximum der für die Knotengruppe nicht verfügbar ist.
3. Es entleert die Pods aus den ausgewählten Knoten. Wenn die Pods den Knoten nicht innerhalb von 15 Minuten verlassen und es kein Erzwingungs-Flag gibt, scheitert die Aktualisierungsphase mit einem `PodEvictionFailure`-Fehler. Für dieses Szenario können Sie das Erzwingungs-Flag mit der `update-nodegroup-version`-Anforderung anwenden, um die Pods zu löschen.
4. Nachdem jeder Pod entfernt wurde und 60 Sekunden gewartet hat, sendet er eine Beendigungsanforderung an die Auto-Scaling-Gruppe für die ausgewählten Knoten. Die Auto-Scaling-Gruppe erstellt neue Knoten (entsprechend der Anzahl der ausgewählten Knoten), um die fehlende Kapazität zu ersetzen.
5. Es wiederholt die vorherigen Aktualisierungsschritte, bis keine Knoten in der Knotengruppe mehr vorhanden sind, die mit der früheren Version der Startvorlage bereitgestellt wurden.

`PodEvictionFailure`-Fehler während der Aktualisierungsphase

Die folgenden sind bekannte Gründe, die zu einem `PodEvictionFailure`-Fehler in dieser Phase führen:

Aggressive PDB

Aggressive PDB ist auf dem Pod definiert oder es gibt mehrere PDBs, die auf denselben Pod zeigen.

Bereitstellung unterstützt alle Taints

Sobald jeder Pod entfernt wurde, wird erwartet, dass der Knoten leer ist, da der Knoten in den vorherigen Schritten verunreinigt ist. Wenn die Bereitstellung jedoch jeden Taint toleriert, ist der Knoten eher nicht leer, was zu einem Pod-Bereinigungsfehler führt.

Abskalierungsphase

Die Abskalierungsphase verringert die maximale Größe der Auto-Scaling-Gruppe und die gewünschte Größe um eins, um zu den Werten zurückzukehren, bevor die Aktualisierung gestartet wurde.

Wenn der Upgrade-Workflow feststellt, dass der Cluster-Autoscaler die Knotengruppe während der Herunterskalierungsphase des Workflows skaliert, wird er sofort beendet, ohne die Knotengruppe wieder auf ihre ursprüngliche Größe zu bringen.

Verwaltete Knoten mit Startvorlagen anpassen

Für ein Höchstmaß an Anpassung können Sie verwaltete Knoten mit Ihrer eigenen Startvorlage bereitstellen, die auf den Schritten auf dieser Seite basiert. Die Verwendung einer Startvorlage ermöglicht Funktionen wie das Bereitstellen von Bootstrap-Argumenten während der Bereitstellung eines Knotens (z. B. zusätzliche Kubelet-Argumente), das Zuweisen von IP-Adressen zu Pods aus einem anderen CIDR-Block als der dem Knoten zugewiesenen IP-Adresse, das Bereitstellen Ihres eigenen benutzerdefinierten AMI für Knoten oder das Bereitstellen Ihres eigenen benutzerdefinierten CNI für Knoten.

Wenn Sie beim ersten Erstellen einer verwalteten Knotengruppe Ihre eigene Startvorlage angeben, haben Sie auch später mehr Flexibilität. Wenn Sie eine verwaltete Knotengruppe mit Ihrer eigenen Startvorlage bereitstellen, können Sie sie schrittweise mit einer anderen Version derselben Startvorlage aktualisieren. Wenn Sie Ihre Knotengruppe auf eine andere Version Ihrer Startvorlage aktualisieren, werden alle Knoten in der Gruppe wiederverwertet, damit sie der neuen Konfiguration der angegebenen Startvorlagenversion entsprechen.

Verwaltete Knotengruppen werden immer mit einer Startvorlage bereitgestellt, die mit der Amazon EC2 Auto Scaling Scaling-Gruppe verwendet werden kann. Wenn Sie keine Startvorlage bereitstellen, erstellt die Amazon-EKS-API automatisch eine mit Standardwerten in Ihrem Konto. Es wird jedoch nicht empfohlen, automatisch generierte Startvorlagen zu ändern. Außerdem können vorhandene Knotengruppen, die keine benutzerdefinierte Startvorlage verwenden, nicht direkt

aktualisiert werden. Stattdessen müssen Sie eine neue Knotengruppe mit einer benutzerdefinierten Startvorlage erstellen.

Grundlagen der Konfiguration der Vorlage starten

Sie können eine Amazon EC2 Auto Scaling Scaling-Startvorlage mit der AWS-Managementkonsole AWS CLI oder einem AWS SDK erstellen. Weitere Informationen finden Sie unter Erstellen einer Startvorlage für eine Auto Scaling Scaling-Gruppe im Amazon EC2 Auto Scaling Scaling-Benutzerhandbuch. Einige der Einstellungen in einer Startvorlage ähneln den Einstellungen, die für die Konfiguration verwalteter Knoten verwendet werden. Beim Bereitstellen oder Aktualisieren einer Knotengruppe mit einer Startvorlage müssen einige Einstellungen entweder in der Knotengruppenkonfiguration oder in der Startvorlage angegeben werden. Geben Sie eine Einstellung nicht an beiden Stellen an. Wenn eine Einstellung vorhanden ist, wo sie nicht sollte, schlagen Vorgänge wie das Erstellen oder Aktualisieren einer Knotengruppe fehl.

In der folgenden Tabelle werden die Einstellungen aufgelistet, die in einer Startvorlage nicht zulässig sind. Es listet auch ähnliche Einstellungen auf, falls verfügbar, die in der Konfiguration der verwalteten Knotengruppe erforderlich sind. Die aufgelisteten Einstellungen sind die Einstellungen, die in der Konsole angezeigt werden. Sie haben möglicherweise ähnliche, aber unterschiedliche Namen in der AWS CLI und im SDK.

Startvorlage - Verboten	Amazon-EKS-Knotengruppenkonfiguration
Subnetz in Netzwerkschnittstellen (Hinzufügen einer Netzwerkschnittstelle)	Subnets (Subnetze) unter Node group network configuration (Netzwerkkonfiguration der Knotengruppe) auf der Seite Specify networkin g (Netzwerk angeben)
IAM-Instance-Profil in Erweiterte Details	Node IAM role (Knoten-IAM-Rolle) unter Node group configuration (Knotengruppenkonf iguration) auf der Seite Configure Node group (Knotengruppe konfigurieren)
Verhalten beim Herunterfahren und Stop – Ruhezustand Verhalten in Erweiterte Details. Standardeinstellung Nicht in Startvorlage einschließen für beide Einstellungen beibehalt en.	Keine Entsprechung Amazon EKS muss den Instance-Lebenszyklus steuern, nicht die Auto-Scaling-Gruppe.

In der folgenden Tabelle sind die unzulässigen Einstellungen in einer Konfiguration einer verwalteten Knotengruppe aufgeführt. Sie listet auch ähnliche Einstellungen auf, falls vorhanden, die in einer Startvorlage erforderlich sind. Die aufgelisteten Einstellungen sind die Einstellungen, die in der Konsole angezeigt werden. Sie haben möglicherweise ähnliche Namen in der AWS CLI und im SDK.

Konfiguration der Amazon-EKS-Knotengruppe — Verboten	Startvorlage
(Nur wenn Sie ein benutzerdefiniertes AMI in einer Startvorlage angegeben haben) AMI type (AMI-Typ) unter Node Group compute configuration (Computing-Konfiguration der Knotengruppe) auf der Seite Set compute and scaling configuration (Computing- und Skalierungskonfiguration festlegen) – Die Konsole zeigt Specified in launch template (In Startvorlage angegeben) und die angegebene AMI-ID an. Wenn Anwendungs- und Betriebssystem-Images (Amazon Machine Image) nicht in der Startvorlage kein AMI-Typ angegeben wurde, können Sie in der Knotengruppenkonfiguration ein AMI auswählen.	Application and OS Images (Amazon Machine Image) (Anwendungs- und Betriebssystem-Images (Amazon Machine Image)) unter Inhalt der Startvorlage – Sie müssen eine ID angeben, wenn Sie eine der folgenden Anforderungen haben: • Verwenden eines benutzerdefinierten AMI Wenn Sie eine AMI angeben, die nicht den unter Angabe einer AMI aufgeführten Anforderungen entspricht, schlägt die Bereitstellung der Knotengruppe fehl. • Sie möchten Benutzerdaten bereitstellen, um Argumente für die Datei bereitzustellen, die in einem für Amazon EKS optimierten AMI enthalten ist. Sie können Ihre Instances so konfigurieren, dass sie Pods eine deutlich höhere Anzahl von IP-Adressen zuweisen, Pods IP-Adressen aus einem anderen CIDR-Block als dem der Instance zuweisen oder einen privaten Cluster ohne ausgehenden Internetzugang bereitstellen. Weitere Informationen finden Sie unter den folgenden Themen: • Zuweisung weiterer IP-Adressen mit Präfixen zu Amazon-EKS-Knoten

Konfiguration der Amazon-EKS-Knotengruppe — Verboten	Startvorlage
	• Bereitstellung von Pods in alternativen Subnetzen mit benutzerdefiniertem Netzwerk • Bereitstellung privater Cluster mit eingeschränktem Internetzugang • Angabe eines AMI
Datenträgergröße unter Node Group compute configuration (Computing-Konfiguration der Knotengruppe) auf der Seite Set compute and scaling configuration – Die Konsole zeigt Specified in launch template (In Startvorlage angegeben) an.	Größe unter Speicher (Volumes) (Hinzufügen eines neuen Volumes) enthalten. Sie müssen dies in der Startvorlage angeben.
SSH key pair (SSH-Schlüsselpaar) unter Node group configuration (Knotengruppenkonfiguration) auf der Seite Specify Networking (Netzwerk angeben) – Die Konsole zeigt den in der Startvorlage angegebenen Schlüssel an oder Not specified in launch template (Nicht in der Startvorlage angegeben).	Schlüsselpaarname in Schlüsselpaar (Login).
Bei Verwendung einer Startvorlage können Sie keine Quell-Sicherheitsgruppen angeben, für die der Fernzugriff zulässig ist.	Sicherheitsgruppen unter Netzwerkeinstellungen für die Instance oder Sicherheitsgruppen unter Netzwerkschnittstellen (Netzwerkschnittstelle hinzufügen), aber nicht beides. Weitere Informationen finden Sie unter the section called “Benutzerdefinierte Sicherheitsgruppen”.

Note

- Wenn Sie eine Knotengruppe mithilfe einer Startvorlage bereitstellen, geben Sie in einer Startvorlage unter Inhalt der Startvorlage null oder einen Instance-Typ an. Alternativ können Sie 0 bis 20 Instance-Typen für Instance-Typen auf der Seite Einstellen von Compute- und Skalierungskonfiguration in der Konsole angeben. Oder Sie können dies mit anderen Tools tun, die die Amazon EKS API verwenden. Wenn Sie in einer Startvorlage einen Instance-Typ angeben und diese Startvorlage zum Bereitstellen Ihrer Knotengruppe verwenden, können Sie keine Instance-Typen in der Konsole oder mit anderen Tools angeben, welche die Amazon-EKS-API verwenden. Wenn Sie für eine Startvorlage, in der Konsole oder mit anderen Tools, welche die Amazon-EKS-API verwenden, keinen Instance-Typ angeben, wird der `t3.medium`-Instance-Typ verwendet. Wenn Ihre Knotengruppe den Spot-Kapazitätstyp verwendet, empfehlen wir, mehrere Instance-Typen über die Konsole anzugeben. Weitere Informationen finden Sie unter the section called “Kapazitätstypen für verwaltete Knotengruppen”.
- Wenn Container, die Sie für die Knotengruppe bereitstellen, den Instance-Metadaten-Service Version 2 verwenden, stellen Sie sicher, dass Sie die Hop-Limits für Metadaten auf 2 in Ihrer Startvorlage festsetzen. Weitere Informationen finden Sie unter Instance-Metadaten und Benutzerdaten im EC2 Amazon-Benutzerhandbuch.
- Startvorlagen unterstützen das `InstanceRequirements`-Feature zur flexiblen Auswahl des Instance-Typs nicht.

EC2 Amazon-Instances taggen

Sie können den `TagSpecification` Parameter einer Startvorlage verwenden, um anzugeben, welche Tags auf EC2 Amazon-Instances in Ihrer Knotengruppe angewendet werden sollen. Die IAM-Entität, die das `CreateNodegroup` oder aufruft, `UpdateNodegroupVersion` APIs muss über Berechtigungen für `ec2:RunInstances` und verfügen`ec2:CreateTags`, und die Tags müssen der Startvorlage hinzugefügt werden.

Benutzerdefinierte Sicherheitsgruppen

Sie können eine Startvorlage verwenden, um benutzerdefinierte EC2 Amazon-Sicherheitsgruppen anzugeben, die auf Instances in Ihrer Knotengruppe angewendet werden sollen. Dies kann entweder im Parameter für Sicherheitsgruppen auf Instance-Ebene oder als Teil der Konfigurationsparameter der Netzwerkschnittstelle sein. Sie können eine Instance nicht über eine Startvorlage starten, die

Sicherheitsgruppen und eine Netzwerkschnittstelle angibt. Beachten Sie die folgenden Bedingungen, die für die Verwendung benutzerdefinierter Sicherheitsgruppen mit verwalteten Knotengruppen gelten:

- Bei Verwendung von erlaubt Amazon EKS nur Startvorlagen mit einer einzigen Netzwerkschnittstellenspezifikation. AWS-Managementkonsole
- Standardmäßig wendet Amazon EKS die Cluster-Sicherheitsgruppe zu den Instances in Ihrer Knotengruppe hinzu, um die Kommunikation zwischen Knoten und der Steuerungsebene zu erleichtern. Wenn Sie benutzerdefinierte Sicherheitsgruppen in der Startvorlage mit einer der oben genannten Optionen angeben, fügt Amazon EKS die Cluster-Sicherheitsgruppe nicht hinzu. Sie müssen daher sicherstellen, dass die eingehenden und ausgehenden Regeln Ihrer Sicherheitsgruppen die Kommunikation mit dem Endpunkt Ihres Clusters ermöglichen. Wenn Ihre Sicherheitsgruppenregeln falsch sind, können die Worker-Knoten dem Cluster nicht beitreten. Weitere Informationen zu Sicherheitsgruppenregeln finden Sie unter the section called “Anforderungen an Sicherheitsgruppen”.
- Wenn Sie SSH-Zugriff auf die Instances in Ihrer Knotengruppe benötigen, müssen Sie eine Sicherheitsgruppe einschließen, die diesen Zugriff zulässt.

EC2 Amazon-Benutzerdaten

Die Einführungsvorlage enthält einen Abschnitt für benutzerdefinierte Benutzerdaten. In diesem Abschnitt können Sie die Konfigurationseinstellungen für Ihre Knotengruppe angeben, ohne manuell individuelle benutzerdefinierte Einstellungen erstellen zu müssen AMIs. Weitere Informationen zu den für Bottlerocket verfügbaren Einstellungen finden Sie unter Benutzerdaten verwenden am. GitHub

Sie können EC2 Amazon-Benutzerdaten in Ihrer Startvorlage angeben, die Sie `cloud-init` beim Starten Ihrer Instances verwenden. Weitere Informationen dazu finden Sie in der cloud-init-Dokumentation. Ihre Benutzerdaten können verwendet werden, um allgemeine Konfigurationsvorgänge durchzuführen. Dieser Filter umfasst die folgenden Optionen:

- Einschließen von Benutzern oder Gruppen
- Installieren von Paketen

EC2 Amazon-Benutzerdaten in Startvorlagen, die mit verwalteten Knotengruppen verwendet werden, müssen im mehrteiligen MIME-Archivformat für Amazon Linux AMIs und im TOML-Format für Bottlerocket vorliegen. AMIs Dies liegt daran, dass Ihre Benutzerdaten mit Amazon-EKS-Benutzerdaten zusammengeführt werden, die für Knoten erforderlich sind, um dem Cluster

beizutreten. Geben Sie keine Befehle in Ihren Benutzerdaten an, die `kubelet` starten oder ändern. Dies wird als Teil der von Amazon EKS zusammengeführten Benutzerdaten durchgeführt. Bestimmte `kubelet`-Parameter, z. B. das Festlegen von Beschriftungen auf Knoten, können direkt über die API für verwaltete Knotengruppen konfiguriert werden.

> **Note**
>
> Weitere Hinweise zur erweiterten `kubelet`-Anpassung, einschließlich des manuellen Startens oder Übergebens benutzerdefinierter Konfigurationsparameter, finden Sie unter the section called “Angeben eines AMI”. Wenn in einer Startvorlage eine benutzerdefinierte AMI-ID angegeben ist, führt Amazon EKS keine Benutzerdaten zusammen.

Die folgenden Details enthalten weitere Informationen zum Abschnitt Benutzerdaten.

Benutzerdaten von Amazon Linux 2

Sie können mehrere Benutzerdatenblöcke in einer einzelnen mehrteiligen MIME-Datei kombinieren. So können Sie beispielsweise einen Cloud-Boothook, der den Docker-Daemon konfiguriert, mit einem Benutzerdaten-Shell-Skript kombinieren, das ein benutzerdefiniertes Paket installiert. Eine mehrteilige MIME-Datei umfasst folgende Komponenten:

- Deklaration von Inhaltstyp und Teilgrenze – `Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="`
- Deklaration der MIME-Version – `MIME-Version: 1.0`
- Ein oder mehrere Benutzerdatenblöcke mit folgenden Komponenten:
  - Eröffnungsgrenze, die den Beginn eines Benutzerdatenblocks signalisiert – `--==MYBOUNDARY==`
  - Die Inhaltstypdeklaration für den Block: `Content-Type: text/cloud-config; charset="us-ascii"`. Weitere Informationen zu Inhaltstypen finden Sie in der Cloud-Init-Dokumentation.
  - Der Inhalt der Benutzerdaten (z. B. eine Liste von Shell-Befehlen oder `cloud-init`-Direktiven).
  - Abschlussgrenze, die das Ende der mehrteiligen MIME-Datei signalisiert: `--==MYBOUNDARY==--`

Im Folgenden finden Sie ein Beispiel für eine mehrteilige MIME-Datei, die Sie verwenden können, um Ihre eigene zu erstellen.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
echo "Running custom user data script"

--==MYBOUNDARY==--
```

Benutzerdaten von Amazon Linux 2023

Amazon Linux 2023 (AL2023) führt einen neuen Knoteninitialisierungsprozess ein`nodeadm`, der ein YAML-Konfigurationsschema verwendet. Wenn Sie selbstverwaltete Knotengruppen oder eine AMI mit einer Startvorlage verwenden, müssen Sie nun beim Erstellen einer neuen Knotengruppe explizit zusätzliche Cluster-Metadaten angeben. Ein Beispiel für die mindestens erforderlichen Parameter ist wie folgt, wobei jetzt `apiServerEndpoint`, `certificateAuthority` und Service-`cidr` erforderlich sind:

```
---
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name: my-cluster
    apiServerEndpoint: https://example.com
    certificateAuthority: Y2VydGlmaWNhdGVBdXRob3JpdHk=
    cidr: 10.100.0.0/16
```

Normalerweise legen Sie diese Konfiguration in Ihren Benutzerdaten fest, entweder unverändert oder eingebettet in ein MIME-Multipart-Dokument:

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="BOUNDARY"

--BOUNDARY
Content-Type: application/node.eks.aws

---
apiVersion: node.eks.aws/v1alpha1
```

```
kind: NodeConfig spec: [...]

--BOUNDARY--
```

AL2In wurden die Metadaten dieser Parameter aus dem Amazon `DescribeCluster` EKS-API-Aufruf ermittelt. Mit AL2 023 hat sich dieses Verhalten geändert, da durch den zusätzlichen API-Aufruf die Gefahr einer Drosselung bei der Skalierung großer Knoten besteht. Diese Änderung betrifft Sie nicht, wenn Sie verwaltete Knotengruppen ohne Startvorlage verwenden oder wenn Sie Karpenter verwenden. Weitere Informationen zu `certificateAuthority` und Service `cidr` finden Sie unter `DescribeCluster` in der API-Referenz von Amazon EKS.

Hier ist ein vollständiges Beispiel für AL2 023-Benutzerdaten, das ein Shell-Skript für die Anpassung des Nodes (wie die Installation von Paketen oder das Pre-Caching von Container-Images) mit der erforderlichen Konfiguration kombiniert. `nodeadm` Dieses Beispiel veranschaulicht gängige Anpassungen, darunter: * Installation zusätzlicher Systempakete * Vorab-Caching von Container-Images zur Verbesserung der Pod-Startup-Zeit * Einrichtung der HTTP-Proxy-Konfiguration * Konfiguration von `kubelet`-Flags für die Knotenbeschriftung

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="BOUNDARY"

--BOUNDARY
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
set -o errexit
set -o pipefail
set -o nounset

# Install additional packages
yum install -y htop jq iptables-services

# Pre-cache commonly used container images
nohup docker pull public.ecr.aws/eks-distro/kubernetes/pause:3.2 &

# Configure HTTP proxy if needed
cat > /etc/profile.d/http-proxy.sh << 'EOF'
export HTTP_PROXY="http://proxy.example.com:3128"
export HTTPS_PROXY="http://proxy.example.com:3128"
export NO_PROXY="localhost,127.0.0.1,169.254.169.254,.internal"
EOF
```

```
--BOUNDARY
Content-Type: application/node.eks.aws

apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name: my-cluster
    apiServerEndpoint: https://example.com
    certificateAuthority: Y2VydGlmaWNhdGVBdXRob3JpdHk=
    cidr: 10.100.0.0/16
  kubelet:
    config:
      clusterDNS:
      - 10.100.0.10
    flags:
    - --node-labels=app=my-app,environment=production

--BOUNDARY--
```

Bottlerocket-Benutzerdaten

Bottlerocket strukturiert Benutzerdaten im TOML-Format. Sie können Benutzerdaten angeben, die mit den von Amazon EKS bereitgestellten Benutzerdaten zusammengeführt werden sollen. Zum Beispiel können Sie zusätzliche `kubelet`-Einstellungen bereitstellen.

```
[settings.kubernetes.system-reserved]
cpu = "10m"
memory = "100Mi"
ephemeral-storage= "1Gi"
```

Weitere Informationen zu unterstützten Einstellungen finden Sie in der Bottlerocket-Dokumentation. Sie können Knotenbezeichnungen und Taints in Ihren Benutzerdaten konfigurieren. Es wird jedoch empfohlen, diese stattdessen in Ihrer Knotengruppe zu konfigurieren. Amazon EKS wendet diese Konfigurationen an, wenn Sie dies tun.

Wenn Benutzerdaten zusammengeführt werden, wird die Formatierung nicht beibehalten, der Inhalt bleibt jedoch unverändert. Die Konfiguration, die Sie in Ihren Benutzerdaten angeben, überschreibt alle Einstellungen, die von Amazon EKS konfiguriert wurden. Wenn Sie also `settings.kubernetes.max-pods` oder `settings.kubernetes.cluster-dns-ip` festlegen, werden diese Werte in Ihren Benutzerdaten auf die Knoten angewendet.

Amazon EKS unterstützt nicht alle gültigen TOML. Im Folgenden finden Sie eine Liste bekannter nicht unterstützter Formate:

- Zitate in Anführungszeichen: `'quoted "value"' = "value"`
- Anführungszeichen mit Escapezeichen in Werten: `str = "I'm a string. \"You can quote me\""`
- Gemischte Gleitkommazahlen und ganze Zahlen: `numbers = [ 0.1, 0.2, 0.5, 1, 2, 5 ]`
- Gemischte Typen in Arrays: `contributors = ["foo@example.com", { name = "Baz", email = "baz@example.com" }]`
- Kopfzeilen in Klammern mit Anführungszeichen: `[foo."bar.baz"]`

Windows-Benutzerdaten

Windows-Benutzerdaten verwenden Befehle. PowerShell Beim Erstellen einer verwalteten Knotengruppe werden Ihre benutzerdefinierten Benutzerdaten mit den von Amazon EKS verwalteten Benutzerdaten kombiniert. Ihre PowerShell Befehle stehen an erster Stelle, gefolgt von den Befehlen für verwaltete Benutzerdaten, alles innerhalb eines `<powershell></powershell>` Tags.

> **Important**
>
> Beim Erstellen von Windows-Knotengruppen aktualisiert Amazon EKS die `aws-auth ConfigMap`, damit Linux-basierte Knoten dem Cluster beitreten können. Der Dienst konfiguriert die Berechtigungen für Windows nicht automatisch AMIs. Wenn Sie Windows-Knoten verwenden, müssen Sie den Zugriff entweder über die Zugriffseintrag-API oder durch direkte Aktualisierung der `aws-auth ConfigMap` verwalten. Weitere Informationen finden Sie unter the section called “Windows-Support aktivieren”.

> **Note**
>
> Wenn in der Startvorlage keine AMI-ID angegeben ist, verwenden Sie nicht das Amazon-EKS-Bootstrap-Skript von Windows in den Benutzerdaten, um Amazon EKS zu konfigurieren.

Beispielbenutzerdaten lauten wie folgt.

```
<powershell>
Write-Host "Running custom user data script"
</powershell>
```

Angeben eines AMI

Wenn Sie eine der folgenden Anforderungen haben, geben Sie eine AMI-ID in der `ImageId`-Startvorlage. Wählen Sie die Anforderung, die Sie für zusätzliche Informationen haben.

Geben Sie Benutzerdaten an, um Argumente an die **`bootstrap.sh`** Datei zu übergeben, die in einem Amazon EKS-optimierten Linux/Bottlerocket AMI enthalten ist

Bootstrapping ist ein Begriff, der verwendet wird, um das Hinzufügen von Befehlen zu beschreiben, die beim Start einer Instance ausgeführt werden können. Bootstrapping ermöglicht beispielsweise die Verwendung zusätzlicher kubelet-Argumente. Sie können Argumente mithilfe von `eksctl` an das `bootstrap.sh`-Skript übergeben, ohne eine Startvorlage anzugeben. Oder Sie können dies tun, indem Sie die Informationen im Abschnitt Benutzerdaten einer Startvorlage angeben.

eksctl ohne Angabe einer Startvorlage

Erstellen Sie eine Datei mit dem Namen *`my-nodegroup.yaml`* und dem folgenden Inhalt. Ersetzen Sie jede *`example value`* durch Ihre eigenen Werte. Die Argumente `--apiserver-endpoint`, `--b64-cluster-ca` und `--dns-cluster-ip` sind optional. Wenn sie definiert werden, verhindert dies jedoch, dass das `bootstrap.sh`-Skript einen `describeCluster`-Aufruf durchführt. Dies ist nützlich in privaten Cluster-Konfigurationen oder Clustern, in denen Sie häufig Knoten ab- und ausskalieren. Weitere Informationen zum `bootstrap.sh` Skript finden Sie in der Datei bootstrap.sh unter GitHub.

- Das einzige erforderliche Argument ist der Clustername (*`my-cluster`*).
- Informationen zum Abrufen einer optimierten AMI-ID für `ami-`*`1234567890abcdef0`* finden Sie in den folgenden Abschnitten:
  - Rufen Sie das empfohlene Amazon Linux AMI ab IDs
  - Rufen Sie das empfohlene Bottlerocket-AMI ab IDs
  - Rufen Sie das empfohlene Microsoft Windows AMI ab IDs
- Um die *`certificate-authority`* für Ihren Cluster zu überprüfen, führen Sie den folgenden Befehl aus.

```
aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text
 --name my-cluster --region region-code
```

- Um den *api-server-endpoint* für Ihren Cluster zu überprüfen, führen Sie den folgenden Befehl aus.

```
aws eks describe-cluster --query "cluster.endpoint" --output text --name my-
cluster --region region-code
```

- Der Wert für `--dns-cluster-ip`ist Ihr Service CIDR mit `.10`am Ende. Um das *service-cidr* für Ihren Cluster zu überprüfen, führen Sie den folgenden Befehl aus. Wenn der zurückgegebene Wert beispielsweise `ipv4 10.100.0.0/16` ist, ist Ihr Wert *10.100.0.10*.

```
aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr"
 --output text --name my-cluster --region region-code
```

- Dieses Beispiel umfasst ein `kubelet`-Argument, mit dem ein benutzerdefinierter `max-pods`-Wert anhand des `bootstrap.sh`-Skripts festgelegt wird, das im Amazon-EKS-optimierten AMI enthalten ist. Der Name der Knotengruppe darf nicht länger als 63 Zeichen sein. Er muss mit einem Buchstaben oder einer Ziffer beginnen, kann danach aber auch Bindestriche und Unterstriche enthalten. Hilfe zur Auswahl von *my-max-pods-value* finden Sie unter the section called “Von Amazon EKS empfohlene maximale Pods für jeden Amazon-EC2-Instance-Typ”.

```
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code

managedNodeGroups:
  - name: my-nodegroup
    ami: ami-1234567890abcdef0
    instanceType: m5.large
    privateNetworking: true
    disableIMDSv1: true
    labels: { x86-al2-specified-mng }
    overrideBootstrapCommand: |
```

```
      #!/bin/bash
      /etc/eks/bootstrap.sh my-cluster \
        --b64-cluster-ca certificate-authority \
        --apiserver-endpoint api-server-endpoint \
        --dns-cluster-ip service-cidr.10 \
        --kubelet-extra-args '--max-pods=my-max-pods-value' \
        --use-max-pods false
```

Informationen zu allen verfügbaren `eksctl-config`-Dateioptionen finden Sie im Config file schema (Config-Datei-Schema) in der `eksctl`-Dokumentation. Das `eksctl`-Dienstprogramm erstellt eine Startvorlage für Sie und füllt die Benutzerdaten mit den Daten aus, die Sie in der `config`-Datei angeben.

Erstellen Sie eine Knoten-Gruppe mit dem folgenden Befehl.

```
eksctl create nodegroup --config-file=my-nodegroup.yaml
```

Benutzerdaten in einer Startvorlage

Geben Sie die folgenden Informationen im Abschnitt Benutzerdaten Ihrer Startvorlage an. Ersetzen Sie jede *`example value`* durch Ihre eigenen Werte. Die Argumente `--apiserver-endpoint`, `--b64-cluster-ca` und `--dns-cluster-ip` sind optional. Wenn sie definiert werden, verhindert dies jedoch, dass das `bootstrap.sh`-Skript einen `describeCluster`-Aufruf durchführt. Dies ist nützlich in privaten Cluster-Konfigurationen oder Clustern, in denen Sie häufig Knoten ab- und ausskalieren. Weitere Informationen zum `bootstrap.sh` Skript finden Sie in der Datei bootstrap.sh unter GitHub.

- Das einzige erforderliche Argument ist der Clustername (*`my-cluster`*).
- Um die *`certificate-authority`* für Ihren Cluster zu überprüfen, führen Sie den folgenden Befehl aus.

```
aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text
 --name my-cluster --region region-code
```

- Um den *`api-server-endpoint`* für Ihren Cluster zu überprüfen, führen Sie den folgenden Befehl aus.

```
aws eks describe-cluster --query "cluster.endpoint" --output text --name my-
cluster --region region-code
```

- Der Wert für `--dns-cluster-ip` ist Ihr Service CIDR mit `.10` am Ende. Um das *`service-cidr`* für Ihren Cluster zu überprüfen, führen Sie den folgenden Befehl aus. Wenn der zurückgegebene Wert beispielsweise `ipv4 10.100.0.0/16` ist, ist Ihr Wert *`10.100.0.10`*.

  ```
  aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr"
   --output text --name my-cluster --region region-code
  ```

- Dieses Beispiel umfasst ein `kubelet`-Argument, mit dem ein benutzerdefinierter `max-pods`-Wert anhand des `bootstrap.sh`-Skripts festgelegt wird, das im Amazon-EKS-optimierten AMI enthalten ist. Hilfe zur Auswahl von *`my-max-pods-value`* finden Sie unter the section called “Von Amazon EKS empfohlene maximale Pods für jeden Amazon-EC2-Instance-Typ”.

  ```
  MIME-Version: 1.0
  Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

  --==MYBOUNDARY==
  Content-Type: text/x-shellscript; charset="us-ascii"

  #!/bin/bash
  set -ex
  /etc/eks/bootstrap.sh my-cluster \
    --b64-cluster-ca certificate-authority \
    --apiserver-endpoint api-server-endpoint \
    --dns-cluster-ip service-cidr.10 \
    --kubelet-extra-args '--max-pods=my-max-pods-value' \
    --use-max-pods false

  --==MYBOUNDARY==--
  ```

Stellen Sie Benutzerdaten bereit, um Argumente an die **`Start-EKSBootstrap.ps1`**-Datei zu übergeben, die in einer für Amazon EKS optimierten Windows-AMI enthalten ist.

Bootstrapping ist ein Begriff, der verwendet wird, um das Hinzufügen von Befehlen zu beschreiben, die beim Start einer Instance ausgeführt werden können. Sie können Argumente mithilfe von `eksctl` an das `Start-EKSBootstrap.ps1`-Skript übergeben, ohne eine Startvorlage anzugeben. Oder Sie können dies tun, indem Sie die Informationen im Abschnitt Benutzerdaten einer Startvorlage angeben.

Beachten Sie beim Angeben einer benutzerdefinierten Windows-AMI-ID die folgenden Überlegungen:

- Sie müssen eine Startvorlage verwenden und die erforderlichen Bootstrap-Befehle im Abschnitt Benutzerdaten angeben. Um Ihre gewünschte Windows-ID abzurufen, können Sie die Tabelle unter Knoten mit optimiertem Windows erstellen verwenden AMIs.
- Es gibt verschiedene Grenzwerte und Bedingungen. Sie müssen beispielsweise `eks:kube-proxy-windows` zu Ihrer AWS IAM Authenticator-Konfigurationsübersicht etwas hinzufügen. Weitere Informationen finden Sie unter the section called “Grenzen und Bedingungen bei der Angabe einer AMI-ID”.

Geben Sie die folgenden Informationen im Abschnitt Benutzerdaten Ihrer Startvorlage an. Ersetzen Sie jede *`example value`* durch Ihre eigenen Werte. Die Argumente `-APIServerEndpoint`, `-Base64ClusterCA` und `-DNSClusterIP` sind optional. Wenn sie definiert werden, verhindert dies jedoch, dass das `Start-EKSBootstrap.ps1`-Skript einen `describeCluster`-Aufruf durchführt.

- Das einzige erforderliche Argument ist der Clustername (*`my-cluster`*).
- Um die *`certificate-authority`* für Ihren Cluster zu überprüfen, führen Sie den folgenden Befehl aus.

```
aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text --
name my-cluster --region region-code
```

- Um den *`api-server-endpoint`* für Ihren Cluster zu überprüfen, führen Sie den folgenden Befehl aus.

```
aws eks describe-cluster --query "cluster.endpoint" --output text --name my-cluster
 --region region-code
```

- Der Wert für`--dns-cluster-ip`ist Ihr Service CIDR mit`.10`am Ende. Um das *`service-cidr`* für Ihren Cluster zu überprüfen, führen Sie den folgenden Befehl aus. Wenn der zurückgegebene Wert beispielsweise `ipv4 10.100.0.0/16` ist, ist Ihr Wert *`10.100.0.10`*.

```
aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr" --
output text --name my-cluster --region region-code
```

- Weitere Argumente finden Sie unter the section called “Bootstrap-Skript-Konfigurationsparameter”.

> **Note**
>
> Wenn Sie CIDR für den benutzerdefinierten Service verwenden, müssen Sie diesen mithilfe des `-ServiceCIDR`-Parameters angeben. Andernfalls schlägt die DNS-Auflösung für Pods im Cluster fehl.

```
<powershell>
[string]$EKSBootstrapScriptFile = "$env:ProgramFiles\Amazon\EKS\Start-EKSBootstrap.ps1"
& $EKSBootstrapScriptFile -EKSClusterName my-cluster `
  -Base64ClusterCA certificate-authority `
  -APIServerEndpoint api-server-endpoint `
  -DNSClusterIP service-cidr.10
</powershell>
```

Führen Sie ein benutzerdefiniertes AMI aufgrund bestimmter Sicherheits-, Compliance- oder interner Richtlinienanforderungen aus

Weitere Informationen finden Sie unter Amazon Machine Images (AMI) im EC2 Amazon-Benutzerhandbuch. Die Amazon-EKS-AMI-Entwicklungsspezifikation enthält Ressourcen und Konfigurationsskripte für die Erstellung eines benutzerdefinierten Amazon EKS AMI basierend auf Amazon Linux. Weitere Informationen finden Sie unter Amazon-EKS-AMI-Build-Spezifikation auf GitHub. Informationen zum Erstellen benutzerdefinierter AMIs Installationen mit anderen Betriebssystemen finden Sie unter Amazon EKS Sample Custom AMIs unter GitHub.

Sie können keine dynamischen Parameterreferenzen für AMI IDs in Startvorlagen verwenden, die mit verwalteten Knotengruppen verwendet werden.

> **Important**
>
> - Wenn Sie ein AMI angeben, validiert Amazon EKS die in Ihr AMI eingebettete Kubernetes-Version nicht mit der Version der Kontrollebene Ihres Clusters. Sie sind dafür verantwortlich, sicherzustellen, dass die Kubernetes-Version Ihres benutzerdefinierten AMI der Kubernetes-Richtlinie für Versionsverzerrungen entspricht:
>   - Die `kubelet` Version auf Ihren Knoten darf nicht neuer als Ihre Cluster-Version sein

- Die `kubelet` Version auf Ihren Knoten muss mindestens 3 Nebenversionen hinter Ihrer Cluster-Version (für Kubernetes-Version `1.28` oder höher) oder bis zu 2 Nebenversionen hinter Ihrer Cluster-Version (für Kubernetes-Version oder niedriger) sein `1.27`

  Das Erstellen von verwalteten Knotengruppen mit Verstößen gegen den Versionsunterschied kann zu folgenden Ergebnissen führen:

- Knoten können dem Cluster nicht beitreten
- Undefiniertes Verhalten oder API-Inkompatibilitäten
- Cluster-Instabilität oder Workload-Ausfälle

- Bei der Angabe eines AMI führt Amazon EKS keine Benutzerdaten zusammen. Vielmehr sind Sie verantwortlich für die Bereitstellung der erforderlichen `bootstrap`-Befehle für Knoten, um dem Cluster beizutreten. Wenn Ihre Knoten dem Cluster nicht beitreten können, wird Amazon EKS `CreateNodegroup` und `UpdateNodegroupVersion`-Aktionen ebenfalls fehlschlagen.

Grenzen und Bedingungen bei der Angabe einer AMI-ID

Im Folgenden sind die Grenzen und Bedingungen aufgeführt, die mit der Angabe einer AMI-ID mit verwalteten Knotengruppen verbunden sind:

- Sie müssen eine neue Knotengruppe erstellen, um zwischen der Angabe einer AMI-ID in einer Startvorlage und der Nicht-Angabe einer AMI-ID zu wechseln.
- Sie werden in der Konsole nicht benachrichtigt, wenn eine neuere AMI-Version verfügbar ist. Um Ihre Knotengruppe auf eine neuere AMI-Version zu aktualisieren, müssen Sie eine neue Version Ihrer Startvorlage mit einer aktualisierten AMI-ID erstellen. Dann müssen Sie die Knotengruppe mit der neuen Version der Startvorlage aktualisieren.
- Die folgenden Felder können in der API nicht festgelegt werden, wenn Sie eine AMI-ID angeben:
  - `amiType`
  - `releaseVersion`
  - `version`
- Alle in der API festgelegten `taints` werden asynchron angewendet, wenn Sie eine AMI-ID angeben. Um Taints anzuwenden, bevor ein Knoten mit dem Cluster verbunden wird, müssen Sie die Taints über das Befehlszeilen-Flag `--register-with-taints` an `kubelet` in Ihren

Benutzerdaten weitergeben. Weitere Informationen finden Sie unter kubelet in der Kubernetes-Dokumentation.

- Wenn Sie eine benutzerdefinierte AMI-ID für von Windows verwaltete Knotengruppen angeben, fügen Sie `eks:kube-proxy-windows` diese Ihrer AWS IAM Authenticator-Konfigurationsübersicht hinzu. Dies ist erforderlich, damit DNS ordnungsgemäß funktioniert.
  a. Öffnen Sie die AWS IAM Authenticator-Konfigurationsübersicht zur Bearbeitung.

  ```
  kubectl edit -n kube-system cm aws-auth
  ```

  b. Fügen Sie diesen Eintrag zur `groups`-Liste unter jedem `rolearn` hinzu, das Windows-Knoten zugeordnet ist. Ihre Konfigurationsübersicht sollte ähnlich wie .yaml aussehen. aws-auth-cm-windows

  ```
  - eks:kube-proxy-windows
  ```

  c. Speichern Sie die Datei und beenden Sie den Text-Editor.
- Für jedes AMI, das eine benutzerdefinierte Startvorlage verwendet, ist das Standard-`HttpPutResponseHopLimit` für verwaltete Knotengruppen auf 2 festgelegt.

Verwaltete Knotengruppe aus Ihrem Cluster löschen

In diesem Thema wird beschrieben, wie Sie eine verwaltete Amazon-EKS-Knotengruppe löschen. Wenn Sie eine verwaltete Knotengruppe löschen, legt Amazon EKS zunächst die minimale, maximale und gewünschte Größe Ihrer Auto-Scaling-Gruppe auf Null fest. Dies bewirkt dann, dass Ihre Knotengruppe abskaliert wird.

Bevor jede Instance beendet wird, sendet Amazon EKS ein Signal, um diesen Knoten zu entleeren. Während des Drainprozesses führt Kubernetes für jeden Pod auf dem Knoten Folgendes aus: führt alle konfigurierten `preStop` Lifecycle-Hooks aus, sendet `SIGTERM` Signale an die Container und wartet dann auf das ordnungsgemäße Herunterfahren`terminationGracePeriodSeconds`. Wenn der Knoten nach 5 Minuten nicht entleert wurde, lässt Amazon EKS Auto Scaling die erzwungene Beendigung der Instance fortsetzen. Nachdem alle Instanzen beendet wurden, wird die Auto Scaling Scaling-Gruppe gelöscht.

> **Important**
>
> Wenn Sie eine verwaltete Knotengruppe löschen, die eine Knoten-IAM-Rolle verwendet, die von keiner anderen verwalteten Knotengruppe im Cluster genutzt wird, wird die Rolle

aus `aws-auth ConfigMap` entfernt. Wenn eine der selbstverwalteten Knotengruppen im Cluster dieselbe Knoten-IAM-Rolle verwendet, werden die selbstverwalteten Knoten in den `NotReady`-Status verschoben. Außerdem wird auch der Cluster-Betrieb unterbrochen. Informationen zum Hinzufügen einer Zuordnung für die Rolle, die Sie ausschließlich für selbstverwaltete Knotengruppen verwenden, finden Sie unter the section called "Zugriffseinträge erstellen". Voraussetzung ist, dass die Plattformversion Ihres Clusters mindestens der im Abschnitt „Voraussetzungen" unter IAM-Benutzern Zugriff auf Kubernetes mit EKS-Zugriffseinträgen gewähren angegebenen Mindestversion entspricht. Wenn Ihre Plattformversion älter ist als die erforderliche Mindestversion für Zugriffseinträge, können Sie den Eintrag wieder in `aws-auth ConfigMap` hinzufügen. Weitere Informationen erhalten Sie durch Eingabe von `eksctl create iamidentitymapping --help` im Terminal.

Sie können eine verwaltete Knotengruppe wie folgt löschen:

- the section called "`eksctl`"
- the section called "AWS-Managementkonsole"
- the section called "AWS CLI"

`eksctl`

Eine verwaltete Knotengruppe mit **`eksctl`** löschen

Geben Sie den folgenden Befehl ein. Ersetzen Sie jede `<example value>` durch Ihre eigenen Werte.

```
eksctl delete nodegroup \
  --cluster <my-cluster> \
  --name <my-mng> \
  --region <region-code>
```

Weitere Optionen finden Sie unter Löschen und Löschen von Knotengruppen in der `eksctl`-Dokumentation.

AWS-Managementkonsole

Eine verwaltete Knotengruppe mit AWS-Managementkonsole löschen

1. Öffnen Sie die Amazon-EKS-Konsole.

2. Wählen Sie auf der Seite Clusters den Cluster aus, der die zu löschende Knotengruppe enthält.
3. Wählen Sie auf der ausgewählten Cluster-Seite die Registerkarte Datenverarbeitung aus.
4. Wählen Sie im Abschnitt Node groups (Knotengruppen) die zu löschende Knotengruppe aus. Wählen Sie dann Löschen aus.
5. Geben Sie im Bestätigungsdialogfeld Knotengruppe löschen den Namen der Knotengruppe ein. Wählen Sie dann Löschen aus.

AWS CLI

Löschen Sie eine verwaltete Knotengruppe mit AWS CLI

1. Geben Sie den folgenden Befehl ein. Ersetzen Sie jede `<example value>` durch Ihre eigenen Werte.

```
aws eks delete-nodegroup \
  --cluster-name <my-cluster> \
  --nodegroup-name <my-mng> \
  --region <region-code>
```

2. Wenn `cli_pager=` es in der CLI-Konfiguration festgelegt ist, verwenden Sie die Pfeiltasten auf Ihrer Tastatur, um durch die Antwortausgabe zu blättern. Drücken Sie die q-Taste, wenn Sie fertig sind.

   Weitere Optionen finden Sie unter dem `delete-nodegroup` Befehl in der AWS CLI-Befehlsreferenz.

Knoten selbst mit selbstverwalteten Knoten verwalten

Ein Cluster enthält einen oder mehrere Amazon-EC2-Knoten, auf denen Pods geplant sind. Amazon-EKS-Knoten werden in Ihrem AWS-Konto ausgeführt und stellen eine Verbindung zur Steuerebene Ihres Clusters über den Cluster-API-Server-Endpunkt her. Sie werden basierend auf Amazon-EC2-Preisen in Rechnung gestellt. Weitere Informationen dazu finden Sie unter Amazon EC2 – Preise.

Ein Cluster kann mehrere Knotengruppen enthalten. Jede Knotengruppe enthält eine oder mehrere Knoten, die in einer Gruppe von Amazon EC2 Auto Scaling bereitgestellt werden. Der Instance-Typ der Knoten innerhalb der Gruppe kann variieren, z. B. wenn die attributbasierte Instance-Typauswahl mit Karpenter verwendet wird. Alle Instances in einer Knotengruppe müssen die Amazon-EKS-Knoten-IAM-Rolle verwenden.

Amazon EKS bietet spezialisierte Amazon Machine Images (AMIs), die als für Amazon EKS optimierte AMIs bezeichnet werden. Die AMIs sind so konfiguriert, dass sie mit Amazon EKS funktionieren. Zu ihren Komponenten gehören `containerd`, `kubelet`, und der AWS-IAM-Authenticator. Das AMI enthält auch ein spezialisiertes Bootstrap-Skript, mit dem Sie die Steuerebene Ihres Clusters automatisch erkennen und eine Verbindung damit herstellen können.

Wenn Sie den Zugriff auf den öffentlichen Endpunkt Ihres Clusters mithilfe von CIDR-Blöcken einschränken, empfehlen wir, dass Sie auch den privaten Endpunktzugriff aktivieren. Auf diese Weise können Knoten mit dem Cluster kommunizieren. Wenn der private Endpunkt nicht aktiviert ist, müssen die CIDR-Blöcke, die Sie für den öffentlichen Zugriff angeben, die Ausgangsquellen aus Ihrer VPC enthalten. Weitere Informationen finden Sie unter the section called “Cluster-Endpunktzugriff”.

Informationen zum Hinzufügen selbstverwalteter Knoten zu Ihrem Amazon-EKS-Cluster finden Sie in den folgenden Themen. Wenn Sie selbstverwaltete Knoten manuell starten, fügen Sie jedem Knoten das folgende Tag hinzu und stellen Sie sicher, dass `<cluster-name>` mit Ihrem Cluster übereinstimmt. Weitere Informationen dazu finden Sie unter Hinzufügen und Löschen von Markierungen für einzelne Ressourcen. Wenn Sie die Schritte in der Anleitung ausführen, wird die erforderliche Markierung für Sie zum Knoten hinzugefügt.

Schlüssel	Value (Wert)
`kubernetes.io/cluster/<cluster-name>`	`owned`

> **Important**
>
> Tags im Amazon EC2 Instance Metadata Service (IMDS) sind nicht mit EKS-Knoten kompatibel. Wenn Instance Metadata Tags aktiviert sind, wird die Verwendung von Schrägstrichen („/“) in Tag-Werten verhindert. Diese Einschränkung kann zu Fehlern beim Starten von Instances führen, insbesondere bei der Verwendung von Knotenmanagement-Tools wie Karpenter oder Cluster Autoscaler, da diese Services für ihre ordnungsgemäße Funktion auf Tags mit Schrägstrichen angewiesen sind.

Weitere Informationen zu Worker-Knoten aus einer allgemeinen Kubernetes-Perspektive finden Sie unter Nodes in der Kubernetes-Dokumentation.

Themen

- Selbstverwaltete Amazon-Linux-Knoten erstellen
- Selbstverwaltete Bottlerocket-Knoten erstellen
- Selbstverwaltete Microsoft-Windows-Knoten erstellen
- Selbstverwaltete Ubuntu Linux-Knoten erstellen
- Selbstverwaltete Knoten für Ihren Cluster aktualisieren

Selbstverwaltete Amazon-Linux-Knoten erstellen

Dieses Thema beschreibt Hinweise zum Starten von Auto-Scaling-Gruppen von Linux-Knoten, die mit Ihrem Amazon-EKS-Cluster registriert sind. Nachdem die Knoten dem Cluster beigetreten sind, können Sie Kubernetes-Anwendungen darin bereitstellen. Sie können auch selbstverwaltete Amazon Linux-Knoten mit `eksctl` oder dem AWS-Managementkonsole starten. Informationen dazu, wie Sie Knoten auf AWS Outposts starten müssen, finden Sie unterthe section called “Knoten”.

- Ein vorhandener Amazon-EKS-Cluster. Informationen zum Bereitstellen finden Sie unter the section called “Erstellen eines Clusters”. Wenn Sie Subnetze in der AWS Region haben, in der Sie AWS Outposts, AWS Wavelength oder AWS Local Zones aktiviert haben, dürfen diese Subnetze bei der Erstellung Ihres Clusters nicht weitergegeben worden sein.
- Eine vorhandene IAM-Rolle, die von den Knoten verwendet werden soll. Informationen zum Erstellen finden Sie unter the section called “Knoten-IAM-Rolle”. Wenn diese Rolle keine der beiden Richtlinien für die VPC CNI enthält, ist die folgende separate Rolle für die VPC-CNI-Pods erforderlich.
- (Optional, aber empfohlen) Das Amazon-VPC-CNI-Plugin für Kubernetes-Add-On, konfiguriert mit einer eigenen IAM-Rolle, der die erforderliche IAM-Richtlinie zugeordnet ist. Weitere Informationen finden Sie unter the section called “Konfiguration für IRSA”.
- Vertrautheit mit den unter Wählen Sie einen optimalen EC2 Amazon-Node-Instance-Typ aufgeführten Überlegungen. Je nachdem, welchen Instance-Typ Sie wählen, kann es zusätzliche Voraussetzungen für Ihren Cluster und Ihre VPC geben.

Sie können selbstverwaltete Linux-Knoten mit einer der folgenden Optionen starten:

- the section called “`eksctl`”
- the section called “AWS-Managementkonsole”

`eksctl`

Selbstverwaltete Linux-Knoten mithilfe der **`eksctl`** starten

1. Installieren Sie Version `0.215.0` oder höher des auf Ihrem Gerät installierten `eksctl` Befehlszeilentools oder AWS CloudShell. Informationen zum Installieren und Aktualisieren von `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.
2. (Optional) Wenn die verwaltete IAM-Richtlinie AmazonEKS_CNI_Policy Ihrer IAM-Rolle des Amazon-EKS-Knoten zugeordnet ist, empfehlen wir, sie stattdessen einer IAM-Rolle zuzuweisen, die Sie dem Kubernetes-Servicekonto `aws-node` zuordnen. Weitere Informationen finden Sie unter the section called “Konfiguration für IRSA”.
3. Der folgende Befehl erstellt eine Knotengruppe in einem bestehenden Cluster. Ersetzen Sie *`al-nodes`* durch einen Namen für Ihre Knotengruppe. Der Name der Knotengruppe darf nicht länger als 63 Zeichen sein. Er muss mit einem Buchstaben oder einer Ziffer beginnen, kann danach aber auch Bindestriche und Unterstriche enthalten. Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters. Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein. Ersetzen Sie den Rest der *`example value`* durch Ihre eigenen Werte. Die Knoten werden standardmäßig mit derselben Kubernetes-Version wie die Steuerungsebene erstellt.

   Bevor Sie einen Wert für auswählen`--node-type`, lesen Sie den Artikel Wählen Sie einen optimalen EC2 Amazon-Node-Instance-Typ.

   *`my-key`*Ersetzen Sie es durch den Namen Ihres EC2 Amazon-Schlüsselpaars oder öffentlichen Schlüssels. Dieser Schlüssel wird für den SSH-Zugriff zu Ihren Knoten verwendet, nachdem diese gestartet wurden. Wenn Sie noch kein EC2 Amazon-Schlüsselpaar haben, können Sie eines in der erstellen AWS-Managementkonsole. Weitere Informationen finden Sie unter EC2 Amazon-Schlüsselpaare im EC2 Amazon-Benutzerhandbuch.

   Erstellen Sie Ihre Knoten-Gruppe mit dem folgenden Befehl.

   > **Important**
   >
   > Wenn Sie eine Knotengruppe in AWS Outposts-, Wavelength- oder Local Zone-Subnetzen bereitstellen möchten, gibt es zusätzliche Überlegungen:
   >
   > - Die Subnetze dürfen beim Erstellen des Clusters nicht übergeben worden sein.

- Sie müssen die Knotengruppe mit einer Konfigurationsdatei erstellen, die die Subnetze und `volumeType: gp2` angibt. Weitere Informationen dazu finden Sie unter Verwenden von Config-Dateien und im Config-Datei-Schema in der `eksctl`-Dokumentation.

```
eksctl create nodegroup \
  --cluster my-cluster \
  --name al-nodes \
  --node-type t3.medium \
  --nodes 3 \
  --nodes-min 1 \
  --nodes-max 4 \
  --ssh-access \
  --managed=false \
  --ssh-public-key my-key
```

Zum Bereitstellen einer Knotengruppe, die:

- kann Pods eine deutlich höhere Anzahl von IP-Adressen zuweisen als die Standardkonfiguration, siehe the section called “IP-Adressen erhöhen”.
- kann `IPv4`-Adressen an Pods aus einem anderen CIDR-Block als dem der Instance zuweisen, siehe the section called “Benutzerdefinierte Netzwerke”.
- kann `IPv6`-Adressen an Pods und Services zuweisen, siehe the section called “IPv6”.
- hat keinen ausgehenden Internetzugriff, siehe the section called “Private Cluster”.

  Geben Sie den folgenden Befehl ein, um eine vollständige Liste aller verfügbaren Optionen und Standardwerte anzuzeigen.

  ```
  eksctl create nodegroup --help
  ```

  Sollten Knoten nicht in den Cluster aufgenommen werden können, finden Sie weitere Informationen unter the section called “Knoten können nicht mit dem Cluster verknüpft werden” im Kapitel zur Fehlerbehebung.

  Eine Beispielausgabe sieht wie folgt aus. Mehrere Zeilen werden ausgegeben, während die Knoten erstellt werden. Die letzte Ausgabezeile ähnelt der folgenden Beispielzeile.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

4. (Optional) ??? Stellen Sie eine Beispielanwendung bereit, um Ihren Cluster und Ihre Linux-Worker-Knoten zu testen.
5. Wir empfehlen, den Pod-Zugriff auf das IMDS zu blockieren, wenn die folgenden Bedingungen erfüllt sind:
   - Sie planen, allen Ihren Kubernetes-Servicekonten IAM-Rollen zuzuweisen, damit Pods nur die Mindestberechtigungen haben, die sie benötigen.
   - Keine Pods im Cluster benötigen aus anderen Gründen Zugriff auf den Amazon EC2 Instance Metadata Service (IMDS), z. B. zum Abrufen der aktuellen AWS Region.

   Weitere Informationen finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.

AWS-Managementkonsole

Schritt 1: Selbstverwaltete Linux-Knoten mit AWS-Managementkonsole starten

1. Laden Sie die neueste Version der AWS CloudFormation Vorlage herunter.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2025-11-26/
amazon-eks-nodegroup.yaml
```

2. Warten Sie, bis der Status des Clusters als `ACTIVE` angezeigt wird. Wenn Sie Ihre Knoten starten, bevor der Cluster aktiv ist, werden die Knoten nicht mit dem Cluster registriert und Sie müssen sie neu starten.
3. Öffnen Sie die AWS CloudFormation -Konsole.
4. Wählen Sie Create stack (Stack erstellen) und dann With new resources (standard) (Mit neuen Ressourcen [Standard]) aus.
5. Wählen Sie für Specify template (Vorlage festlegen) Upload a template file (Vorlagendatei hochladen) aus und wählen Sie dann Choose file (Datei wählen).
6. Wählen Sie die heruntergeladene `amazon-eks-nodegroup.yaml`-Datei aus.
7. Klicken Sie auf Weiter.
8. Geben Sie auf der Seite Specify stack details (Stack-Details angeben) die folgenden Parameter ein und klicken Sie dann auf Next (Weiter):

- Stack-Name: Wählen Sie einen Stack-Namen für Ihren AWS CloudFormation Stack. Sie können ihn beispielsweise *my-cluster-nodes* nennen. Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein.
- ClusterName: Geben Sie den Namen ein, den Sie bei der Erstellung Ihres Amazon EKS-Clusters verwendet haben. Dieser Name muss exakt mit dem Cluster-Namen übereinstimmen, andernfalls werden Ihre Knoten dem Cluster nicht beitreten.
- ClusterControlPlaneSecurityGroup: Wählen Sie den SecurityGroupsWert aus der AWS CloudFormation Ausgabe aus, die Sie bei der Erstellung Ihrer VPC generiert haben.

  Die folgenden Schritte zeigen einen Vorgang zum Abrufen der entsprechenden Gruppe.

  a. Öffnen Sie die Amazon-EKS-Konsole.

  b. Wählen Sie den Namen des Clusters.

  c. Wählen Sie die Registerkarte Network (Network) aus.

  d. Verwenden Sie den Wert Zusätzliche Sicherheitsgruppen als Referenz, wenn Sie aus der ClusterControlPlaneSecurityGroupDropdownliste auswählen.
- ApiServerEndpoint: Geben Sie den API-Server-Endpunkt für Ihren EKS-Cluster ein. Dies finden Sie im Abschnitt „Details“ der EKS-Cluster-Konsole
- CertificateAuthorityData: Geben Sie die Base64-kodierten Zertifizierungsstellendaten ein, die Sie auch im Abschnitt „Details“ der EKS-Cluster-Konsole finden.
- ServiceCidr: Geben Sie den CIDR-Bereich ein, der für die Zuweisung von IP-Adressen zu Kubernetes-Diensten innerhalb des Clusters verwendet wird. Dieser befindet sich auf der Registerkarte „Netzwerk“ der EKS-Cluster-Konsole.
- AuthenticationMode: Wählen Sie den im EKS-Cluster verwendeten Authentifizierungsmodus aus, indem Sie auf der Registerkarte „Zugriff“ in der EKS-Cluster-Konsole nachsehen.
- NodeGroupName: Geben Sie einen Namen für Ihre Knotengruppe ein. Dieser Name kann zu einem späteren Zeitpunkt zum Identifizieren der Auto-Scaling-Knotengruppe verwendet werden, die für Ihre Knoten erstellt wurde. Der Name der Knotengruppe darf nicht länger als 63 Zeichen sein. Er muss mit einem Buchstaben oder einer Ziffer beginnen, kann danach aber auch Bindestriche und Unterstriche enthalten.
- NodeAutoScalingGroupMinSize: Geben Sie die Mindestanzahl von Knoten ein, auf die Ihre Auto Scaling Scaling-Gruppe für Knoten skalieren kann.

- NodeAutoScalingGroupDesiredCapacity: Geben Sie die gewünschte Anzahl von Knoten ein, auf die bei der Erstellung Ihres Stacks skaliert werden soll.
- NodeAutoScalingGroupMaxSize: Geben Sie die maximale Anzahl von Knoten ein, auf die Ihre Auto Scaling Scaling-Gruppe für Knoten skalieren kann.
- NodeInstanceType: Wählen Sie einen Instance-Typ für Ihre Knoten. Weitere Informationen finden Sie unter the section called “Amazon EC2-Instance-Typen”.
- NodeImageIdSSMParam: Vorab mit dem Amazon EC2 Systems Manager Manager-Parameter eines kürzlich von Amazon EKS optimierten Amazon Linux 2023 AMI für eine variable Kubernetes-Version gefüllt. Um eine andere Kubernetes-Nebenversion zu verwenden, die von Amazon EKS unterstützt wird, ersetzen Sie *1.XX* durch eine andere unterstützte Version. Wir empfehlen, dieselbe Kubernetes-Version wie Ihr Cluster anzugeben.

  Sie können es auch *amazon-linux-2023* durch einen anderen AMI-Typ ersetzen. Weitere Informationen finden Sie unter the section called “Holen Sie sich die neuesten IDs”.

  > **Note**
  >
  > Die Amazon EKS-Knoten AMIs basieren auf Amazon Linux. Sie können Sicherheits- oder Datenschutzereignisse für Amazon Linux 2023 im Amazon Linux Security Center verfolgen oder den zugehörigen RSS-Feed abonnieren. Sicherheits- oder Datenschutzereignisse enthalten eine Übersicht über das Problem, welche Pakete betroffen sind und wie Sie Ihre Instances aktualisieren, um das Problem zu beheben.

- NodeImageId: (Optional) Wenn Sie Ihr eigenes benutzerdefiniertes AMI (anstelle eines für Amazon EKS optimierten AMI) verwenden, geben Sie eine Knoten-AMI-ID für Ihre AWS Region ein. Wenn Sie hier einen Wert angeben, überschreibt dieser alle Werte im NodeImageIdSSMParamFeld.
- NodeVolumeSize: Geben Sie eine Root-Volume-Größe für Ihre Knoten in GiB an.
- NodeVolumeType: Geben Sie einen Root-Volume-Typ für Ihre Knoten an.
- KeyName: Geben Sie den Namen eines Amazon EC2 SSH-Schlüsselpaars ein, mit dem Sie nach dem Start über SSH eine Verbindung zu Ihren Knoten herstellen können. Wenn Sie noch kein EC2 Amazon-Schlüsselpaar haben, können Sie eines in der erstellen AWS-Managementkonsole. Weitere Informationen finden Sie unter EC2 Amazon-Schlüsselpaare im EC2 Amazon-Benutzerhandbuch.
- VpcId: Geben Sie die ID für die VPC ein, die Sie erstellt haben.

- Subnetze: Wählen Sie die Subnetze aus, die Sie für Ihre VPC erstellt haben. Wenn Sie Ihre VPC anhand der unter Erstellen einer Amazon VPC für Ihren Amazon-EKS-Cluster beschriebenen Schritte erstellt haben, geben Sie nur die privaten Subnetze innerhalb der VPC an, in denen Ihre Knoten gestartet werden sollen. Sie können sehen, welche Subnetze privat sind, indem Sie den jeweiligen Subnetzlink in der Registerkarte Networking (Netzwerk) Ihres Clusters öffnen.

  > **Important**
  >
  > - Wenn es sich bei einem oder einigen der Subnetze um öffentliche Subnetze handelt, muss die Einstellung für die automatische Zuweisung öffentlicher IP-Adressen aktiviert sein. Wenn die Einstellung für das öffentliche Subnetz nicht aktiviert ist, wird allen Knoten, die Sie in diesem öffentlichen Subnetz bereitstellen, keine öffentliche IP-Adresse zugewiesen und sie können nicht mit dem Cluster oder anderen Diensten kommunizieren. AWS Wenn das Subnetz vor dem 26. März 2020 mithilfe einer der Amazon AWS CloudFormation EKS-VPC-Vorlagen oder mithilfe `eksctl` von bereitgestellt wurde, ist die automatische Zuweisung öffentlicher IP-Adressen für öffentliche Subnetze deaktiviert. Informationen dazu, wie Sie die Zuweisung öffentlicher IP-Adressen für ein Subnetz aktivieren, finden Sie unter Ändern des Attributs für die öffentliche IPv4 Adressierung für Ihr Subnetz. Wenn der Knoten in einem privaten Subnetz bereitgestellt wird, kann er über ein NAT-Gateway mit dem Cluster und anderen AWS Diensten kommunizieren.
  > - Wenn die Subnetze keinen Internetzugang haben, stellen Sie sicher, dass Sie die Überlegungen und zusätzlichen Schritte unter Bereitstellen privater Cluster mit eingeschränktem Internetzugang kennen.
  > - Wenn Sie AWS Outposts-, Wavelength- oder Local Zone-Subnetze auswählen, dürfen die Subnetze bei der Erstellung des Clusters nicht übergeben worden sein.

9. Treffen Sie die gewünschte Auswahl auf der Seite Configure stack options (Stackoptionen konfigurieren) und wählen Sie dann Next (Weiter) aus.
10. Aktivieren Sie das Kontrollkästchen links neben Ich bestätige, dass AWS CloudFormation möglicherweise IAM-Ressourcen erstellt werden. , und wählen Sie dann Stapel erstellen aus.
11. Wenn Ihr Stack fertig erstellt wurde, wählen Sie ihn in der Konsole aus und klicken Sie auf Outputs (Ausgaben). Wenn Sie den `EKS API and ConfigMap` Authentifizierungsmodus `EKS API` oder verwenden, ist dies der letzte Schritt.
12. Wenn Sie den `ConfigMap` Authentifizierungsmodus verwenden, notieren Sie den NodeInstanceRolefür die Knotengruppe, die erstellt wurde.

Schritt 2: Knoten, die Ihrem Cluster beitreten sollen, aktivieren

> **Note**
>
> Die folgenden zwei Schritte sind nur erforderlich, wenn Sie den Configmap-Authentifizierungsmodus innerhalb des EKS-Clusters verwenden. Wenn Sie Knoten in einer privaten VPC ohne ausgehenden Internetzugang gestartet haben, stellen Sie außerdem sicher, dass Knoten Ihrem Cluster von der VPC aus beitreten können.

1. Überprüfen Sie, ob Sie bereits über eine `aws-auth` `ConfigMap` verfügen.

   ```
   kubectl describe configmap -n kube-system aws-auth
   ```

2. Wenn eine `aws-auth` `ConfigMap` angezeigt wird, aktualisieren Sie sie nach Bedarf.

   a. Öffnen Sie `ConfigMap` zum Bearbeiten.

   ```
   kubectl edit -n kube-system configmap/aws-auth
   ```

   b. Fügen Sie nach Bedarf einen neuen `mapRoles`-Eintrag hinzu. Setzen Sie den `rolearn` Wert auf den NodeInstanceRoleWert, den Sie im vorherigen Verfahren aufgezeichnet haben.

   ```
   [...]
   data:
     mapRoles: |
       - rolearn: <ARN of instance role (not instance profile)>
         username: system:node:{{EC2PrivateDNSName}}
         groups:
           - system:bootstrappers
           - system:nodes
   [...]
   ```

   c. Speichern Sie die Datei und beenden Sie den Text-Editor.

3. Wenn die Fehlermeldung `Error from server (NotFound): configmaps "aws-auth" not found` angezeigt wird, wenden Sie die standardmäßige `ConfigMap` an.

   a. Laden Sie die Konfigurationszuordnung herunter.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/
   aws-auth-cm.yaml
   ```

b. Stellen Sie in der `aws-auth-cm.yaml` Datei den `rolearn` Wert auf den NodeInstanceRoleWert ein, den Sie im vorherigen Verfahren aufgezeichnet haben. Hierzu können Sie einen Texteditor verwenden oder *`my-node-instance-role`* ersetzen und den folgenden Befehl ausführen:

```
sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-instance-
role|' aws-auth-cm.yaml
```

c. Wenden Sie die Konfiguration an. Die Ausführung dieses Befehls kann einige Minuten dauern.

```
kubectl apply -f aws-auth-cm.yaml
```

4. Sehen Sie sich den Status Ihrer Knoten an und warten Sie, bis diese in den `Ready`-Status eintreten.

```
kubectl get nodes --watch
```

Geben Sie `Ctrl+C` ein, um zum Shell-Prompt zurückzukehren.

> **Note**
>
> Wenn Sie Autorisierungs- oder Ressourcenfehler erhalten, finden Sie weitere Informationen unter the section called “Nicht autorisiert oder Zugriff verweigert (`kubectl`)” im Thema zur Fehlerbehebung.

Sollten Knoten nicht in den Cluster aufgenommen werden können, finden Sie weitere Informationen unter the section called “Knoten können nicht mit dem Cluster verknüpft werden” im Kapitel zur Fehlerbehebung.

5. (Nur GPU-Knoten) Wenn Sie einen GPU-Instance-Typ und das für Amazon EKS optimierte beschleunigte AMI ausgewählt haben, müssen Sie das NVIDIA-Geräte-Plug-In für Kubernetes als a DaemonSet auf Ihrem Cluster anwenden. Ersetzen Sie es *`vX.X.X`* durch die gewünschte s-device-pluginNVIDIA/K8-Version, bevor Sie den folgenden Befehl ausführen.

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/
deployments/static/nvidia-device-plugin.yml
```

Schritt 3: Zusätzliche Aktionen

1. (Optional) ??? Stellen Sie eine Beispielanwendung bereit, um Ihren Cluster und Ihre Linux-Worker-Knoten zu testen.
2. (Optional) Wenn die von AmazonEKS_CNI_Policy verwaltete IAM-Richtlinie (wenn Sie einen `IPv4` Cluster haben) oder die *`AmazonEKS_CNI_IPv6_Policy`* (die Sie selbst erstellt haben, wenn Sie einen `IPv6` Cluster haben) mit Ihrer Amazon EKS-Knoten-IAM-Rolle verknüpft ist, empfehlen wir, sie stattdessen einer IAM-Rolle zuzuweisen, die Sie dem Kubernetes-Servicekonto zuordnen. `aws-node` Weitere Informationen finden Sie unter the section called “Konfiguration für IRSA”.
3. Wir empfehlen, den Pod-Zugriff auf das IMDS zu blockieren, wenn die folgenden Bedingungen erfüllt sind:
   - Sie planen, allen Ihren Kubernetes-Servicekonten IAM-Rollen zuzuweisen, damit Pods nur die Mindestberechtigungen haben, die sie benötigen.
   - Keine Pods im Cluster benötigen aus anderen Gründen Zugriff auf den Amazon EC2 Instance Metadata Service (IMDS), z. B. zum Abrufen der aktuellen AWS Region.

   Weitere Informationen finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.

Selbstverwaltete Bottlerocket-Knoten erstellen

> **Note**
>
> Verwaltete Knotengruppen bieten möglicherweise einige Vorteile für Ihren Anwendungsfall. Weitere Informationen finden Sie unter the section called “Verwaltete Knotengruppen”.

In diesem Thema wird beschrieben, wie Sie Auto-Scaling-Gruppen von Bottlerocket-Knoten starten, die mit Ihrem Amazon-EKS-Cluster registriert sind. Bottlerocket ist ein Linux-basiertes Open-Source-Betriebssystem, mit dem Sie Container auf virtuellen AWS Maschinen oder Bare-Metal-Hosts ausführen können. Nachdem die Knoten dem Cluster beigetreten sind, können Sie Kubernetes-Anwendungen darin bereitstellen. Weitere Informationen zu Bottlerocket finden Sie in der Dokumentation unter Using a Bottlerocket AMI with Amazon EKS on GitHub and Custom AMI support. `eksctl`

Informationen zu direkten Upgrades finden Sie unter Bottlerocket Update Operator unter. GitHub

Important

- Amazon EKS-Knoten sind EC2 Standard-Amazon-Instances, und sie werden Ihnen auf der Grundlage der normalen EC2 Amazon-Instance-Preise in Rechnung gestellt. Weitere Informationen finden Sie unter EC2 Amazon-Preise.
- Sie können Bottlerocket-Knoten in erweiterten Amazon EKS-Clustern auf AWS Outposts starten, aber Sie können sie nicht in lokalen Clustern auf Outposts starten. AWS Weitere Informationen finden Sie unter Amazon EKS in AWS Outposts.
- Sie können die Bereitstellung auf EC2 Amazon-Instances mit x86 oder ARM-Prozessoren durchführen. Sie können jedoch nicht auf Instances bereitstellen, die über Inferentia-Chips verfügen.
- Bottlerocket ist kompatibel mit. AWS CloudFormation Es gibt jedoch keine offizielle CloudFormation Vorlage, die kopiert werden kann, um Bottlerocket-Knoten für Amazon EKS bereitzustellen.
- Bottlerocket-Images werden nicht mit einem SSH-Server oder einer Shell geliefert. Sie können out-of-band Zugriffsmethoden verwenden, um die SSH-Aktivierung des Admin-Containers zu ermöglichen und einige Bootstrapping-Konfigurationsschritte mit Benutzerdaten zu durchlaufen. Weitere Informationen finden Sie in diesen Abschnitten im bottlerocket README.md auf GitHub:
  - Exploration (Erkundung)
  - Administrator-Container
  - Kubernetes-Einstellungen

Für diesen Vorgang ist `eksctl` Version `0.215.0` oder höher erforderlich. Sie können Ihre -Version mit dem folgenden Befehl überprüfen:

```
eksctl version
```

Anweisungen zum Installieren oder Aktualisieren von `eksctl` finden Sie unter Installation in der `eksctl`-Dokumentation. HINWEIS: Dieses Verfahren funktioniert nur für Cluster, die mit `eksctl` erstellt wurden.

1. Kopieren Sie den folgenden Inhalt auf Ihr Gerät. Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters. Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung

beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein. Ersetzen Sie *ng-bottlerocket* durch einen Namen für Ihre Knotengruppe. Der Name der Knotengruppe darf nicht länger als 63 Zeichen sein. Er muss mit einem Buchstaben oder einer Ziffer beginnen, kann danach aber auch Bindestriche und Unterstriche enthalten. Um die Bereitstellung auf Arm-Instances durchzuführen, ersetzen Sie *m5.large* durch einen Arm-Instance-Typ. *my-ec2-keypair-name*Ersetzen Sie es durch den Namen eines Amazon EC2 SSH-Schlüsselpaars, mit dem Sie nach dem Start über SSH eine Verbindung zu Ihren Knoten herstellen können. Wenn Sie noch kein EC2 Amazon-Schlüsselpaar haben, können Sie eines in der erstellen AWS-Managementkonsole. Weitere Informationen finden Sie unter EC2 Amazon-Schlüsselpaare im EC2 Amazon-Benutzerhandbuch. Ersetzen Sie alle verbleibenden Beispielwerte durch Ihre eigenen Werte. Nachdem Sie das Ersetzen vorgenommen haben, führen Sie den geänderten Befehl aus, um die `bottlerocket.yaml`-Datei zu erstellen.

Wenn Sie einen Arm EC2 Amazon-Instance-Typ angeben, lesen Sie AMIs vor der Bereitstellung die Überlegungen in Amazon EKS-optimiertem Arm Amazon Linux durch. Anweisungen zur Bereitstellung mit einem benutzerdefinierten AMI finden Sie in der Dokumentation unter Building Bottlerocket on GitHub und Custom AMI support. `eksctl` Um eine verwaltete Knotengruppe bereitzustellen, stellen Sie ein benutzerdefiniertes AMI mithilfe einer Startvorlage bereit. Weitere Informationen finden Sie unter the section called “Startvorlagen”.

> **Important**
>
> Um eine Knotengruppe in AWS Outposts-, AWS Wavelength- oder AWS Local Zone-Subnetzen bereitzustellen, übergeben Sie beim Erstellen des Clusters keine AWS Outposts-, AWS Wavelength- oder AWS Local Zone-Subnetze. Sie müssen die Subnetze im folgenden Beispiel angeben. Weitere Informationen finden Sie unter Verwenden von Config-Dateien und im Config-Datei-Schema in der `eksctl`-Dokumentation. Ersetzen Sie es *region-code* durch die AWS Region, in der sich Ihr Cluster befindet.

```
cat >bottlerocket.yaml <<EOF
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
```

```
  name: my-cluster
  region: region-code
  version: '1.33'

iam:
  withOIDC: true

nodeGroups:
  - name: ng-bottlerocket
    instanceType: m5.large
    desiredCapacity: 3
    amiFamily: Bottlerocket
    ami: auto-ssm
    iam:
       attachPolicyARNs:
          - arn:aws: iam::aws:policy/AmazonEKSWorkerNodePolicy
          - arn:aws: iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
          - arn:aws: iam::aws:policy/AmazonSSMManagedInstanceCore
          - arn:aws: iam::aws:policy/AmazonEKS_CNI_Policy
    ssh:
        allow: true
        publicKeyName: my-ec2-keypair-name
EOF
```

2. Stellen Sie den Treiber mit dem folgenden Befehl bereit.

```
eksctl create nodegroup --config-file=bottlerocket.yaml
```

   Eine Beispielausgabe sieht wie folgt aus.

   Mehrere Zeilen werden ausgegeben, während die Knoten erstellt werden. Die letzte Ausgabezeile ähnelt der folgenden Beispielzeile.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Optional) Erstellen eines KubernetesPersistenteraufeinem Bottlerocket-Knoten mit demAmazon-EBS-CSI-Plugin. Der standardmäßige Amazon-EBS-Treiber basiert auf Dateisystem-Tools, die nicht in Bottlerocket enthalten sind. Weitere Informationen zur Erstellung einer Speicherklasse mit dem Treiber finden Sie unter the section called “Amazon EBS”.
4. (Optional) Standardmäßig setzt `kube-proxy` den Kernelparameter `nf_conntrack_max` auf einen Standardwert, der sich von dem unterscheiden kann, was Bottlerocket ursprünglich beim

Booten festgelegt hat. Um die Standardeinstellung von Bottlerocket beizubehalten, bearbeiten Sie die `kube-proxy`-Konfiguration im folgenden Befehl.

```
kubectl edit -n kube-system daemonset kube-proxy
```

Fügen Sie `--conntrack-max-per-core` und `--conntrack-min` zu den `kube-proxy`-Argumenten im folgenden Beispiel hinzu. Eine Einstellung von`0`impliziert keine Änderung.

```
      containers:
      - command:
        - kube-proxy
        - --v=2
        - --config=/var/lib/kube-proxy-config/config
        - --conntrack-max-per-core=0
        - --conntrack-min=0
```

5. (Optional) Bereitstellen einer-Beispielanwendung, um Ihre Bottlerocket-Knoten zu testen.
6. Wir empfehlen, den Pod-Zugriff auf das IMDS zu blockieren, wenn die folgenden Bedingungen erfüllt sind:
   - Sie planen, allen Ihren Kubernetes-Servicekonten IAM-Rollen zuzuweisen, damit Pods nur die Mindestberechtigungen haben, die sie benötigen.
   - Keine Pods im Cluster benötigen aus anderen Gründen Zugriff auf den Amazon EC2 Instance Metadata Service (IMDS), z. B. zum Abrufen der aktuellen AWS Region.

   Weitere Informationen finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.

Selbstverwaltete Microsoft-Windows-Knoten erstellen

Dieses Thema beschreibt Hinweise zum Starten von Auto-Scaling-Gruppen von -Windows-Knoten, die mit Ihrem Amazon-EKS-Cluster registriert sind. Nachdem die Knoten dem Cluster beigetreten sind, können Sie Kubernetes-Anwendungen darin bereitstellen.

> ⚠ Important
>
> - Amazon EKS-Knoten sind EC2 Standard-Amazon-Instances, und sie werden Ihnen auf der Grundlage der normalen EC2 Amazon-Instance-Preise in Rechnung gestellt. Weitere Informationen finden Sie unter EC2 Amazon-Preise.

- Sie können Windows-Knoten in erweiterten Amazon EKS-Clustern auf AWS Outposts starten, aber Sie können sie nicht in lokalen Clustern auf AWS Outposts starten. Weitere Informationen finden Sie unter Amazon EKS in AWS Outposts.

Aktivieren Sie den Windows-Support für Ihren Cluster. Es wird empfohlen, wichtige Überlegungen zu berücksichtigen, bevor Sie eine Windows-Knotengruppe starten. Weitere Informationen finden Sie unter the section called “Windows-Support aktivieren”.

Sie können selbstverwaltete Windows-Knoten mit einer der folgenden Methoden starten:

- the section called “`eksctl`”
- the section called “AWS-Managementkonsole”

`eksctl`

Selbstverwaltete Windows-Knoten mithilfe von **`eksctl`** starten

Bei diesem Verfahren wird davon ausgegangen, dass Sie `eksctl` installiert haben und dass Ihre `eksctl`-Version mindestens `0.215.0` ist. Sie können Ihre Version mit dem folgenden Befehl überprüfen.

```
eksctl version
```

Eine Installations- und Upgrade-Anleitung für `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.

> **Note**
>
> Dieses Verfahren funktioniert nur für Cluster, die mit `eksctl` erstellt wurden.

1. (Optional) Wenn die von AmazonEKS_CNI_Policy verwaltete IAM-Richtlinie (wenn Sie einen `IPv4` Cluster haben) oder die *`AmazonEKS_CNI_IPv6_Policy`* (die Sie selbst erstellt haben, wenn Sie einen `IPv6` Cluster haben) mit Ihrer Amazon EKS-Knoten-IAM-Rolle verknüpft ist, empfehlen wir, sie stattdessen einer IAM-Rolle zuzuweisen, die Sie dem Kubernetes-Servicekonto zuordnen. `aws-node` Weitere Informationen finden Sie unter the section called “Konfiguration für IRSA”.

2. Bei diesem Verfahren wird davon ausgegangen, dass Sie einen bestehenden Cluster haben. Wenn Sie nicht über einen Amazon-EKS-Cluster und eine Amazon-Linux-Knotengruppe verfügen, zu der Sie eine Windows-Knotengruppe hinzufügen können, empfehlen wir Ihnen, the section called “Cluster erstellen (eksctl)” zu befolgen. Dieses Handbuch bietet eine vollständige Anleitung zum Erstellen eines Amazon-EKS-Clusters mit Amazon-Linux-Knoten.

   Erstellen Sie Ihre Knoten-Gruppe mit dem folgenden Befehl. Ersetzen *region-code* Sie es durch die Region, in der sich Ihr Cluster befindet. AWS Ersetzen Sie *my-cluster* mit Ihrem Clusternamen. Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein. Ersetzen Sie *ng-windows* durch einen Namen für Ihre Knotengruppe. Der Name der Knotengruppe darf nicht länger als 63 Zeichen sein. Er muss mit einem Buchstaben oder einer Ziffer beginnen, kann danach aber auch Bindestriche und Unterstriche enthalten. Sie können es durch *2019* ersetzen`2022`, um Windows Server 2022 zu verwenden. Ersetzen Sie die restlichen Beispielwerte durch Ihre eigenen Werte.

   > **Important**
   >
   > Um eine Knotengruppe in AWS Outposts-, AWS Wavelength- oder AWS Local Zone-Subnetzen bereitzustellen, übergeben Sie die Subnetze AWS Outposts, Wavelength oder Local Zone nicht, wenn Sie den Cluster erstellen. Erstellen Sie die Knotengruppe mit einer Konfigurationsdatei, in der Sie die Subnetze AWS Outposts, Wavelength oder Local Zone angeben. Weitere Informationen finden Sie unter Verwenden von Config-Dateien und im Config-Datei-Schema in der `eksctl`-Dokumentation.

```
eksctl create nodegroup \
    --region region-code \
    --cluster my-cluster \
    --name ng-windows \
    --node-type t2.large \
    --nodes 3 \
    --nodes-min 1 \
    --nodes-max 4 \
    --managed=false \
    --node-ami-family WindowsServer2019FullContainer
```

> **Note**
>
> - Wenn Arbeitsknoten dem Cluster nicht beitreten können, finden Sie weitere Informationen unter the section called “Knoten können nicht mit dem Cluster verknüpft werden” im Handbuch zur Fehlerbehebung.
> - Geben Sie den folgenden Befehl ein, um die verfügbaren Optionen für `eksctl`-Befehle anzuzeigen.
>
>   ```
>   eksctl command -help
>   ```

Eine Beispielausgabe sieht wie folgt aus. Mehrere Zeilen werden ausgegeben, während die Knoten erstellt werden. Die letzte Ausgabezeile ähnelt der folgenden Beispielzeile.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Optional) ??? Stellen Sie eine Beispielanwendung bereit, um Ihren Cluster und Ihre Windows-Worker-Knoten zu testen.
4. Wir empfehlen, den Pod-Zugriff auf das IMDS zu blockieren, wenn die folgenden Bedingungen erfüllt sind:
   - Sie planen, allen Ihren Kubernetes-Servicekonten IAM-Rollen zuzuweisen, damit Pods nur die Mindestberechtigungen haben, die sie benötigen.
   - Keine Pods im Cluster benötigen aus anderen Gründen Zugriff auf den Amazon EC2 Instance Metadata Service (IMDS), z. B. zum Abrufen der aktuellen AWS Region.

   Weitere Informationen finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.

AWS-Managementkonsole

Voraussetzungen

- Ein bestehender Amazon-EKS-Cluster und eine Linux-Knotengruppe. Wenn Sie nicht über diese Ressourcen verfügen, empfehlen wir Ihnen, diese mithilfe einer unserer Anleitungen in Erste Schritte zu erstellen. Diese Anleitungen beschreiben, wie Sie einen Amazon-EKS-Cluster mit Linux-Knoten erstellen.

- Eine vorhandene VPC und eine Sicherheitsgruppe, die die Voraussetzungen für einen Amazon-EKS-Cluster erfüllen. Weitere Informationen erhalten Sie unter the section called “VPC- und Subnetz-Anforderungen” und the section called “Anforderungen an Sicherheitsgruppen”. Die Guides in Erste Schritte erstellen eine VPC, die den Anforderungen entspricht. Alternativ können Sie auch den Abschnitt Erstellen einer Amazon VPC für Ihren Amazon-EKS-Cluster befolgen, um manuell eine VPC zu erstellen.
- Ein vorhandener Amazon-EKS-Cluster, der eine VPC und eine Sicherheitsgruppe verwendet, die die Voraussetzungen eines Amazon-EKS-Clusters erfüllt. Weitere Informationen finden Sie unter the section called “Erstellen eines Clusters”. Wenn Sie Subnetze in der AWS Region haben, in der Sie AWS Outposts, AWS Wavelength oder AWS Local Zones aktiviert haben, dürfen diese Subnetze nicht weitergegeben worden sein, als Sie den Cluster erstellt haben.

Schritt 1: Selbstverwaltete Windows-Knoten mit der AWS-Managementkonsole starten

1. Warten Sie, bis der Status des Clusters als `ACTIVE` angezeigt wird. Wenn Sie Ihre Knoten starten, bevor der Cluster aktiv ist, werden die Knoten nicht mit dem Cluster registriert und Sie müssen sie neu starten.
2. Öffnen Sie die AWS CloudFormation -Konsole.
3. Wählen Sie Stack erstellen aus.
4. Wählen Sie unter Vorlage angeben die Option Amazon-S3-URL aus.
5. Kopieren Sie die folgende URL und fügen Sie sie in die Amazon S3 URL (Amazon-S3-URL) ein.

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2023-02-09/amazon-eks-windows-nodegroup.yaml
```

6. Wählen Sie zweimal Next (Weiter) aus.
7. Füllen Sie auf der Seite Quick create stack (Stack schnell erstellen) die folgenden Parameter entsprechend aus:
   - Stack-Name: Wählen Sie einen Stack-Namen für Ihren Stack. AWS CloudFormation Sie können ihn beispielsweise `my-cluster-nodes` nennen.
   - ClusterName: Geben Sie den Namen ein, den Sie bei der Erstellung Ihres Amazon EKS-Clusters verwendet haben.

> **Important**
>
> Dieser Name muss genau mit dem Namen übereinstimmen, den Sie in Schritt 1: Amazon-EKS-Cluster erstellen verwendet haben. Andernfalls können Ihre Knoten dem Cluster nicht beitreten.

- ClusterControlPlaneSecurityGroup: Wählen Sie die Sicherheitsgruppe aus der AWS CloudFormation Ausgabe aus, die Sie bei der Erstellung Ihrer VPC generiert haben. Die folgenden Schritte zeigen eine Methode zum Abrufen der entsprechenden Gruppe.
  a. Öffnen Sie die Amazon-EKS-Konsole.
  b. Wählen Sie den Namen des Clusters.
  c. Wählen Sie die Registerkarte Network (Network) aus.
  d. Verwenden Sie den Wert Zusätzliche Sicherheitsgruppen als Referenz, wenn Sie aus der ClusterControlPlaneSecurityGroupDropdownliste auswählen.
- NodeGroupName: Geben Sie einen Namen für Ihre Knotengruppe ein. Dieser Name kann zu einem späteren Zeitpunkt zum Identifizieren der Auto-Scaling-Knotengruppe verwendet werden, die für Ihre Knoten erstellt wurde. Der Name der Knotengruppe darf nicht länger als 63 Zeichen sein. Er muss mit einem Buchstaben oder einer Ziffer beginnen, kann danach aber auch Bindestriche und Unterstriche enthalten.
- NodeAutoScalingGroupMinSize: Geben Sie die Mindestanzahl von Knoten ein, auf die Ihre Auto Scaling Scaling-Gruppe für Knoten skalieren kann.
- NodeAutoScalingGroupDesiredCapacity: Geben Sie die gewünschte Anzahl von Knoten ein, auf die bei der Erstellung Ihres Stacks skaliert werden soll.
- NodeAutoScalingGroupMaxSize: Geben Sie die maximale Anzahl von Knoten ein, auf die Ihre Auto Scaling Scaling-Gruppe für Knoten skalieren kann.
- NodeInstanceType: Wählen Sie einen Instance-Typ für Ihre Knoten. Weitere Informationen finden Sie unter the section called “Amazon EC2-Instance-Typen”.

> **Note**
>
> Die unterstützten Instance-Typen für die neueste Version des Amazon VPC CNI-Plug-ins für Kubernetes sind in vpc_ip_resource_limit.go on aufgeführt. GitHub Möglicherweise müssen Sie Ihre CNI-Version aktualisieren, um die neuesten

unterstützten Instance-Typen zu nutzen. Weitere Informationen finden Sie unter the section called “Amazon VPC CNI”.

- NodeImageIdSSMParam: Vorab mit dem Amazon EC2 Systems Manager Manager-Parameter der aktuell empfohlenen Amazon EKS-optimierten Windows Core AMI-ID gefüllt. Um die Vollversion von Windows zu verwenden, ersetzen Sie *Core* durch `Full`.
- NodeImageId: (Optional) Wenn Sie Ihr eigenes benutzerdefiniertes AMI (anstelle eines für Amazon EKS optimierten AMI) verwenden, geben Sie eine Knoten-AMI-ID für Ihre AWS Region ein. Wenn Sie einen Wert für dieses Feld angeben, überschreibt dieser alle Werte in dem NodeImageIdSSMParamFeld.
- NodeVolumeSize: Geben Sie eine Root-Volume-Größe für Ihre Knoten in GiB an.
- KeyName: Geben Sie den Namen eines Amazon EC2 SSH-Schlüsselpaars ein, mit dem Sie nach dem Start über SSH eine Verbindung zu Ihren Knoten herstellen können. Wenn Sie noch kein EC2 Amazon-Schlüsselpaar haben, können Sie eines in der erstellen AWS-Managementkonsole. Weitere Informationen finden Sie unter EC2 Amazon-Schlüsselpaare im EC2 Amazon-Benutzerhandbuch.

  > **Note**
  >
  > Wenn Sie hier kein key pair angeben, kann der AWS CloudFormation Stack nicht erstellt werden.

- BootstrapArguments: Geben Sie alle optionalen Argumente an, die an das Node-Bootstrap-Skript übergeben werden sollen, z. B. zusätzliche `kubelet` Argumente mit-`KubeletExtraArgs`.
- Deaktivieren IMDSv1: Standardmäßig unterstützt jeder Knoten den Instanz-Metadatendienst Version 1 (IMDSv1) und IMDSv2. Sie können deaktivieren IMDSv1. Um zu verhindern, dass future Knoten und Pods in der Knotengruppe verwendet werden MDSv1, setzen Sie Disable IMDSv1 auf true. Weitere Informationen finden Sie unter Konfiguration des Instance-Metadatenservice.
- VpcId: Wählen Sie die ID für die VPC aus, die Sie erstellt haben.
- NodeSecurityGroups: Wählen Sie die Sicherheitsgruppe aus, die für Ihre Linux-Knotengruppe erstellt wurde, als Sie Ihre VPC erstellt haben. Wenn Ihren Linux-Knoten mehr als eine Sicherheitsgruppe angehängt ist, geben Sie alle an. Dies z. B., wenn die Linux-Knotengruppe mit `eksctl` erstellt wurde.

- Subnets (Subnetze): Wählen Sie die Subnetze aus, die Sie erstellt haben. Wenn Sie Ihre VPC gemäß den Schritten unter Erstellen einer Amazon VPC für Ihren Amazon-EKS-Cluster erstellt haben, geben Sie nur die privaten Subnetze innerhalb der VPC an, in denen Ihre Knoten gestartet werden sollen.

> **Important**
>
> - Wenn es sich bei einem oder einigen der Subnetze um öffentliche Subnetze handelt, muss die Einstellung für die automatische Zuweisung öffentlicher IP-Adressen aktiviert sein. Wenn die Einstellung für das öffentliche Subnetz nicht aktiviert ist, wird allen Knoten, die Sie in diesem öffentlichen Subnetz bereitstellen, keine öffentliche IP-Adresse zugewiesen und sie können nicht mit dem Cluster oder anderen Diensten kommunizieren. AWS Wenn das Subnetz vor dem 26. März 2020 mithilfe einer der Amazon AWS CloudFormation EKS-VPC-Vorlagen oder mithilfe `eksctl` von bereitgestellt wurde, ist die automatische Zuweisung öffentlicher IP-Adressen für öffentliche Subnetze deaktiviert. Informationen zum Aktivieren der Zuweisung öffentlicher IP-Adressen für ein Subnetz finden Sie unter Ändern des Attributs für die öffentliche IPv4 Adressierung für Ihr Subnetz. Wenn der Knoten in einem privaten Subnetz bereitgestellt wird, kann er über ein NAT-Gateway mit dem Cluster und anderen AWS Diensten kommunizieren.
> - Wenn die Subnetze keinen Internetzugang haben, stellen Sie sicher, dass Sie die Überlegungen und zusätzlichen Schritte unter Bereitstellen privater Cluster mit eingeschränktem Internetzugang kennen.
> - Wenn Sie AWS Outposts-, Wavelength- oder Local Zone-Subnetze auswählen, dürfen die Subnetze bei der Erstellung des Clusters nicht übergeben worden sein.

8. Bestätigen Sie, dass der Stack IAM-Ressourcen erstellen kann, und wählen Sie dann Create stack (Stack erstellen) aus.
9. Wenn Ihr Stack fertig erstellt wurde, wählen Sie ihn in der Konsole aus und klicken Sie auf Outputs (Ausgaben).
10. Notieren Sie das NodeInstanceRolefür die Knotengruppe, die erstellt wurde. Sie benötigen diese, wenn Sie Ihre Amazon-EKS-Windows-Knoten konfigurieren.

Schritt 2: Knoten, die Ihrem Cluster beitreten sollen, aktivieren

1. Überprüfen Sie, ob Sie bereits über eine `aws-auth` `ConfigMap` verfügen.

```
kubectl describe configmap -n kube-system aws-auth
```

2. Wenn eine `aws-auth` `ConfigMap` angezeigt wird, aktualisieren Sie sie nach Bedarf.

   a. Öffnen Sie `ConfigMap` zum Bearbeiten.

   ```
   kubectl edit -n kube-system configmap/aws-auth
   ```

   b. Fügen Sie nach Bedarf neue `mapRoles`-Einträge hinzu. Stellen Sie die `rolearn` Werte auf die NodeInstanceRoleWerte ein, die Sie in den vorherigen Verfahren aufgezeichnet haben.

   ```
   [...]
   data:
     mapRoles: |
   - rolearn: <ARN of linux instance role (not instance profile)>
         username: system:node:{{EC2PrivateDNSName}}
         groups:
           - system:bootstrappers
           - system:nodes
       - rolearn: <ARN of windows instance role (not instance profile)>
         username: system:node:{{EC2PrivateDNSName}}
         groups:
           - system:bootstrappers
           - system:nodes
           - eks:kube-proxy-windows
   [...]
   ```

   c. Speichern Sie die Datei und beenden Sie den Text-Editor.

3. Wenn die Fehlermeldung `Error from server (NotFound): configmaps "aws-auth" not found` angezeigt wird, wenden Sie die standardmäßige `ConfigMap` an.

   a. Laden Sie die Konfigurationszuordnung herunter.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/
   aws-auth-cm-windows.yaml
   ```

   b. Stellen Sie in der `aws-auth-cm-windows.yaml` Datei die `rolearn` Werte auf die entsprechenden NodeInstanceRoleWerte ein, die Sie in den vorherigen Verfahren aufgezeichnet haben. Hierzu können Sie einen Texteditor verwenden oder die Beispielwerte ersetzen und den folgenden Befehl ausführen:

```
sed -i.bak -e 's|<ARN of linux instance role (not instance profile)>|my-node-
linux-instance-role|' \
    -e 's|<ARN of windows instance role (not instance profile)>|my-node-windows-
instance-role|' aws-auth-cm-windows.yaml
```

> **Important**
>
> - Ändern Sie keine weiteren Zeilen in dieser Datei.
> - Verwenden Sie nicht dieselbe IAM-Rolle sowohl für Windows- als auch für Linux-Knoten.

c. Wenden Sie die Konfiguration an. Die Ausführung dieses Befehls kann einige Minuten dauern.

```
kubectl apply -f aws-auth-cm-windows.yaml
```

4. Sehen Sie sich den Status Ihrer Knoten an und warten Sie, bis diese in den `Ready`-Status eintreten.

```
kubectl get nodes --watch
```

Geben Sie `Ctrl+C` ein, um zum Shell-Prompt zurückzukehren.

> **Note**
>
> Wenn Sie Autorisierungs- oder Ressourcenfehler erhalten, finden Sie weitere Informationen unter the section called “Nicht autorisiert oder Zugriff verweigert (`kubectl`)” im Thema zur Fehlerbehebung.

Sollten Knoten nicht in den Cluster aufgenommen werden können, finden Sie weitere Informationen unter the section called “Knoten können nicht mit dem Cluster verknüpft werden” im Kapitel zur Fehlerbehebung.

Schritt 3: Zusätzliche Aktionen

1. (Optional) ??? Stellen Sie eine Beispielanwendung bereit, um Ihren Cluster und Ihre Windows-Worker-Knoten zu testen.

2. (Optional) Wenn die von AmazonEKS_CNI_Policy verwaltete IAM-Richtlinie (wenn Sie einen `IPv4` Cluster haben) oder die *`AmazonEKS_CNI_IPv6_Policy`* (die Sie selbst erstellt haben, wenn Sie einen `IPv6` Cluster haben) mit Ihrer Amazon EKS-Knoten-IAM-Rolle verknüpft ist, empfehlen wir, sie stattdessen einer IAM-Rolle zuzuweisen, die Sie dem Kubernetes-Servicekonto zuordnen. `aws-node` Weitere Informationen finden Sie unter the section called “Konfiguration für IRSA”.
3. Wir empfehlen, den Pod-Zugriff auf das IMDS zu blockieren, wenn die folgenden Bedingungen erfüllt sind:
   - Sie planen, allen Ihren Kubernetes-Servicekonten IAM-Rollen zuzuweisen, damit Pods nur die Mindestberechtigungen haben, die sie benötigen.
   - Keine Pods im Cluster benötigen aus anderen Gründen Zugriff auf den Amazon EC2 Instance Metadata Service (IMDS), z. B. zum Abrufen der aktuellen AWS Region.

   Weitere Informationen finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.

Selbstverwaltete Ubuntu Linux-Knoten erstellen

> **Note**
>
> Verwaltete Knotengruppen bieten möglicherweise einige Vorteile für Ihren Anwendungsfall. Weitere Informationen finden Sie unter the section called “Verwaltete Knotengruppen”.

In diesem Thema wird beschrieben, wie Sie Auto Scaling-Gruppen von Knoten von Ubuntu in Amazon Elastic Kubernetes Service (EKS) oder Knoten von Ubuntu Pro in Amazon Elastic Kubernetes Service (EKS) starten, die bei Ihrem Amazon-EKS-Cluster registriert sind. Ubuntu und Ubuntu Pro für EKS basieren auf dem offiziellen Ubuntu Minimal LTS, enthalten den benutzerdefinierten AWS Kernel, der gemeinsam mit EKS entwickelt wurde AWS, und wurden speziell für EKS gebaut. Ubuntu Pro bietet zusätzliche Sicherheit durch die Unterstützung erweiterter EKS-Supportzeiträume, Kernel-Live-Patches, FIPS-Konformität und die Möglichkeit, eine unbegrenzte Anzahl von Pro-Containern auszuführen.

Nachdem die Knoten dem Cluster beigetreten sind, können Sie darin containerisierte Anwendungen bereitstellen. Weitere Informationen finden Sie in der Dokumentation zu Ubuntu in AWS und zum Benutzerdefinierten AMI-Support in der `eksctl`-Dokumentation.

Important

- Amazon EKS-Knoten sind EC2 Standard-Amazon-Instances, und sie werden Ihnen auf der Grundlage der normalen EC2 Amazon-Instance-Preise in Rechnung gestellt. Weitere Informationen finden Sie unter EC2 Amazon-Preise.
- Sie können Ubuntu-Knoten in erweiterten Amazon EKS-Clustern auf AWS Outposts starten, aber Sie können sie nicht in lokalen Clustern auf AWS Outposts starten. Weitere Informationen finden Sie unter Amazon EKS in AWS Outposts.
- Sie können die Bereitstellung auf EC2 Amazon-Instances mit x86 oder ARM-Prozessoren durchführen. Bei Instances mit Inferentia-Chips muss jedoch möglicherweise zuerst das Neuron SDK installiert werden.

Für diesen Vorgang ist `eksctl` Version `0.215.0` oder höher erforderlich. Sie können Ihre -Version mit dem folgenden Befehl überprüfen:

```
eksctl version
```

Anweisungen zum Installieren oder Aktualisieren von `eksctl` finden Sie unter Installation in der `eksctl`-Dokumentation. HINWEIS: Dieses Verfahren funktioniert nur für Cluster, die mit `eksctl` erstellt wurden.

1. Kopieren Sie den folgenden Inhalt auf Ihr Gerät. Ersetzen Sie `my-cluster` mit dem Namen Ihres Clusters. Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphabetischen Zeichen beginnen und darf nicht mehr als 100 Zeichen umfassen. Ersetzen Sie `ng-ubuntu` durch einen Namen für Ihre Knotengruppe. Der Name der Knotengruppe darf nicht länger als 63 Zeichen sein. Er muss mit einem Buchstaben oder einer Ziffer beginnen, kann danach aber auch Bindestriche und Unterstriche enthalten. Um die Bereitstellung auf Arm-Instances durchzuführen, ersetzen Sie `m5.large` durch einen Arm-Instance-Typ. `my-ec2-keypair-name`Ersetzen Sie es durch den Namen eines Amazon EC2 SSH-Schlüsselpaars, mit dem Sie nach dem Start über SSH eine Verbindung zu Ihren Knoten herstellen können. Wenn Sie noch kein EC2 Amazon-Schlüsselpaar haben, können Sie eines in der erstellen AWS-Managementkonsole. Weitere Informationen finden Sie unter EC2 Amazon-Schlüsselpaare im EC2 Amazon-Benutzerhandbuch. Ersetzen Sie alle verbleibenden Beispielwerte durch Ihre eigenen Werte. Nachdem Sie das Ersetzen vorgenommen haben, führen Sie den geänderten Befehl aus, um die `ubuntu.yaml`-Datei zu erstellen.

> **Important**
>
> Um eine Knotengruppe in AWS Outposts-, AWS Wavelength- oder AWS Local Zone-Subnetzen bereitzustellen, übergeben Sie beim Erstellen des Clusters keine AWS Outposts-, AWS Wavelength- oder AWS Local Zone-Subnetze. Sie müssen die Subnetze im folgenden Beispiel angeben. Weitere Informationen finden Sie unter Verwenden von Config-Dateien und im Config-Datei-Schema in der `eksctl`-Dokumentation. Ersetzen Sie es *`region-code`* durch die AWS Region, in der sich Ihr Cluster befindet.

```
cat >ubuntu.yaml <<EOF
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code
  version: '1.33'

iam:
  withOIDC: true

nodeGroups:
  - name: ng-ubuntu
    instanceType: m5.large
    desiredCapacity: 3
    amiFamily: Ubuntu2204
    iam:
       attachPolicyARNs:
          - arn:aws: iam::aws:policy/AmazonEKSWorkerNodePolicy
          - arn:aws: iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
          - arn:aws: iam::aws:policy/AmazonSSMManagedInstanceCore
          - arn:aws: iam::aws:policy/AmazonEKS_CNI_Policy
    ssh:
        allow: true
        publicKeyName: my-ec2-keypair-name
EOF
```

Um eine Ubuntu Pro-Knotengruppe zu erstellen, ändern Sie einfach den `amiFamily`-Wert in `UbuntuPro2204`.

2. Stellen Sie den Treiber mit dem folgenden Befehl bereit.

```
eksctl create nodegroup --config-file=ubuntu.yaml
```

Eine Beispielausgabe sieht wie folgt aus.

Mehrere Zeilen werden ausgegeben, während die Knoten erstellt werden. Die letzte Ausgabezeile ähnelt der folgenden Beispielzeile.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Optional) Stellen Sie eine Beispielanwendung bereit, um Ihre Ubuntu-Knoten zu testen.
4. Wir empfehlen, den Pod-Zugriff auf das IMDS zu blockieren, wenn die folgenden Bedingungen erfüllt sind:
   - Sie planen, allen Ihren Kubernetes-Servicekonten IAM-Rollen zuzuweisen, damit Pods nur die Mindestberechtigungen haben, die sie benötigen.
   - Keine Pods im Cluster benötigen aus anderen Gründen Zugriff auf den Amazon EC2 Instance Metadata Service (IMDS), z. B. zum Abrufen der aktuellen AWS Region.

   Weitere Informationen finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.

Selbstverwaltete Knoten für Ihren Cluster aktualisieren

Wenn eine neue Amazon-EKS-optimierte AMI veröffentlicht wird, ziehen Sie in Betracht, die Knoten in Ihrer selbstverwalteten Knoten-Gruppe durch die neue AMI zu ersetzen. Ebenso gilt, dass Sie bei einer Aktualisierung der Kubernetes-Version für Ihren Amazon-EKS-Cluster auch die Worker-Knoten aktualisieren, um Knoten mit derselben Kubernetes-Version zu verwenden.

> **Important**
>
> In diesem Thema werden Worker-Knotenaktualisierungen für selbstverwaltete Knotengruppen behandelt. Wenn Sie verwaltete Knotengruppen verwenden, lesen Sie the section called “Aktualisierung”.

Es gibt zwei grundlegende Möglichkeiten, selbstverwaltete Knotengruppen in Ihren Clustern zu aktualisieren, um ein neues AMI zu verwenden:

Anwendungen zu einer neuen Knotengruppe migrieren

Erstellen Sie eine neue Worker-Knotengruppe und migrieren Sie Ihre Pods zu dieser Gruppe. Die Migration zu einer neuen Knotengruppe ist sinnvoller als die Aktualisierung der AMI-ID in einem bestehenden AWS-CloudFormation-Stack. Dies liegt daran, dass der Migrationsprozess die alte Knotengruppe als `NoSchedule` verunreinigt und die Knoten leert, nachdem ein neuer Stack bereit ist, die vorhandene Pod-Workload zu akzeptieren.

AWS-CloudFormation-Knoten-Stack aktualisieren

Aktualisieren Sie den AWS-CloudFormation-Stack für eine vorhandene Worker-Knotengruppe, um das neue AMI zu verwenden. Diese Methode wird nicht bei Knoten-Gruppen unterstützt, die mit `eksctl` erstellt wurden.

Anwendungen zu einer neuen Knotengruppe migrieren

Dieses Thema beschreibt Hinweise zum Erstellen einer neuen Knoten-Gruppe, zum ordnungsgemäßen Migrieren Ihrer vorhandenen Anwendungen zur neuen Gruppe und zum Entfernen der alten Knoten-Gruppe aus Ihrem Cluster. Sie können zu einer neuen Knotengruppe migrieren, indem Sie `eksctl` oder AWS-Managementkonsole benutzen.

- the section called “`eksctl`”
- the section called “AWS-Managementkonsole und AWS CLI”

`eksctl`

Migration Ihrer Anwendungen zu einer neuen Knotengruppe mit **`eksctl`**

Weitere Informationen zur Verwendung von eksctl finden Sie unter Upgrades von nicht verwalteten Knotengruppen in der `eksctl`-Dokumentation.

Für diesen Vorgang ist `eksctl` Version `0.215.0` oder höher erforderlich. Sie können Ihre -Version mit dem folgenden Befehl überprüfen:

```
eksctl version
```

Eine Installations- und Upgrade-Anleitung für `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.

> **Note**
>
> Dieses Verfahren funktioniert nur für Cluster, die mit `eksctl` erstellt wurden.

1. Rufen Sie den Namen Ihrer vorhandenen Knotengruppen ab und ersetzen Sie *`my-cluster`* durch Ihren Clusternamen.

   ```
   eksctl get nodegroups --cluster=my-cluster
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   CLUSTER      NODEGROUP          CREATED               MIN SIZE      MAX SIZE
    DESIRED CAPACITY     INSTANCE TYPE     IMAGE ID
   default      standard-nodes   2019-05-01T22:26:58Z  1             4             3
                  t3.medium         ami-05a71d034119ffc12
   ```

2. Starten Sie eine neue Knotengruppe mit `eksctl` mit dem folgenden Befehl. Ersetzen Sie im Befehl jedes *`example value`* durch Ihre eigenen Werte. Die Versionsnummer darf nicht höher als die Kubernetes-Version für Ihre Steuerebene sein. Außerdem darf sie nicht mehr als zwei Nebenversionen älter sein als die Kubernetes-Version für Ihre Steuerebene. Es wird empfohlen, dieselbe Version wie die Steuerebene zu verwenden.

   Wir empfehlen, den Pod-Zugriff auf das IMDS zu blockieren, wenn die folgenden Bedingungen erfüllt sind:

   - Sie planen, allen Ihren Kubernetes-Servicekonten IAM-Rollen zuzuweisen, damit Pods nur die Mindestberechtigungen haben, die sie benötigen.
   - Keine Pods im Cluster benötigen aus anderen Gründen Zugriff auf den Amazon EC2 Instance Metadata Service (IMDS), z. B. zum Abrufen der aktuellen AWS Region.

     Weitere Informationen finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.

     Um den Pod-Zugriff auf IMDS zu blockieren, fügen Sie dem folgenden Befehl die Option `--disable-pod-imds` hinzu.

> **Note**
>
> Weitere verfügbare Flags und deren Beschreibungen finden Sie unterhttps://eksctl.io/.

```
eksctl create nodegroup \
  --cluster my-cluster \
  --version 1.33 \
  --name standard-nodes-new \
  --node-type t3.medium \
  --nodes 3 \
  --nodes-min 1 \
  --nodes-max 4 \
  --managed=false
```

3. Wenn der vorherige Befehl abgeschlossen ist, bestätigen Sie mit folgendem Befehl, dass alle Ihre Worker-Knoten den `Ready`-Status erreicht haben:

```
kubectl get nodes
```

4. Löschen Sie die ursprüngliche Knotengruppe mit dem folgenden Befehl. Ersetzen Sie im Befehl alle *`example value`* mit Ihren Cluster- und Knotengruppennamen:

```
eksctl delete nodegroup --cluster my-cluster --name standard-nodes-old
```

AWS-Managementkonsole und AWS CLI

Migrieren Sie Ihre Anwendungen mit der AWS-Managementkonsole und AWS CLI auf eine neue Knotengruppe

1. Starten Sie eine neue Knotengruppe, indem Sie die unter Erstellen selbstverwalteter Amazon Linux-Knoten beschriebenen Schritte befolgen.
2. Wenn Ihr Stack fertig erstellt wurde, wählen Sie ihn in der Konsole aus und klicken Sie auf Outputs (Ausgaben).
3. Notieren Sie das NodeInstanceRolefür die Knotengruppe, die erstellt wurde. Sie benötigen diese Informationen zum Hinzufügen der neuen Amazon EKS-Knoten zu Ihrem Cluster.

> **Note**
>
> Wenn Sie der IAM-Rolle Ihrer alten Knotengruppen zusätzliche IAM-Richtlinien angefügt haben, dann sollten Sie die gleichen Richtlinien auch der IAM-Rolle Ihrer neuen Knotengruppe zuweisen, um diese Funktionalität für die neue Gruppe zu erhalten. Dies gilt für Sie, wenn Sie beispielsweise Berechtigungen für den Kubernetes Cluster Autoscaler hinzugefügt haben.

4. Aktualisieren Sie die Sicherheitsgruppen für beide Worker-Knoten-Gruppen, sodass sie miteinander kommunizieren können. Weitere Informationen finden Sie unter the section called “Anforderungen an Sicherheitsgruppen”.

   a. Notieren Sie die Sicherheitsgruppe IDs für beide Knotengruppen. Dies wird als NodeSecurityGroupWert in den AWS CloudFormation Stack-Ausgaben angezeigt.

      Sie können die folgenden AWS CLI-Befehle verwenden, um die Sicherheitsgruppe IDs aus den Stack-Namen abzurufen. In diesen Befehlen `oldNodes` steht der AWS CloudFormation Stack-Name für Ihren älteren Knoten-Stack und `newNodes` der Name des Stacks, zu dem Sie migrieren. Ersetzen Sie jede *`example value`* durch Ihre eigenen Werte.

      ```
      oldNodes="old_node_CFN_stack_name"
      newNodes="new_node_CFN_stack_name"

      oldSecGroup=$(aws cloudformation describe-stack-resources --stack-name $oldNodes \
      --query 'StackResources[?
      ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
      --output text)
      newSecGroup=$(aws cloudformation describe-stack-resources --stack-name $newNodes \
      --query 'StackResources[?
      ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
      --output text)
      ```

   b. Fügen Sie Regeln für eingehenden Datenverkehr für jede Worker-Knoten-Sicherheitsgruppe hinzu, sodass sie voneinander Datenverkehr annehmen können.

      Die folgenden AWS CLI-Befehle fügen jeder Sicherheitsgruppe Regeln für eingehenden Datenverkehr hinzu, die den gesamten Datenverkehr auf allen Protokollen der anderen Sicherheitsgruppe zulassen. Auf diese Weise können Pods in jeder Knoten-Gruppe miteinander kommunizieren, während Sie Ihr Workload zur neuen Gruppe migrieren.

```
aws ec2 authorize-security-group-ingress --group-id $oldSecGroup \
--source-group $newSecGroup --protocol -1
aws ec2 authorize-security-group-ingress --group-id $newSecGroup \
--source-group $oldSecGroup --protocol -1
```

5. Bearbeiten Sie die `aws-auth`-configmap, um die neue Worker-Knoten-Instance-Rolle in RBAC zuzuordnen.

   ```
   kubectl edit configmap -n kube-system aws-auth
   ```

   Fügen Sie einen neuen `mapRoles`-Eintrag für die neue Worker-Knoten-Gruppe hinzu.

   ```
   apiVersion: v1
   data:
     mapRoles: |
       - rolearn: ARN of instance role (not instance profile)
         username: system:node:{{EC2PrivateDNSName}}
         groups:
           - system:bootstrappers
           - system:nodes>
       - rolearn: arn:aws: iam::111122223333:role/nodes-1-16-NodeInstanceRole-
   U11V27W93CX5
         username: system:node:{{EC2PrivateDNSName}}
         groups:
           - system:bootstrappers
           - system:nodes
   ```

   Ersetzen Sie das *`ARN of instance role (not instance profile)`* Snippet durch den NodeInstanceRoleWert, den Sie in einem vorherigen Schritt aufgezeichnet haben. Speichern und schließen Sie dann die Datei, um die aktualisierte configmap anzuwenden.

6. Achten Sie auf den Status Ihrer Knoten und warten Sie, bis Ihre neuen Worker-Knoten Ihrem Cluster beigetreten sind und den Status `Ready` angenommen haben.

   ```
   kubectl get nodes --watch
   ```

7. (Optional) Wenn Sie Kubernetes Cluster Autoscaler verwenden, skalieren Sie die Bereitstellung nach unten auf null (0) Replikate, um Konflikte zwischen Skalierungsaktionen zu vermeiden.

   ```
   kubectl scale deployments/cluster-autoscaler --replicas=0 -n kube-system
   ```

8. Verwenden Sie den folgenden Befehl, um jeden der Knoten, die Sie mit `NoSchedule` entfernen möchten, mit einem Taint zu versehen. Auf diese Weise werden neue Pods auf den Knoten, die Sie ersetzen, nicht geplant oder neu geplant. Weitere Informationen finden Sie unter Taints und Toleranzen in der Kubernetes-Dokumentation.

```
kubectl taint nodes node_name key=value:NoSchedule
```

   Wenn Sie Ihre Knoten auf eine neue Kubernetes-Version aktualisieren, können Sie alle Knoten einer bestimmten Kubernetes-Version (in diesem Fall `1.31`) mit dem folgenden Code-Ausschnitt identifizieren und mit einem Taint versehen. Die Versionsnummer darf nicht höher als die Kubernetes-Version Ihrer Steuerebene sein. Sie darf auch nicht mehr als zwei Nebenversionen älter sein als die Kubernetes-Version Ihrer Steuerebene. Es wird empfohlen, dieselbe Version wie die Steuerebene zu verwenden.

```
K8S_VERSION=1.31
nodes=$(kubectl get nodes -o jsonpath="{.items[?(@.status.nodeInfo.kubeletVersion==
\"v$K8S_VERSION\")].metadata.name}")
for node in ${nodes[@]}
do
    echo "Tainting $node"
    kubectl taint nodes $node key=value:NoSchedule
done
```

9. Bestimmen Sie den DNS-Anbieter Ihres Clusters.

```
kubectl get deployments -l k8s-app=kube-dns -n kube-system
```

   Eine Beispielausgabe sieht wie folgt aus. Dieser Cluster verwendet für die DNS-Auflösung, aber Ihr Cluster kann stattdessen `kube-dns` zurückgeben):

```
NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
coredns   1         1         1            1           31m
```

10. Wenn Ihre aktuelle Bereitstellung weniger als zwei Replikate ausführt, skalieren die Bereitstellung auf zwei Replikate. Ersetzen Sie `kubedns` durch *`coredns`*, falls Ihre vorherige Befehlsausgabe dies stattdessen zurückgegeben hat.

```
kubectl scale deployments/coredns --replicas=2 -n kube-system
```

11 Lassen Sie die einzelnen Knoten, die Sie aus Ihrem Cluster entfernen möchten, mit dem folgenden Befehl sperren:

```
kubectl drain node_name --ignore-daemonsets --delete-local-data
```

Wenn Sie Ihre Knoten auf eine neue Kubernetes-Version aktualisieren, identifizieren und löschen Sie alle Knoten einer bestimmten Kubernetes-Version (in diesem Fall*1.31*) mit dem folgenden Codeausschnitt.

```
K8S_VERSION=1.31
nodes=$(kubectl get nodes -o jsonpath="{.items[?(@.status.nodeInfo.kubeletVersion==
\"v$K8S_VERSION\")].metadata.name}")
for node in ${nodes[@]}
do
    echo "Draining $node"
    kubectl drain $node --ignore-daemonsets --delete-local-data
done
```

12 Nachdem die alten Knoten entladen wurden, widerrufen Sie die Regeln für eingehenden Datenverkehr für Sicherheitsgruppen, die Sie zuvor autorisiert haben. Löschen Sie anschließend den Stack, um die Instanzen zu beenden AWS CloudFormation .

> **Note**
>
> Wenn Sie Ihrer alten Knotengruppen-IAM-Rolle zusätzliche IAM-Richtlinien hinzugefügt haben, z. B. das Hinzufügen von Berechtigungen für den Kubernetes Cluster Autoscaler, trennen Sie diese zusätzlichen Richtlinien von der Rolle, bevor Sie Ihren Stack löschen können. AWS CloudFormation

a. Heben Sie die Regeln für eingehenden Datenverkehr auf, die Sie zuvor für die Knoten-Sicherheitsgruppen erstellt haben. In diesen Befehlen `oldNodes` steht der AWS CloudFormation Stack-Name für Ihren älteren Knoten-Stack und `newNodes` der Name des Stacks, zu dem Sie migrieren.

```
oldNodes="old_node_CFN_stack_name"
newNodes="new_node_CFN_stack_name"

oldSecGroup=$(aws cloudformation describe-stack-resources --stack-name $oldNodes \
```

```
--query 'StackResources[?
ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
--output text)
newSecGroup=$(aws cloudformation describe-stack-resources --stack-name $newNodes \
--query 'StackResources[?
ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
--output text)
aws ec2 revoke-security-group-ingress --group-id $oldSecGroup \
--source-group $newSecGroup --protocol -1
aws ec2 revoke-security-group-ingress --group-id $newSecGroup \
--source-group $oldSecGroup --protocol -1
```

b. Öffnen Sie die AWS CloudFormation -Konsole.

c. Wählen Sie Ihren alten Worker-Knoten-Stack aus.

d. Wählen Sie Löschen aus.

e. Wählen Sie im Bestätigungsdialogfeld Stack löschen Stack löschen aus.

13 Bearbeiten Sie die `aws-auth`-configmap, um die alten Worker-Knoten-Instance-Rolle aus RBAC zu entfernen.

```
kubectl edit configmap -n kube-system aws-auth
```

Löschen Sie den `mapRoles`-Eintrag für die alte Worker-Knoten-Gruppe.

```
apiVersion: v1
data:
  mapRoles: |
    - rolearn: arn:aws: iam::111122223333:role/nodes-1-16-NodeInstanceRole-
W70725MZQFF8
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
    - rolearn: arn:aws: iam::111122223333:role/nodes-1-15-NodeInstanceRole-
U11V27W93CX5
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes>
```

Speichern und schließen Sie die Datei, um die aktualisierte configmap anzuwenden.

14. (Optional) Wenn Sie den Kubernetes-Cluster Autoscaler verwenden, skalieren Sie die Bereitstellung wieder auf ein Replikat.

> **Note**
>
> Außerdem müssen Sie Ihre neue Auto-Scaling-Gruppe (z. B. `k8s.io/cluster-autoscaler/enabled,k8s.io/cluster-autoscaler/my-cluster`) mit einem Tag versehen und den Befehl für die Cluster-Autoscaler-Bereitstellung so aktualisieren, dass er auf die neu getaggte Auto-Scaling-Gruppe verweist. Weitere Informationen finden Sie unter Cluster Autoscaler on. AWS

```
kubectl scale deployments/cluster-autoscaler --replicas=1 -n kube-system
```

15. (Optional) Stellen Sie sicher, dass Sie die neueste Version des Amazon-VPC-CNI-Plugins für Kubernetes verwenden. Möglicherweise müssen Sie Ihre CNI-Version aktualisieren, um die neuesten unterstützten Instance-Typen zu nutzen. Weitere Informationen finden Sie unter the section called “Amazon VPC CNI”.

16. Wenn Ihr Cluster `kube-dns` für die DNS-Auflösung verwendet (siehe [migrate-determine-dns-step]), skalieren Sie in der `kube-dns`-Bereitstellung auf ein Replikat.

```
kubectl scale deployments/kube-dns --replicas=1 -n kube-system
```

Einen AWS CloudFormation Knotenstapel aktualisieren

In diesem Thema wird beschrieben, wie Sie einen vorhandenen AWS CloudFormation selbstverwalteten Knotenstapel mit einem neuen AMI aktualisieren können. Sie können diese Anleitung verwenden, um Ihre Knoten nach einer Cluster-Aktualisierung auf eine neue Version von Kubernetes zu aktualisieren. Andernfalls können Sie auf das neueste von Amazon EKS optimierte AMI für eine vorhandene Kubernetes-Version aktualisieren.

> **Important**
>
> In diesem Thema werden Worker-Knotenaktualisierungen für selbstverwaltete Knotengruppen behandelt. Informationen zur Verwendung des Vereinfachten Knoten-

Lebenszyklus mit verwalteten Knotengruppen finden Sie unter the section called "Aktualisierung".

Die neueste Amazon AWS CloudFormation EKS-Standardknotenvorlage ist so konfiguriert, dass eine Instance mit dem neuen AMI in Ihrem Cluster gestartet wird, bevor nacheinander eine alte entfernt wird. Diese Konfiguration stellt sicher, dass Sie immer über die gewünschte Anzahl der aktiven Instances Ihre Auto-Scaling-Gruppe in Ihrem Cluster verfügen, während die Aktualisierung durchgeführt wird.

> **Note**
>
> Diese Methode wird nicht bei Knoten-Gruppen unterstützt, die mit `eksctl` erstellt wurden. Wenn Sie Ihr Cluster oder Ihre Worker-Knoten-Gruppe Sie `eksctl` erstellt haben, finden Sie Informationen unter the section called "Migration".

1. Bestimmen Sie den DNS-Anbieter Ihres Clusters.

   ```
   kubectl get deployments -l k8s-app=kube-dns -n kube-system
   ```

   Eine Beispielausgabe sieht wie folgt aus. Dieser Cluster verwendet CoreDNS für die DNS-Auflösung, aber Ihr Cluster kann stattdessen `kube-dns` zurückgeben. Ihre Ausgabe kann je nach verwendeter `kubectl`-Version anders aussehen.

   ```
   NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
   coredns   1         1         1            1           31m
   ```

2. Wenn Ihre aktuelle Bereitstellung weniger als zwei Replikate ausführt, skalieren die Bereitstellung auf zwei Replikate. Ersetzen Sie `kube-dns` durch *`coredns`*, falls Ihre vorherige Befehlsausgabe dies stattdessen zurückgegeben hat.

   ```
   kubectl scale deployments/coredns --replicas=2 -n kube-system
   ```

3. (Optional) Wenn Sie Kubernetes Cluster Autoscaler verwenden, skalieren Sie die Bereitstellung nach unten auf null (0) Replikate, um Konflikte zwischen Skalierungsaktionen zu vermeiden.

   ```
   kubectl scale deployments/cluster-autoscaler --replicas=0 -n kube-system
   ```

4. Bestimmen Sie den gewünschten Instance-Typ und die gewünschte Anzahl von Instances Ihrer aktuellen Knotengruppe. Sie geben diese Werte später ein, wenn Sie die AWS CloudFormation Vorlage für die Gruppe aktualisieren.
   a. Öffnen Sie die EC2 Amazon-Konsole unter https://console.aws.amazon.com/ec2/.
   b. Wählen Sie im linken Navigationsbereich Launch Configurations (Startkonfigurationen) aus und beachten Sie den Instance-Typ für die Startkonfiguration der vorhandenen Knoten.
   c. Wählen Sie im linken Navigationsbereich Auto Scaling Groups (Auto-Scaling-Gruppen) aus und beachten Sie die Desired (gewünschte) Instance-Anzahl für die Auto-Scaling-Gruppe der vorhandenen Knoten.
5. Öffnen Sie die AWS CloudFormation -Konsole.
6. Wählen Sie Ihren Workerknoten-Gruppen-Stack aus und klicken Sie dann auf Update (Aktualisieren).
7. Wählen Sie Replace current template (Aktuelle Vorlage ersetzen) und dann Amazon S3 URL (Amazon S3-URL) aus.
8. Fügen Sie für Amazon S3 S3-URL die folgende URL in den Textbereich ein, um sicherzustellen, dass Sie die neueste Version der AWS CloudFormation Node-Vorlage verwenden. Klicken Sie dann auf Next (Weiter):

   ```
   https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2022-12-23/amazon-eks-nodegroup.yaml
   ```

9. Geben Sie auf der Seite Specify stack details (Stack-Details angeben) die folgenden Parameter ein und wählen Sie Next (Weiter) aus:
   - NodeAutoScalingGroupDesiredCapacity— Geben Sie die gewünschte Anzahl von Instanzen ein, die Sie in einem vorherigen Schritt aufgezeichnet haben. Oder geben Sie die neue gewünschte Anzahl von Knoten ein, auf die bei der Aktualisierung Ihres Stacks skaliert werden soll.
   - NodeAutoScalingGroupMaxSize— Geben Sie die maximale Anzahl von Knoten ein, auf die Ihre Node-Auto-Scaling-Gruppe skalieren kann. Dieser Wert muss mindestens einen Knoten größer sein als Ihre gewünschte Kapazität. Auf diese Weise können Sie eine fortlaufende Aktualisierung Ihrer Knoten durchführen, ohne die Knotenanzahl während der Aktualisierung zu reduzieren.
   - NodeInstanceType— Wählen Sie den Instance-Typ, den Sie in einem vorherigen Schritt aufgezeichnet haben. Wählen Sie alternativ einen anderen Instance-Typ für Ihre Knoten aus. Bevor Sie einen anderen Instance-Typ auswählen, lesen Sie den Artikel Wählen Sie einen optimalen EC2 Amazon-Node-Instance-Typ. Jeder EC2 Amazon-Instance-Typ unterstützt eine maximale Anzahl von elastischen Netzwerkschnittstellen (Netzwerkschnittstelle) und jede

Netzwerkschnittstelle unterstützt eine maximale Anzahl von IP-Adressen. Da jedem Worker-Knoten und Pod eine eigene IP-Adresse zugewiesen wird, ist es wichtig, einen Instance-Typ zu wählen, der die maximale Anzahl von Pods unterstützt, die Sie auf jedem EC2 Amazon-Knoten ausführen möchten. Eine Liste der Netzwerkschnittstellen und IP-Adressen, die von Instance-Typen unterstützt werden, finden Sie unter IP-Adressen pro Netzwerkschnittstelle pro Instance-Typ. Der Instance–Typ `m5.large` unterstützt zum Beispiel maximal 30 IP-Adressen für den Worker-Knoten und die Pods.

> **Note**
>
> Die unterstützten Instance-Typen für die neueste Version des Amazon-VPC-CNI-Plugins für Kubernetes sind in vpc_ip_resource_limit.go auf GitHub aufgeführt. Möglicherweise müssen Sie Ihr Amazon-VPC-CNI-Plugin für die Kubernetes-Version aktualisieren, um die neuesten unterstützten Instance-Typen zu verwenden. Weitere Informationen finden Sie unter the section called “Amazon VPC CNI”.

> **Important**
>
> Einige Instance-Typen sind möglicherweise nicht in allen AWS Regionen verfügbar.

- NodeImageIdSSMParam— Der Amazon EC2 Systems Manager Manager-Parameter der AMI-ID, auf die Sie aktualisieren möchten. Der folgende Wert verwendet das neueste Amazon-EKS-optimierte AMI für Kubernetes-Version `1.33`.

```
/aws/service/eks/optimized-ami/1.33/amazon-linux-2/recommended/image_id
```

Sie können es *`1.33`* durch eine Plattformversion ersetzen, die identisch ist. Oder er sollte bis zu einer Version älter sein als die Kubernetes-Version, die auf Ihrer Steuerebene läuft. Es wird empfohlen, die Knoten auf der gleichen Version wie die Steuerungsebene zu halten. Sie können es auch *`amazon-linux-2`* durch einen anderen AMI-Typ ersetzen. Weitere Informationen finden Sie unter the section called “Holen Sie sich die neuesten IDs”.

> **Note**
>
> Mithilfe des Amazon EC2 Systems Manager Manager-Parameters können Sie Ihre Knoten in future aktualisieren, ohne eine AMI-ID suchen und angeben zu müssen. Wenn

Ihr AWS CloudFormation Stack diesen Wert verwendet, startet jedes Stack-Update immer das neueste empfohlene Amazon EKS-optimierte AMI für Ihre angegebene Kubernetes-Version. Dies ist auch dann der Fall, wenn Sie keine Werte in der Vorlage ändern.

- NodeImageId— Um Ihr eigenes benutzerdefiniertes AMI zu verwenden, geben Sie die ID ein, die das AMI verwenden soll.

  Important

  Dieser Wert überschreibt jeden Wert, der für NodeImageIdSSMParamangegeben wurde. Wenn Sie den NodeImageIdSSMParamWert verwenden möchten, stellen Sie sicher, dass der Wert für leer NodeImageIdist.

- Deaktivieren IMDSv1 — Standardmäßig unterstützt jeder Knoten den Instanz-Metadatendienst Version 1 (IMDSv1) und IMDSv2. Sie können ihn jedoch deaktivieren IMDSv1. Wählen Sie true aus, wenn Sie nicht möchten, dass Knoten oder Pods, die in der Knotengruppe geplant sind, verwendet IMDSv1 werden. Weitere Informationen finden Sie unter Konfiguration des Instance-Metadatenservice. Wenn Sie IAM-Rollen für Dienstkonten implementiert haben, weisen Sie allen Pods, die Zugriff auf AWS Dienste benötigen, die erforderlichen Berechtigungen direkt zu. Auf diese Weise benötigen keine Pods in Ihrem Cluster Zugriff auf IMDS aus anderen Gründen, z. B. zum Abrufen der aktuellen Region. AWS Anschließend können Sie den Zugriff auf auch IMDSv2 für Pods deaktivieren, die kein Host-Netzwerk verwenden. Weitere Informationen finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.

10. (Optional) Markieren Sie auf der Seite Options (Optionen) Ihre Stack-Ressourcen. Wählen Sie Weiter aus.
11. Überprüfen Sie Ihre Angaben auf der Seite Review (Überprüfen), bestätigen Sie, dass der Stack IAM-Ressourcen erstellen kann, und klicken Sie dann auf Update stack (Stack aktualisieren).

Note

Die Aktualisierung jedes Knotens im Cluster dauert mehrere Minuten. Warten Sie, bis die Aktualisierung aller Knoten abgeschlossen ist, bevor Sie die nächsten Schritte durchführen.

12. Wenn der DNS-Anbieter Ihres Clusters `kube-dns` ist, skalieren Sie die `kube-dns`-Bereitstellung auf ein Replikat.

```
kubectl scale deployments/kube-dns --replicas=1 -n kube-system
```

13. (Optional) Wenn Sie den Kubernetes-Cluster Autoscaler verwenden, skalieren Sie die Bereitstellung zurück auf die gewünschte Zahl von Replikaten.

```
kubectl scale deployments/cluster-autoscaler --replicas=1 -n kube-system
```

14. (Optional) Stellen Sie sicher, dass Sie die neueste Version des Amazon-VPC-CNI-Plugins für Kubernetes verwenden. Möglicherweise müssen Sie Ihr Amazon-VPC-CNI-Plugin für die Kubernetes-Version aktualisieren, um die neuesten unterstützten Instance-Typen zu verwenden. Weitere Informationen finden Sie unter the section called “Amazon VPC CNI”.

Vereinfachung der Rechenverwaltung mit AWS Fargate

In diesem Thema wird die Verwendung von Amazon EKS zum Ausführen von Kubernetes-Pods in AWS Fargate erläutert. Fargate ist eine Technologie, die bedarfsgerechte On-Demand-Rechenkapazität für Container bereitstellt. Mit Fargate müssen Sie keine Gruppen virtueller Maschinen selbst bereitstellen, konfigurieren oder skalieren, um Container auszuführen. Sie müssen auch keine Servertypen auswählen, entscheiden, wann Sie Ihre Knotengruppen skalieren oder die Cluster-Zusammenstellung optimieren.

Sie können steuern, welche Pods in Fargate starten und wie sie mit Fargate-Profilen ausgeführt werden. Fargate-Profile werden als Teil Ihres Amazon-EKS-Clusters definiert. Amazon EKS wird in Kubernetes mit Fargate integriert, indem Controller verwendet werden, die von AWS mithilfe des Upstream-Erweiterungsmodells von Kubernetes erstellt werden. Diese Controller werden als Teil der von Amazon EKS verwalteten Kubernetes-Steuerebene ausgeführt und sind für die Planung nativer Kubernetes-Pods in Fargate verantwortlich. Die Fargate-Controller enthalten einen neuen Scheduler, der neben dem Standard-Kubernetes-Scheduler und zusätzlich zu mehreren mutierenden und validierenden Zugangscontrollern ausgeführt wird. Wenn Sie einen Pod starten, der die Kriterien für die Ausführung in Fargate erfüllt, erkennen die Fargate-Controller, die im Cluster ausgeführt werden den Pod, und aktualisieren und planen ihn in Fargate.

In diesem Thema werden die verschiedenen Komponenten von Pods beschrieben, die in Fargate ausgeführt werden, und auf besondere Überlegungen für die Verwendung von Fargate mit Amazon EKS hingewiesen.

Überlegungen zu AWS-Fargate

Hier sind einige Dinge, die Sie bei der Verwendung von Fargate auf Amazon EKS beachten sollten.

- Jeder Pod, der in Fargate ausgeführt wird, verfügt über eine eigene Rechengrenze. Sie teilen sich weder den zugrunde liegenden Kernel noch CPU-Ressourcen, Speicherressourcen oder elastische Netzwerkschnittstellen mit anderen Pods.
- Network Load Balancer und Application Load Balancer (ALBs) können mit Fargate nur mit IP-Zielen verwendet werden. Weitere Informationen erhalten Sie unter the section called “Erstellen eines Network Load Balancers” und the section called “Application Load Balancing”.
- Fargate-exponierte Services werden nur im IP-Modus des Zieltyps und nicht im Knoten-IP-Modus ausgeführt. Die empfohlene Möglichkeit, die Konnektivität von einem Service zu überprüfen, der auf einem verwalteten Knoten ausgeführt wird, und einem Service, der auf Fargate ausgeführt wird, besteht darin, eine Verbindung über den Servicenamen herzustellen.
- Pods müssen zu dem Zeitpunkt, zu dem sie in Fargate ausgeführt werden sollen, mit einem Fargate-Profil übereinstimmen. Pods, die nicht mit einem Fargate-Profil übereinstimmen, können als `Pending` hängen bleiben. Wenn ein übereinstimmendes Fargate-Profil vorhanden ist, können Sie ausstehende Pods, die Sie erstellt haben, löschen, um sie in Fargate neu zu planen.
- Daemonsets werden von Fargate nicht unterstützt. Wenn Ihre Anwendung einen Daemon benötigt, konfigurieren Sie diesen Daemon neu, sodass er als Sidecar-Container in Ihren Pods ausgeführt wird.
- Privilegierte Container werden in Fargate nicht unterstützt.
- Pods, die in Fargate ausgeführt werden, können im Pod-Manifest kein `HostPort` oder `HostNetwork` angeben.
- Das standardmäßige weiche `nofile`- und `nproc`-Limit beträgt 1 024 und das harte Limit 65 535 für Fargate-Pods.
- GPUs sind derzeit in Fargate nicht verfügbar.
- Pods, die in Fargate ausgeführt werden, werden nur in privaten Subnetzen unterstützt (mit NAT Gateway-Zugriff auf AWS-Services, aber ohne direkte Route zu einem Internet-Gateway). Daher müssen für die VPC Ihres Clusters private Subnetze verfügbar sein. Weitere Informationen zu Clustern ohne ausgehenden Internetzugang finden Sie unter the section called “Private Cluster”.
- Sie können die Option Pod-Ressourcen mit Vertical Pod Autoscaler anpassen verwenden, um die anfängliche korrekte Größe von CPU und Arbeitsspeicher für Ihre Fargate-Pods festzulegen, und anschließend die Option Pod-Bereitstellungen mit Horizontal Pod Autoscaler skalieren

verwenden, um diese Pods zu skalieren. Wenn Sie möchten, dass der Vertical Pod Autoscaler Pods mit größeren CPU- und Speicherkombinationen automatisch erneut in Fargate bereitstellt, stellen Sie den Modus für den Vertical Pod Autoscaler entweder auf `Auto` oder `Recreate` ein, um die korrekte Funktionalität sicherzustellen. Weitere Informationen finden Sie in der Vertical Pod Autoscaler-Dokumentation auf GitHub.

- DNS-Auflösung und DNS-Hostnamen müssen für Ihre VPC aktiviert sein. Weitere Informationen finden Sie unter Anzeigen und Aktualisieren der DNS-Unterstützung für Ihre VPC.
- Amazon-EKS-Fargate erweitert die Verteidigung für Kubernetes-Anwendungen, indem jeder Pod innerhalb einer virtuellen Maschine (VM) isoliert wird. Diese VM-Grenze verhindert den Zugriff auf hostbasierte Ressourcen, die von anderen Pods im Falle eines Container-Escapes verwendet werden. Dies ist eine übliche Methode, um containerisierte Anwendungen anzugreifen und Zugriff auf Ressourcen außerhalb des Containers zu erhalten.

  Die Verwendung von Amazon EKS ändert nichts an Ihren Verantwortlichkeiten im Rahmen des Modells der geteilten Verantwortung. Sie sollten die Konfiguration von Cluster-Sicherheits- und Governance-Kontrollen sorgfältig prüfen. Der sicherste Weg, eine Anwendung zu isolieren, besteht immer darin, sie in einem separaten Cluster auszuführen.
- Fargate-Profile unterstützen die Angabe von Subnetzen aus sekundären VPC-CIDR-Blöcken. Möglicherweise möchten Sie einen sekundären CIDR-Block angeben. Das liegt daran, dass in einem Subnetz nur eine begrenzte Anzahl von IP-Adressen verfügbar ist. Daher gibt es auch eine begrenzte Anzahl von Pods, die im Cluster erstellt werden können. Durch die Verwendung verschiedener Subnetze für Pods können Sie die Anzahl der verfügbaren IP-Adressen erhöhen. Weitere Informationen finden Sie unter Hinzufügen von IPv4 CIDR-Blöcken zu einer VPC.
- Amazon EC2 Instance Metadata Service (IMDS) ist für Pods, die auf Fargate-Knoten bereitgestellt werden, nicht verfügbar. Wenn Sie Pods haben, die in Fargate bereitgestellt werden und IAM-Anmeldeinformationen erfordern, weisen Sie diese Ihren Pods mithilfe von IAM-Rollen für Servicekonten zu. Wenn Ihre Pods Zugriff auf andere Informationen benötigen, die über das IMDS verfügbar sind, müssen Sie diese Informationen in Ihre Pod-Spezifikation hart codieren. Dies umfasst die AWS-Region oder Availability Zone, in der ein Pod bereitgestellt wird.
- Sie können Fargate-Pods nicht in AWS Outposts, AWS Wavelength oder AWS Local Zones bereitstellen.
- Amazon EKS muss Fargate-Pods regelmäßig patchen, um deren Sicherheit zu gewährleisten. Wir versuchen, die Auswirkungen von Aktualisierungen möglichst gering zu halten. Es kann jedoch vorkommen, dass Pods gelöscht werden müssen, wenn sie nicht erfolgreich bereinigt wurden. Es gibt einige Maßnahmen, die Sie durchführen können, um Unterbrechungen zu minimieren. Weitere Informationen finden Sie unter the section called “Betriebssystem-Patching-Ereignisse”.

- Das Amazon-VPC-CNI-Plugin für Amazon EKS ist standardmäßig auf Fargate-Knoten installiert. Sie können keine Alternativen CNI-Plugins für Amazon-EKS-Cluster mit Fargate-Knoten verwenden.
- Ein in Fargate ausgeführter Pod bindet automatisch ein Amazon-EFS-Dateisystem ein, ohne dass manuelle Schritte zur Treiberinstallation erforderlich sind. Sie können mit Fargate-Knoten keine dynamische Bereitstellung persistenter Volumes verwenden, aber Sie können eine statische Bereitstellung verwenden.
- Amazon EKS unterstützt Fargate Spot nicht.
- Sie können Amazon-EBS-Volumes nicht in Fargate-Pods mounten.
- Sie können den Amazon-EBS-CSI-Controller in Fargate-Knoten ausführen, aber der Amazon-EBS-CSI-Knoten-DaemonSet kann nur in Amazon-EC2-Instances ausgeführt werden.
- Nachdem ein Kubernetes-Auftrag als `Completed` oder `Failed` gekennzeichnet wurde, existieren die vom Auftrag erstellten Pods normalerweise weiter. Durch dieses Verhalten können Sie die Protokolle und Ergebnisse anzeigen. Bei Fargate entstehen jedoch Kosten, wenn Sie den Auftrag nachher nicht bereinigen.

  Um die zugehörigen Pods automatisch zu löschen, nachdem ein Auftrag abgeschlossen wurde oder fehlgeschlagen ist, können Sie mithilfe des TTL-Controllers (Time-to-Live) einen Zeitraum angeben. Das folgende Beispiel veranschaulicht die Angabe von `.spec.ttlSecondsAfterFinished` im Auftrags-Manifest.

```
apiVersion: batch/v1
kind: Job
metadata:
  name: busybox
spec:
  template:
    spec:
      containers:
      - name: busybox
        image: busybox
        command: ["/bin/sh", "-c", "sleep 10"]
      restartPolicy: Never
  ttlSecondsAfterFinished: 60 # <-- TTL controller
```

Fargate-Vergleichstabelle

Kriterien	AWS-Fargate
Kann in AWS Outposts bereitgestellt werden	Nein
Kann auf eine AWS Lokale Zone bereitgestellt werden	Nein
Kann Container ausführen, die Windows benötigen	Nein
Kann Container ausführen, die Linux benötigen	Ja
Kann Workloads ausführen, die den Inferentia-Chip benötigen	Nein
Kann Workloads ausführen, die eine GPU benötigen	Nein
Kann Workloads ausführen, die Arm-Prozessoren benötigen	Nein
Kann ausgeführt werdenAWS Bottlerocket	Nein
Pods teilen eine Kernel-Laufzeitumgebung mit anderen Pods	Nein – Jeder Pod hat einen bestimmten Kernel
Pods teilen CPU, Arbeitsspeicher, Speicher und Netzwerkressourcen mit anderen Pods.	Nein – Jeder Pod hat bestimmte Ressourcen und kann unabhängig voneinander dimensioniert werden, um die Ressourcenauslastung zu maximieren.
Pods können mehr Hardware und Speicher verwenden, als in den Pod-Spezifikationen angefordert	Nein – Der Pod kann jedoch mit einer größeren vCPU und Speicherkonfiguration erneut bereitgestellt werden.
Bereitstellen und Verwalten von Amazon-EC2-Instances	Nein

Kriterien	AWS-Fargate
Das Betriebssystem von Amazon-EC2-Instanc es muss gesichert, gewartet und gepatcht werden	Nein
Kann Bootstrap-Argumente bei der Bereitste llung eines Knotens bereitstellen, z. B. zusätzlic he kubelet-Argumente.	Nein
IP-Adressen können Pods aus einem anderen CIDR-Block als der dem Knoten zugewiesenen IP-Adresse zuweisen.	Nein
SSH im Knoten nicht möglich	Nein – Es gibt kein Knoten-Host-Betrie bssystem für SSH.
Kann Ihr eigenes benutzerdefiniertes AMI auf Knoten bereitstellen	Nein
Kann Ihr eigenes benutzerdefiniertes CNI auf Knoten bereitstellen	Nein
Knoten-AMI muss selbst aktualisiert werden	Nein
Muss die Kubernetes-Version des Knotens selbst aktualisieren	Nein – Sie verwalten keine Knoten.
Kann Amazon-EBS-Speicher mit Pods verwenden	Nein
Kann Amazon-EFS-Speicher mit Pods verwenden	Ja
Kann Amazon FSx für Lustre Speicher mit Pods verwenden	Nein
Kann Network Load Balancer für Services verwenden	Ja, bei Verwendung der Option Network Load-Balancer erstellen

Kriterien	AWS-Fargate
Pods können in einem öffentlichen Subnetz ausgeführt werden	Nein
Kann einzelnen Pods verschiedene VPC-Sicherheitsgruppen zuweisen	Ja
Kann Kubernetes DaemonSets ausführen	Nein
Unterstützung für `HostPort` und `HostNetwork` im Pod-Manifest	Nein
AWS-Verfügbarkeit in Regionen	Einige von Amazon EKS unterstützte Regionen
Kann Container auf Amazon-EC2-Dedicated-Hosts ausführen	Nein
Preise	Kosten für eine individuelle Fargate-Speicher- und CPU-Konfiguration. Jeder Pod hat seine eigenen Kosten. Weitere Informationen dazu finden Sie unter AWS-Fargate-Preise.

Beginnen Sie mit AWS Fargate für Ihren Cluster

In diesem Thema werden die ersten Schritte zum Ausführen von Pods auf AWS Fargate mit Ihrem Amazon EKS-Cluster beschrieben.

Wenn Sie den Zugriff auf den öffentlichen Endpunkt Ihres Clusters mithilfe von CIDR-Blöcken einschränken, empfehlen wir, dass Sie auch den privaten Endpunktzugriff aktivieren. Auf diese Weise können Fargate-Pods mit dem Cluster kommunizieren. Wenn der private Endpunkt nicht aktiviert ist, müssen die CIDR-Blöcke, die Sie für den öffentlichen Zugriff angeben, die Ausgangsquellen aus Ihrer VPC enthalten. Weitere Informationen finden Sie unter the section called “Cluster-Endpunktzugriff”.

Voraussetzung

Einen vorhandenen -Cluster. Wenn Sie noch über keinen Amazon-EKS-Cluster verfügen, lesen Sie Erste Schritte.

Schritt 1: Sicherstellen, dass vorhandene Knoten mit Fargate-Pods kommunizieren können

Wenn Sie mit einem neuen Cluster ohne Knoten oder einem Cluster nur mit verwalteten Knotengruppen arbeiten (siehe the section called “Verwaltete Knotengruppen”), können Sie mit the section called “Schritt 2: Fargate-Pod-Ausführungsrolle erstellen” fortfahren.

Angenommen, Sie arbeiten mit einem vorhandenen Cluster, dem bereits Knoten zugeordnet sind. Stellen Sie sicher, dass die Pods auf diesen Knoten problemlos mit den Pods kommunizieren können, die in Fargate ausgeführt werden. Pods, die in Fargate ausgeführt werden, werden automatisch so konfiguriert, dass die Cluster-Sicherheitsgruppe für den Cluster verwendet wird, dem sie zugeordnet sind. Stellen Sie sicher, dass alle vorhandenen Knoten in Ihrem Cluster Datenverkehr an die Cluster-Sicherheitsgruppe senden und von dieser empfangen können. Verwaltete Knotengruppen werden automatisch so konfiguriert, dass sie auch die Cluster-Sicherheitsgruppe verwenden. Sie müssen sie also nicht ändern oder auf Kompatibilität prüfen (siehe the section called “Verwaltete Knotengruppen”).

Für bestehende Knotengruppen, die mit `eksctl` oder den von Amazon EKS verwalteten AWS CloudFormation Vorlagen erstellt wurden, können Sie die Cluster-Sicherheitsgruppe manuell zu den Knoten hinzufügen. Alternativ können Sie die Auto-Scaling-Gruppestartvorlage für die Knotengruppe ändern, um die Cluster-Sicherheitsgruppe an die Instances anzuhängen. Weitere Informationen finden Sie unter Ändern der Sicherheitsgruppen einer Instance im Amazon-VPC-Benutzerhandbuch.

Sie können im Abschnitt Netzwerk für den Cluster AWS-Managementkonsole nach einer Sicherheitsgruppe für Ihren Cluster suchen. Oder Sie können dies mit dem folgenden AWS CLI-Befehl tun. Wenn Sie diesen Befehl verwenden, ersetzen Sie `<my-cluster>` durch den Namen Ihres Clusters.

```
aws eks describe-cluster --name <my-cluster> --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId
```

Schritt 2: Fargate-Pod-Ausführungsrolle erstellen

Wenn Ihr Cluster Pods auf AWS Fargate erstellt, müssen die Komponenten, die auf der Fargate-Infrastruktur ausgeführt werden, in AWS APIs Ihrem Namen Aufrufe tätigen. Die Amazon-EKS-Pod-Ausführungsrolle stellt die entsprechenden IAM-Berechtigungen bereit. Informationen zum Erstellen einer AWS Fargate-Pod-Ausführungsrolle finden Sie unterthe section called “IAM-Rolle zur Pod-Ausführung”.

> **Note**
>
> Wenn Sie den Cluster mithilfe von `eksctl` und der `--fargate`-Option erstellt haben, verfügt der Cluster bereits über eine Pod-Ausführungsrolle, die Sie in der IAM-Konsole mit dem Muster `eksctl-my-cluster-FargatePodExecutionRole-ABCDEFGHIJKL` finden können. Wenn Sie Ihre Fargate-Profile mit `eksctl` anlegen, erstellt `eksctl` Ihre Pod-Ausführungsrolle, sofern noch keine erstellt wurde.

Schritt 3: Fargate Profil für Ihren Cluster erstellen

Bevor Sie Pods planen können, die in Fargate in Ihrem Cluster ausgeführt werden, müssen Sie ein Fargate-Profil definieren, das angibt, welche Pods Fargate beim Start verwenden soll. Weitere Informationen finden Sie unter the section called “Profile definieren”.

> **Note**
>
> Wenn Sie Ihren Cluster mithilfe von `eksctl` und der Option `--fargate` erstellt haben, wurde bereits ein Fargate-Profil für Ihren Cluster mit Selektoren für alle Pods in den `kube-system`- und `default`-Namespaces erstellt. Gehen Sie wie folgt vor, um Fargate-Profile für alle anderen Namespaces zu erstellen, die Sie mit Fargate verwenden möchten.

Sie können mit einem dieser Tools ein Fargate-Profil erstellen:

- the section called “`eksctl`”
- the section called “AWS-Managementkonsole”

`eksctl`

Für diesen Vorgang ist `eksctl` Version `0.215.0` oder höher erforderlich. Sie können Ihre -Version mit dem folgenden Befehl überprüfen:

```
eksctl version
```

Eine Installations- und Upgrade-Anleitung für `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.

So erstellen Sie ein Fargate-Profil mit **`eksctl`**

Erstellen Sie Ihr Fargate-Profil mit dem folgenden `eksctl`-Befehl und ersetzen Sie jede `<example value>` durch Ihre eigenen Werte. Sie müssen einen Namespace angeben. Die Option `--labels` ist jedoch nicht erforderlich.

```
eksctl create fargateprofile \
    --cluster <my-cluster> \
    --name <my-fargate-profile> \
    --namespace <my-kubernetes-namespace> \
    --labels <key=value>
```

Sie können bestimmte Platzhalter für `<my-kubernetes-namespace>`- und `<key=value>`-Labels verwenden. Weitere Informationen finden Sie unter the section called “Platzhalter für Fargate-Profile”.

AWS-Managementkonsole

So erstellen Sie ein Fargate-Profil mit AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie den Cluster aus, für den Sie ein Fargate-Profil erstellen möchten.
3. Wählen Sie die Registerkarte Compute (Datenverarbeitung) aus.
4. Wählen Sie unter Fargate-Profile die Option Fargate-Profil hinzufügen aus.
5. Auf der Seite Konfigurieren des Fargate Profils führen Sie folgende Schritte aus:
   a. Geben Sie unter Name einen Namen für Ihr Fargate-Profil ein. Der Name muss eindeutig sein.
   b. Wählen Sie für Pod-Ausführungsrolle die Pod-Ausführungsrolle aus, die mit Ihrem Fargate-Profil verwendet werden soll. Es werden nur IAM-Rollen mit dem `eks-fargate-pods.amazonaws.com`-Service-Prinzipal angezeigt. Wenn keine Rollen aufgelistet sind, müssen Sie eine erstellen. Weitere Informationen finden Sie unter the section called “IAM-Rolle zur Pod-Ausführung”.
   c. Ändern Sie die ausgewählten Subnetze nach Bedarf.

> **Note**
>
> Für Pods, die in Fargate ausgeführt werden, werden nur private Subnetze unterstützt.

d. Für Tags können Sie Ihr Fargate-Profil wahlweise markieren. Diese Tags werden nicht an andere Ressourcen weitergegeben, die dem Profil zugeordnet sind, z. B. Pods.

e. Wählen Sie Weiter aus.

6. Führen Sie auf der Seite Pod-Auswahl konfigurieren die folgenden Schritte aus:

a. Geben Sie für Namespace einen Namespace ein, der mit Pods übereinstimmt.

- Sie können bestimmte Namespaces für den Abgleich verwenden, z. B. `kube-system` oder `default`.
- Sie können bestimmte Platzhalter verwenden (z. B. `prod-*`), um mehrere Namespaces abzugleichen (z. B. `prod-deployment` und `prod-test`). Weitere Informationen finden Sie unter the section called “Platzhalter für Fargate-Profile ”.

b. (Optional) Fügen Sie dem Selektor Kubernetes-Markierungen hinzu. Fügen Sie sie insbesondere demjenigen hinzu, dem die Pods im angegebenen Namespace entsprechen müssen.

- Sie können dem Selektor das Label `infrastructure: fargate` hinzufügen, sodass nur Pods im angegebenen Namespace, die ebenfalls die `infrastructure: fargate`-Kubernetes-Bezeichnung tragen, mit dem Selektor übereinstimmen.
- Sie können bestimmte Platzhalter verwenden (z. B. `key?: value?`), um mehrere Namespaces abzugleichen (z. B. `keya: valuea` und `keyb: valueb`). Weitere Informationen finden Sie unter the section called “Platzhalter für Fargate-Profile ”.

c. Wählen Sie Weiter aus.

7. Überprüfen Sie auf der Seite Überprüfen und erstellen die Informationen für Ihr -Profil und wählen Sie Erstellen aus.

Schritt 4: CoreDNS aktualisieren

Standardmäßig ist CoreDNS so konfiguriert, dass es auf der EC2 Amazon-Infrastruktur auf Amazon EKS-Clustern ausgeführt wird. Wenn Sie Ihre Pods nur in Fargate in Ihrem Cluster ausführen möchten, führen Sie die folgenden Schritte aus.

> **Note**
>
> Wenn Sie einen Cluster mit `eksctl` unter Verwendung von `--fargate`-Option erstellt haben, können Sie zu the section called “Nächste Schritte” springen.

1. Erstellen Sie jedes Fargate-Profil für CoreDNS mit dem folgenden Befehl. <my-cluster>Ersetzen Sie es durch Ihren Clusternamen, <111122223333> durch Ihre Konto-ID, <AmazonEKSFargatePodExecutionRole> durch den Namen Ihrer Pod-Ausführungsrolle und <000000000000000a><000000000000000b>, und <000000000000000c> durch die IDs Ihrer privaten Subnetze. Wenn Sie über keine Pod-Ausführungsrolle verfügen, müssen Sie zuerst eine erstellen (siehe the section called “Schritt 2: Fargate-Pod-Ausführungsrolle erstellen”).

   > **Important**
   >
   > Der Rollen-ARN darf als Pfad nur / enthalten. Wenn der Name Ihrer Rolle also beispielsweise development/apps/AmazonEKSFargatePodExecutionRole lautet, müssen Sie ihn beim Angeben des ARN für die Rolle in AmazonEKSFargatePodExecutionRole ändern. Das Format des Rollen-ARN muss arn:aws: iam::<111122223333>:role/<AmazonEKSFargatePodExecutionRole> sein.

```
aws eks create-fargate-profile \
    --fargate-profile-name coredns \
    --cluster-name <my-cluster> \
    --pod-execution-role-arn arn:aws: iam::<111122223333>:role/
<AmazonEKSFargatePodExecutionRole> \
    --selectors namespace=kube-system,labels={k8s-app=kube-dns} \
    --subnets subnet-<000000000000000a> subnet-<000000000000000b> subnet-
<000000000000000c>
```

2. Lösen Sie die Einführung der coredns-Bereitstellung aus.

```
kubectl rollout restart -n kube-system deployment coredns
```

Nächste Schritte

- Mit dem folgenden Workflow können Sie mit der Migration Ihrer vorhandenen Anwendungen zum Ausführen in Fargate beginnen.
  a. the section called “Erstellen eines Fargate-Profils”, das mit dem Kubernetes-Namespace und den Kubernetes-Labels Ihrer Anwendung übereinstimmt.

b. Löschen Sie alle vorhandenen Pods und erstellen Sie sie neu, damit sie auf Fargate geplant werden. Ändern Sie `<namespace>` und `<deployment-type>`, um Ihre spezifischen Pods zu aktualisieren.

```
kubectl rollout restart -n <namespace> deployment <deployment-type>
```

- Stellen Sie das the section called “Application Load Balancing” bereit, um Eingangsobjekte für Ihre Pods zuzulassen, die auf Fargate ausgeführt werden.
- Sie können the section called “Vertical Pod Autoscaler” verwenden, um die CPU und den Speicher für Ihre Fargate- anfänglich richtig zu dimensionieren, und dann Fargate Pods verwenden, um the section called “Horizontal Pod Autoscaler” diese Pods zu skalieren. Wenn Sie möchten, dass der Vertical Pod Autoscaler `Auto` mit höheren CPU- und Speicherkombinationen automatisch erneut in Fargate bereitstellt, stellen Sie den Modus für den Vertical-Pod-Autoscaler-Modus entweder auf oder `Recreate`. Dies dient der Gewährleistung einer korrekten Funktion. Weitere Informationen finden Sie in der Vertical Pod Autoscaler-Dokumentation auf GitHub.
- Sie können den AWS Distro für OpenTelemetry(ADOT)-Kollektor zur Anwendungsüberwachung durch Befolgen dieser Anweisungen einrichten.

Festlegung, welche Pods beim Start AWS Fargate verwenden

Bevor Sie Pods auf Fargate in Ihrem Cluster planen, müssen Sie mindestens ein Fargate-Profil definieren, das angibt, welche Pods Fargate beim Start verwenden.

Als Administrator können Sie mithilfe eines Fargate-Profils angeben, welche Pods in Fargate ausgeführt werden. Sie können dies über die Selektoren des Profils tun. Sie können bis zu fünf Selektoren zu jedem Profil hinzufügen. Jeder Selektor muss einen Namespace enthalten. Der Selector kann auch Labels enthalten. Das Label-Feld besteht aus mehreren optionalen Schlüssel-Wert-Paaren. Pods, die den Selektoren entsprechen, werden auf Fargate geplant. Pods werden mit einem Namespace und den Labels abgeglichen, die im Selektor angegeben sind. Wenn ein Namespace-Selektor ohne Labels definiert ist, versucht Amazon EKS, alle Pods, die in diesem Namespace ausgeführt werden, mithilfe des Profils in Fargate zu planen. Wenn ein zu planender Pod mit einem der Selektoren im Fargate-Profil übereinstimmt, wird dieser Pod in Fargate geplant.

Wenn ein Pod mit mehreren Fargate-Profilen übereinstimmt, können Sie angeben, welches Profil ein Pod verwendet, indem Sie der Pod-Spezifikation das folgende Kubernetes-Label hinzufügen: `eks.amazonaws.com/fargate-profile: my-fargate-profile`. Der Pod muss mit einem Selektor in diesem Profil übereinstimmen, um in Fargate geplant zu werden. Die Affinitäts-/

Anti-Affinitätsregeln von Kubernetes gelten nicht und sind bei Amazon-EKS-Fargate-Pods nicht erforderlich.

Wenn Sie ein Fargate-Profil erstellen, müssen Sie eine Pod-Ausführungsrolle angeben. Diese Ausführungsrolle ist für die Amazon-EKS-Komponenten vorgesehen, die auf der Fargate-Infrastruktur mit dem Profil ausgeführt werden. Es wird zur Autorisierung der Rollenbasierten Zugriffskontrolle (RBAC) von Kubernetes des Clusters hinzugefügt. Auf diese Weise kann sich `kubelet`, das in der Fargate-Infrastruktur ausgeführt wird, bei Ihrem Amazon-EKS-Cluster registrieren und als Knoten in Ihrem Cluster angezeigt werden. Die Pod-Ausführungsrolle bietet auch IAM-Berechtigungen für die Fargate-Infrastruktur, um Lesezugriff auf Amazon-ECR-Image-Repositorys zu ermöglichen. Weitere Informationen finden Sie unter the section called “IAM-Rolle zur Pod-Ausführung”.

Fargate-Profile können nicht geändert werden. Sie können jedoch ein neues aktuelles Profil erstellen, um ein vorhandenes Profil zu ersetzen, und dann das Original löschen.

> **Note**
>
> Alle Pods, die mit einem Fargate-Profil ausgeführt werden, werden angehalten und als ausstehend gekennzeichnet, wenn das Profil gelöscht wird.

Wenn Fargate-Profile in einem Cluster den Status `DELETING` haben, müssen Sie warten, bis dieses Fargate-Profil gelöscht wurde, bevor Sie andere Profile in diesem Cluster erstellen können.

> **Note**
>
> Fargate unterstützt derzeit keine Kubernetes topologySpreadConstraints.

Amazon EKS und Fargate verteilen Pods auf alle Subnetze, die im Fargate-Profil definiert sind. Es kann jedoch zu einer ungleichmäßigen Verteilung kommen. Wenn Sie eine gleichmäßige Verteilung benötigen, verwenden Sie zwei Fargate-Profile. Eine gleichmäßige Verteilung ist in Szenarien wichtig, in denen Sie zwei Replikate bereitstellen möchten und keine Ausfallzeiten wünschen. Wir empfehlen, dass jedes Profil nur ein Subnetz hat.

Fargate-Profilkomponenten

Die folgenden Komponenten sind in einem Fargate-Profil enthalten.

Pod-Ausführungsrolle

Wenn Ihr Cluster Pods auf AWS Fargate erstellt, muss das auf der Fargate-Infrastruktur ausgeführte `kubelet` in Ihrem Namen AWS-API-Aufrufe durchführen. Beispielsweise muss er Aufrufe tätigen, um Container-Images aus Amazon ECR zu ziehen. Die Amazon-EKS-Pod-Ausführungsrolle stellt die entsprechenden IAM-Berechtigungen bereit.

Wenn Sie ein Fargate-Profil erstellen, müssen Sie eine Pod-Ausführungsrolle angeben, die mit Ihren Pods verwendet werden soll. Diese Rolle wird zur Autorisierung der Rollenbasierten Zugriffskontrolle (RBAC) von Kubernetes des Clusters hinzugefügt. Auf diese Weise kann sich `kubelet`, das in der Fargate-Infrastruktur ausgeführt wird, bei Ihrem Amazon-EKS-Cluster registrieren und als Knoten in Ihrem Cluster angezeigt werden. Weitere Informationen finden Sie unter the section called “IAM-Rolle zur Pod-Ausführung”.

Subnets

Die IDs von Subnetzen, in denen Pods gestartet werden, die dieses Profil verwenden. Derzeit werden Pods, die in Fargate ausgeführt werden, keine öffentlichen IP-Adressen zugewiesen. Daher werden für diesen Parameter nur private Subnetze ohne direkte Route zu einem Internet-Gateway akzeptiert.

Selektoren

Die Selektoren, mit denen Pods, die dieses Fargate-Profil verwenden, übereinstimmen sollen. Sie können bis zu fünf Selektoren in einem Fargate-Profil angeben. Die Selektoren bestehen aus folgenden Komponenten:

- Namespace – Sie müssen einen Namespace für einen Selektor angeben. Der Selektor stimmt nur mit Pods überein, die in diesem Namespace erstellt werden. Sie können jedoch mehrere Selektoren erstellen, um mehrere Namespaces zu adressieren.
- Bezeichnungen – Sie können optional Kubernetes-Bezeichnungen angeben, mit denen der Selektor übereinstimmen muss. Der Selektor stimmt nur mit Pods überein, die alle im Selektor angegebenen Labels aufweisen.

Platzhalter für Fargate-Profile

Zusätzlich zu den Zeichen, die von Kubernetes erlaubt sind, dürfen Sie `*` und `?` in den Selektorkriterien für Namespaces, Label-Schlüssel und Label-Werte verwenden:

- `*` steht für kein, ein oder mehrere Zeichen. Zum Beispiel kann `prod*` `prod` und `prod-metrics` darstellen.

- ? steht für ein einzelnes Zeichen (z. B. kann `value?` `valuea` darstellen). Es kann jedoch nicht `value` und `value-a` darstellen, weil ? nur genau ein Zeichen darstellen kann.

Diese Platzhalterzeichen können an jeder Position und in Kombination verwendet werden (z. B. `prod*`, `*dev` und `frontend*?`). Andere Platzhalter und Formen des Mustervergleichs, wie reguläre Ausdrücke, werden nicht unterstützt.

Wenn mehrere übereinstimmende Profile für den Namespace und die Labels in der Pod-Spezifikation vorhanden sind, übernimmt Fargate das Profil basierend auf einer alphanumerischen Sortierung nach Profilnamen. Wenn zum Beispiel Profil A (mit dem Namen `beta-workload`) und Profil B (mit dem Namen `prod-workload`) passende Selektoren für die zu startenden Pods haben, wählt Fargate Profil A (`beta-workload`) für die Pods aus. Die Pods haben Labels mit Profil A auf den Pods (z. B. `eks.amazonaws.com/fargate-profile=beta-workload`).

Wenn Sie vorhandene Fargate-Pods auf neue Profile migrieren möchten, die Platzhalter verwenden, gibt es zwei Möglichkeiten, dies zu tun:

- Erstellen Sie ein neues Profil mit passenden Selektoren und löschen Sie dann die alten Profile. Mit alten Profilen gekennzeichnete Pods werden in neue übereinstimmende Profile verschoben.
- Wenn Sie Workloads migrieren möchten, sich aber nicht sicher sind, welche Fargate-Labels sich auf jedem Fargate-Pod befinden, können Sie die folgende Methode verwenden. Erstellen Sie ein neues Profil mit einem Namen, das zuerst alphanumerisch unter den Profilen auf demselben Cluster sortiert. Recyceln Sie dann die Fargate-Pods, die in neue Profile migriert werden müssen.

Erstellen eines Fargate-Profils

In diesem Abschnitt wird beschrieben, wie Sie ein Fargate-Profil erstellen. Sie müssen auch eine Pod-Ausführungsrolle erstellt haben, die für Ihr Fargate-Profil verwendet werden soll. Weitere Informationen finden Sie unter the section called “IAM-Rolle zur Pod-Ausführung”. Pods, die in Fargate ausgeführt werden, werden nur in privaten Subnetzen unterstützt (mit NAT Gateway-Zugriff auf AWS-Services, aber ohne direkte Route zu einem Internet-Gateway. Dies ist so, da in der VPC Ihres Clusters private Subnetze verfügbar sein müssen.

Sie können ein Profil mit Folgendem erstellen:

- the section called “`eksctl`”
- the section called “AWS-Managementkonsole”

`eksctl`

So erstellen Sie ein Fargate-Profil mit **`eksctl`**

Erstellen Sie Ihr Fargate-Profil mit dem folgenden `eksctl`-Befehl und ersetzen Sie dabei alle Beispielwerte durch Ihre eigenen Werte. Sie müssen einen Namespace angeben. Die Option `--labels` ist jedoch nicht erforderlich.

```
eksctl create fargateprofile \
    --cluster my-cluster \
    --name my-fargate-profile \
    --namespace my-kubernetes-namespace \
    --labels key=value
```

Sie können bestimmte Platzhalter für `my-kubernetes-namespace`- und `key=value`-Labels verwenden. Weitere Informationen finden Sie unter the section called “Platzhalter für Fargate-Profile”.

AWS-Managementkonsole

So erstellen Sie ein Fargate-Profil mit AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie den Cluster aus, für den Sie ein Fargate-Profil erstellen möchten.
3. Wählen Sie die Registerkarte Compute (Datenverarbeitung) aus.
4. Wählen Sie unter Fargate-Profile die Option Fargate-Profil hinzufügen aus.
5. Auf der Seite Konfigurieren des Fargate Profils führen Sie folgende Schritte aus:
   a. Geben Sie unter Name einen eindeutigen Namen für Ihr Fargate-Profil ein, wie z. B. `my-profile`.
   b. Wählen Sie für Pod-Ausführungsrolle die Pod-Ausführungsrolle aus, die mit Ihrem Fargate-Profil verwendet werden soll. Es werden nur IAM-Rollen mit dem `eks-fargate-pods.amazonaws.com`-Service-Prinzipal angezeigt. Wenn keine Rollen aufgelistet sind, müssen Sie eine erstellen. Weitere Informationen finden Sie unter the section called “IAM-Rolle zur Pod-Ausführung”.
   c. Ändern Sie die ausgewählten Subnetze nach Bedarf.

> **Note**
>
> Für Pods, die in Fargate ausgeführt werden, werden nur private Subnetze unterstützt.

d. Für Tags können Sie Ihr Fargate-Profil wahlweise markieren. Diese Tags werden nicht an andere Ressourcen weitergegeben, die dem Profil zugeordnet sind, z. B. Pods.

e. Wählen Sie Weiter aus.

6. Führen Sie auf der Seite Pod-Auswahl konfigurieren die folgenden Schritte aus:

   a. Geben Sie für Namespace einen Namespace ein, der mit Pods übereinstimmt.

      - Sie können bestimmte Namespaces für den Abgleich verwenden, z. B. `kube-system` oder `default`.
      - Sie können bestimmte Platzhalter verwenden (z. B. `prod-*`), um mehrere Namespaces abzugleichen (z. B. `prod-deployment` und `prod-test`). Weitere Informationen finden Sie unter the section called “Platzhalter für Fargate-Profile ”.

   b. (Optional) Fügen Sie dem Selektor Kubernetes-Markierungen hinzu. Fügen Sie sie insbesondere demjenigen hinzu, dem die Pods im angegebenen Namespace entsprechen müssen.

      - Sie können dem Selektor das Label `infrastructure: fargate` hinzufügen, sodass nur Pods im angegebenen Namespace, die ebenfalls die `infrastructure: fargate`-Kubernetes-Bezeichnung tragen, mit dem Selektor übereinstimmen.
      - Sie können bestimmte Platzhalter verwenden (z. B. `key?: value?`), um mehrere Namespaces abzugleichen (z. B. `keya: valuea` und `keyb: valueb`). Weitere Informationen finden Sie unter the section called “Platzhalter für Fargate-Profile ”.

   c. Wählen Sie Weiter aus.

7. Überprüfen Sie auf der Seite Überprüfen und erstellen die Informationen für Ihr -Profil und wählen Sie Erstellen aus.

Fargate-Profil löschen

In diesem Thema wird beschrieben, wie Sie ein Fargate-Profil löschen. Wenn Sie ein Fargate-Profil löschen, werden alle Pods gelöscht, die in Fargate mit dem Profil geplant wurden. Wenn diese Pods mit einem anderen Fargate-Profil übereinstimmen, werden sie mit diesem Profil in Fargate geplant. Wenn sie nicht mehr mit einem der Fargate-Profile übereinstimmen, werden sie nicht in Fargate geplant und ggf. als ausstehend gekennzeichnet.

Nur ein Fargate-Profil in einem Cluster kann sich jeweils im `DELETING`-Status befinden. Warten Sie, bis das Löschen eines Fargate-Profils abgeschlossen ist, bevor Sie andere Profile in diesem Cluster löschen können.

Sie können ein Profil mit einem der folgenden Tools löschen:

- the section called “`eksctl`”
- the section called “AWS-Managementkonsole”
- the section called “AWS-CLI”

`eksctl`

Löschen eines Fargate-Profils mit **`eksctl`**

Verwenden Sie den folgenden Befehl, um ein Profil aus einem Cluster zu löschen. Ersetzen Sie jeden *`Beispielwert`* durch Ihre eigenen Werte.

```
eksctl delete fargateprofile  --name my-profile --cluster my-cluster
```

AWS-Managementkonsole

Löschen eines Fargate-Profils mit AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich die Option Cluster aus. Wählen Sie in der Clusterliste den Cluster aus, in dem Sie das Fargate-Profil löschen möchten.
3. Wählen Sie die Registerkarte Compute (Datenverarbeitung) aus.
4. Wählen Sie das zu löschende Fargate-Profil und dann Delete (Löschen) aus.
5. Geben Sie auf der Seite Delete Fargate profile (Fargate-Profil löschen) den Namen des Profils ein und wählen Sie dann Delete (Löschen) aus.

AWS-CLI

Löschen eines Fargate-Profils mit der AWS-CLI

Verwenden Sie den folgenden Befehl, um ein Profil aus einem Cluster zu löschen. Ersetzen Sie jeden *`Beispielwert`* durch Ihre eigenen Werte.

```
aws eks delete-fargate-profile --fargate-profile-name my-profile --cluster-name my-
cluster
```

Details zur Fargate-Pod-Konfiguration verstehen

In diesem Abschnitt werden einige der eindeutigen Pod-Konfigurationsdetails für die Ausführung von Kubernetes-Pods in AWS-Fargate beschrieben.

CPU und Arbeitsspeicher des Pods

Mit Kubernetes können Sie Anforderungen, eine minimale vCPU-Menge und Arbeitsspeicherressourcen definieren, die jedem Container in einem Pod zugewiesen werden. Pods werden von Kubernetes geplant, um sicherzustellen, dass mindestens die angeforderten Ressourcen für jeden Pod in der Rechenressource verfügbar sind. Weitere Informationen finden Sie unter Managing Compute Resources for Containers in der Kubernetes-Dokumentation.

> **Note**
>
> Da Amazon-EKS-Fargate nur einen Pod pro Knoten ausführt, tritt das Szenario des Räumens von Pods bei weniger Ressourcen nicht auf. Alle Amazon-EKS-Fargate-Pods werden mit garantierter Priorität ausgeführt, daher müssen die angeforderte CPU-Leistung und der Arbeitsspeicher für alle Container dem Limit entsprechen. Weitere Informationen finden Sie unter Konfigurieren von Dienstqualität für Pods in der Kubernetes-Dokumentation.

Wenn Pods in Fargate geplant sind, bestimmen die vCPU- und Speicherreservierungen innerhalb der Pod-Spezifikation, wie viel CPU und Speicher für den Pod bereitgestellt werden sollen.

- Die maximale Anforderung von Init-Containern wird verwendet, um die vCPU- und Speicheranforderungen für die Init-Anforderung zu bestimmen.
- Anforderungen für alle lang laufenden Container werden addiert, um die Anforderungen an die vCPU und den Arbeitsspeicher für lange laufende Anforderungen zu bestimmen.
- Der größere der beiden zuvor genannten Werte wird für die vCPU und die Speicheranforderung für Ihren Pod ausgewählt.
- Fargate fügt 256 MB zur Speicherreservierung jedes Pods für die erforderlichen Kubernetes-Komponenten (`kubelet`, `kube-proxy` und `containerd`) hinzu.

Fargate rundet auf die folgende Rechenkonfiguration auf, die der Summe von vCPU und Speicheranforderungen am nächsten liegt, damit Pods immer über die Ressourcen verfügen, die sie für ihre Ausführung benötigen.

Wenn Sie keine vCPU- und Speicherkombination angeben, wird die kleinste verfügbare Kombination verwendet (0,25 vCPU und 0,5 GB Arbeitsspeicher).

Die folgende Tabelle zeigt die vCPU- und Speicherkombinationen, die für Pods verfügbar sind, die in Fargate ausgeführt werden.

vCPU-Wert	Speicherwert
0,25 vCPU	0,5 GB, 1 GB, 2 GB
0,5 vCPU	1 GB, 2 GB, 3 GB, 4 GB
1 vCPU	2 GB, 3 GB, 4 GB, 5 GB, 6 GB, 7 GB, 8 GB
2 vCPU	Zwischen 4 GB und 16 GB in 1-GB-Schritten
4 vCPU	Zwischen 8 GB und 30 GB in 1-GB-Schritten
8 vCPU	Zwischen 16 GB und 60 GB in 4-GB-Schritten
16 vCPU	Zwischen 32 GB und 120 GB in 8-GB-Schritten

Der für die Kubernetes-Komponenten reservierte zusätzliche Arbeitsspeicher kann dazu führen, dass eine Fargate-Aufgabe mit mehr vCPUs als angefordert bereitgestellt wird. Bei einer Anforderung von 1 vCPU und 8 GB Speicher werden beispielsweise 256 MB zu ihrer Speicheranforderung hinzugefügt und eine Fargate-Aufgabe mit 2 vCPUs und 9 GB Speicher bereitgestellt, da keine Aufgabe mit 1 vCPU und 9 GB Speicher verfügbar ist.

Es gibt keine Korrelation zwischen der Größe des Pods, der auf Fargate ausgeführt wird, und der von Kubernetes gemeldeten Knotengröße mit `kubectl get nodes`. Die gemeldete Knotengröße ist oft größer als die Kapazität des Pods. Sie können die Pod-Kapazität mit dem folgenden Befehl überprüfen. Ersetzen Sie *`default`* durch den Namespace Ihres Pods und *`pod-name`* durch den Namen Ihres Pods

```
kubectl describe pod --namespace default pod-name
```

Eine Beispielausgabe sieht wie folgt aus.

```
[...]
annotations:
    CapacityProvisioned: 0.25vCPU 0.5GB
[...]
```

Die `CapacityProvisioned`-Annotation stellt die erzwungene Pod-Kapazität dar und bestimmt die Kosten Ihres Pods, der auf Fargate ausgeführt wird. Preisinformationen für die Computing-Konfigurationen finden Sie unter AWS-Fargate-Preise.

Fargate-Speicher

Ein in Fargate ausgeführter Pod bindet automatisch ein Amazon-EFS-Dateisystem ein, ohne dass manuelle Schritte zur Treiberinstallation erforderlich sind. Sie können mit Fargate-Knoten keine dynamische Bereitstellung persistenter Volumes verwenden, aber Sie können eine statische Bereitstellung verwenden. Weitere Informationen finden Sie unter Amazon-EFS-CSI-Treiber in GitHub.

Bei der Bereitstellung erhält jedes Pod, die in Fargate ausgeführt wird, standardmäßig 20 GiB flüchtigen Speicher. Diese Art von Speicher wird gelöscht, nachdem ein Pod stoppt. Bei neu auf Fargate gestarteten Pods ist die Verschlüsselung des flüchtigen Speicher-Volumes standardmäßig aktiviert. Der flüchtige Pod-Speicher wird mit einem AES-256-Verschlüsselungsalgorithmus mit AWS-Fargate-verwalteten Schlüsseln verschlüsselt.

> **Note**
>
> Der standardmäßig nutzbare Speicher für Amazon-EKS-Pods, die in Fargate ausgeführt werden, beträgt weniger als 20 GiB. Dies liegt daran, dass ein Teil des Speicherplatzes von `kubelet` und anderen Kubernetes-Modulen belegt wird, die im Pod geladen werden.

Sie können die Gesamtmenge des flüchtigen Speichers bis zu einem Maximum von 175 GB erhöhen. Um die Größe mit Kubernetes zu konfigurieren, geben Sie die `ephemeral-storage`-Ressourcen-Anfragen für jeden Container in einem Pod an. Wenn Kubernetes Pods plant, stellt es sicher, dass die Summe der Ressourcen-Anfragen für jeden Pod geringer ist als die Kapazität der Fargate-Aufgabe. Weitere Informationen finden Sie unter Ressourcen-Management für Pods und Container in der Kubernetes-Dokumentation.

Amazon EKS Fargate stellt mehr kurzlebigen Speicher bereit, als für die Systemnutzung angefordert wurde. Eine Anforderung von 100 GB stellt beispielsweise eine Fargate-Aufgabe mit 115 GB flüchtigem Speicher bereit.

Festlegung von Maßnahmen für Betriebssystem-Patching-Ereignisse für AWS Fargate

Amazon EKS patcht regelmäßig das Betriebssystem für AWS-Fargate-Knoten, damit sie sicher bleiben. Im Rahmen des Patching-Prozesses werden die Knoten recycelt, um Betriebssystem-Patches zu installieren. Updates werden so durchgeführt, dass sie Ihre Services möglichst wenig beeinflussen. Wenn Pods jedoch nicht erfolgreich bereinigt wurden, kann es sein, dass sie gelöscht werden müssen. Die folgenden Maßnahmen können Ihnen helfen, potenzielle Unterbrechungen zu minimieren:

- Legen Sie geeignete Budgets für Pod-Unterbrechung (PBDs) fest, um die Anzahl der Pods, die gleichzeitig ausfallen, zu steuern.
- Erstellen Sie Amazon-EventBridge-Regeln für den Umgang mit fehlgeschlagenen Bereinigungen, bevor die Pods gelöscht werden.
- Starten Sie die betroffenen Pods manuell neu, bevor das in der Benachrichtigung angegebene Bereinigungsdatum erreicht ist.
- Erstellen Sie eine Benachrichtigungskonfiguration in AWS-Benutzerbenachrichtigungen.

Amazon EKS arbeitet eng mit der Kubernetes-Community zusammen, um Fehlerbehebungen und Sicherheits-Patches so schnell wie möglich bereitzustellen. Alle Fargate-Pods starten mit der aktuellsten Kubernetes-Patch-Version, die über die über Amazon EKS für die Kubernetes-Version Ihres Clusters verfügbar ist. Wenn Sie einen Pod mit einer älteren Patch-Version haben, recycelt Amazon EKS ihn möglicherweise, um ihn auf die neueste Version zu aktualisieren. Das stellt sicher, dass Ihre Pods mit den neuesten Sicherheitsaktualisierungen ausgestattet sind. Wenn ein Häufige Schwachstellen und Aufdeckungen (CVE)-Problem auftritt, bleiben Sie so auf dem Laufenden und können Sicherheitsrisiken reduzieren.

Wenn das AWS-Fargate-Betriebssystem aktualisiert wird, sendet Ihnen Amazon EKS eine Benachrichtigung, die Ihre betroffenen Ressourcen und das Datum der bevorstehenden Pod-Bereinigungen enthält. Wenn das angegebene Bereinigungsdatum ungünstig ist, haben Sie die Möglichkeit, Ihre betroffenen Pods vor dem in der Benachrichtigung angegebenen Bereinigungsdatum manuell neu zu starten. Alle Pods, die vor dem Zeitpunkt der Benachrichtigung

erstellt wurden, können bereinigt werden. Weitere Anweisungen zum manuellen Neustart Ihrer Pods finden Sie in der Kubernetes-Dokumentation.

Um die Anzahl der Pods zu begrenzen, die bei der Wiederverwendung von Pods gleichzeitig ausfallen, können Sie Budgets für Pod-Unterbrechung (PDBs) festlegen. Sie können PDBs verwenden, um die Mindestverfügbarkeit basierend auf den Anforderungen jeder Ihrer Anwendungen zu definieren und gleichzeitig Aktualisierungen zuzulassen. Die Mindestverfügbarkeit Ihres PDB muss unter 100 % liegen. Weitere Informationen finden Sie unter Angabe eines Budgets für Unterbrechung in Ihrer Anwendung in der Kubernetes-Dokumentation.

Amazon EKS verwendet die Bereinigungs-AMI, um den Pod sicher zu leeren. Dabei werden die PDBs eingehalten, die Sie für die Anwendung festgelegt haben. Pods werden von der Availability Zone bereinigt, um Auswirkungen zu minimieren. Wenn die Bereinigung erfolgreich ist, erhält der neue Pod den aktuellsten Patch und es sind keine weiteren Maßnahmen erforderlich.

Wenn die Bereinigung für einen Pod fehlschlägt, sendet Amazon EKS ein Ereignis an Ihr Konto, das Details über die Pods enthält, deren Bereinigung fehlgeschlagen ist. Sie können vor der geplanten Beendigungszeit auf die Nachricht reagieren. Die spezifische Zeit variiert je nach Dringlichkeit des Patches. Zum festgelegten Zeitpunkt versucht Amazon EKS erneut, die Pods zu bereinigen. Dieses Mal wird bei einer fehlgeschlagenen Bereinigung jedoch kein neues Ereignis gesendet. Wenn die Bereinigung erneut fehlschlägt, werden Ihre vorhandenen Pods regelmäßig gelöscht, damit neue Pods den aktuellsten Patch haben.

Nachfolgend sehen Sie ein Beispielereignis, das Sie erhalten könnten, wenn die Pod-Bereinigung fehlschlägt. Es enthält Informationen über den Cluster, den Pod-Namen, den Pod-Namespace, das Fargate-Profil und die geplante Beendigungszeit.

```
{
    "version": "0",
    "id": "12345678-90ab-cdef-0123-4567890abcde",
    "detail-type": "EKS Fargate Pod Scheduled Termination",
    "source": "aws.eks",
    "account": "111122223333",
    "time": "2021-06-27T12:52:44Z",
    "region": "region-code",
    "resources": [
        "default/my-database-deployment"
    ],
    "detail": {
        "clusterName": "my-cluster",
```

```
        "fargateProfileName": "my-fargate-profile",
        "podName": "my-pod-name",
        "podNamespace": "default",
        "evictErrorMessage": "Cannot evict pod as it would violate the pod's disruption
 budget",
        "scheduledTerminationTime": "2021-06-30T12:52:44.832Z[UTC]"
    }
}
```

Einer der Gründe für ein Bereinigungsfehlerereignis kann sein, dass einem Pod mehrere PDBs zugeordnet sind. In diesem Fall gibt das Ereignis die folgende Fehlernachricht zurück.

```
"evictErrorMessage": "This pod has multiple PodDisruptionBudget, which the eviction
 subresource does not support",
```

Basierend auf diesem Ereignis können Sie eine gewünschte Aktion erstellen. Sie können beispielsweise Ihr Budget für Pod-Unterbrechung (PDB) anpassen, um zu steuern, wie die Pods bereinigt werden. Nehmen Sie zum Beispiel an, dass Sie ein PDB starten, dass den Zielprozentsatz von verfügbaren Pods angibt. Bevor Ihre Pods während eines Upgrades automatisch beendet werden, können Sie das PDB an einen unterschiedlichen Prozentsatz von Pods anpassen. Um dieses Ereignis zu erhalten, müssen Sie eine Amazon-EventBridge-Regel im AWS-Konto und der AWS-Region erstellen, zu denen der Cluster gehört. Die Regel muss das folgende benutzerdefinierte Muster verwenden. Weitere Informationen finden Sie unter Erstellen von Erstellen von Amazon-EventBridge-Regeln, die auf Ereignisse reagieren im Amazon-EventBridge-Benutzerhandbuch.

```
{
  "source": ["aws.eks"],
  "detail-type": ["EKS Fargate Pod Scheduled Termination"]
}
```

Es kann ein geeignetes Ziel zur Erfassung durch das Ereignis festgelegt werden. Eine vollständige Liste der verfügbaren Ziele finden Sie unter Amazon-EventBridge-Ziele im Amazon-EventBridge-Benutzerhandbuch. Sie können eine Benachrichtigungskonfiguration in AWS-Benutzerbenachrichtigungen erstellen. Wenn Sie die AWS-Managementkonsole verwenden, um die Benachrichtigung zu erstellen, wählen Sie unter Ereignisregeln Elastic Kubernetes Service (EKS) als AWS-Name und Geplante Beendigung des EKS-Fargate-Pod als Ereignistyp aus. Weitere Informationen finden Sie unter Erste Schritte mit AWS Benutzerbenachrichtigungen im AWS-Benutzerhandbuch für Benutzerbenachrichtigungen.

Häufig gestellte Fragen zu EKS-Pod-Bereinigungen finden Sie in den FAQs: Bereinigungsbeanchrichtigung für Fargate-Pods in AWS re:Post.

Erfassen Sie die App- und Nutzungsmetriken von AWS Fargate

Sie können Systemmetriken und CloudWatch Nutzungsmetriken für AWS Fargate sammeln.

Anwendungsmetriken

Für Anwendungen, die auf Amazon EKS und AWS Fargate ausgeführt werden, können Sie die AWS Distribution for OpenTelemetry (ADOT) verwenden. ADOT ermöglicht es Ihnen, Systemmetriken zu sammeln und sie an Container Insights-Dashboards zu CloudWatch senden. Informationen zu den ersten Schritten mit ADOT für Anwendungen, die auf Fargate ausgeführt werden, finden Sie unter Using CloudWatch Container Insights with AWS Distro for OpenTelemetry in der ADOT-Dokumentation.

Nutzungsmetriken

Sie können CloudWatch Nutzungsmetriken verwenden, um einen Überblick über die Ressourcennutzung Ihres Kontos zu erhalten. Verwenden Sie diese Metriken, um Ihre aktuelle Servicenutzung in CloudWatch Diagrammen und Dashboards zu visualisieren.

AWS Die Nutzungsmetriken von Fargate entsprechen den AWS Servicekontingenten. Sie können Alarme konfigurieren, mit denen Sie benachrichtigt werden, wenn sich Ihre Nutzung einem Servicekontingent nähert. Weitere Hinweise zu Servicekontingenten für Fargate finden Sie unter the section called “Servicekontingente”.

AWS Fargate veröffentlicht die folgenden Metriken im `AWS/Usage` Namespace.

Metrik	Beschreibung
`ResourceCount`	Die Gesamtanzahl der angegebenen Ressourcen, die in Ihrem Konto ausgeführt werden. Die Ressource wird durch die Dimensionen definiert, die der Metrik zugeordnet sind.

Die folgenden Dimensionen werden verwendet, um die von AWS Fargate veröffentlichten Nutzungsmetriken zu verfeinern.

Dimension	Description
`Service`	Der Name des AWS Dienstes, der die Ressource enthält. Für AWS Fargate-Nutzungsmetriken ist `Fargate` der Wert für diese Dimension.
`Type`	Der Typ der Entität, die gemeldet wird. Derzeit ist der einzig gültige Wert für AWS Fargate-Nutzungsmetriken. `Resource`
`Resource`	Der Typ der Ressource, die ausgeführt wird. Derzeit gibt AWS Fargate Informationen zu Ihrer Fargate On-Demand-Nutzung zurück. Der Ressourcenwert für Fargate On-Demand-Nutzung ist `OnDemand`. [HINWEIS] ==== Die Fargate On-Demand-Nutzung kombiniert Amazon-EKS-Pods mit Fargate, Amazon-ECS-Aufgaben mit dem Fargate-Starttyp und Amazon-ECS-Aufgaben mithilfe des `FARGATE`-Kapazitätsanbieters. ====
`Class`	Die Klasse der nachverfolgten Ressource. Derzeit verwendet AWS Fargate die Klassendimension nicht.

Einen CloudWatch Alarm zur Überwachung der Fargate-Ressourcennutzungsmetriken erstellen

AWS Fargate stellt CloudWatch Nutzungsmetriken bereit, die den AWS Servicekontingenten für die Nutzung von Fargate On-Demand-Ressourcen entsprechen. In der Service-Quotas-Konsole können Sie Ihre Nutzung in einem Diagramm visualisieren. Sie können auch Alarme konfigurieren, die Sie

warnen, wenn sich Ihre Nutzung einem Service Quotas nähert. Weitere Informationen finden Sie unter the section called “Metriken erfassen”.

Gehen Sie wie folgt vor, um einen CloudWatch Alarm auf der Grundlage der Fargate-Kennzahlen zur Ressourcennutzung zu erstellen.

1. Öffnen Sie die Service Quotas Quotas-Konsole unter https://console.aws.amazon.com/servicequotas/.
2. Wählen Sie im linken Navigationsbereich AWS Dienste aus.
3. Suchen Sie in der AWS Serviceliste nach AWS Fargate und wählen Sie es aus.
4. Wählen Sie in der Liste Service quotas (Servicekontingente) das Fargate-Nutzungskontingent aus, für das Sie einen Alarm erstellen möchten.
5. Wählen Sie im Bereich CloudWatch Amazon-Alarme die Option Erstellen aus.
6. Wählen Sie bei Alarmschwellenwert den Prozentsatz des angewendeten Kontingentwerts aus, den Sie als Alarmwert festlegen möchten.
7. Geben Sie bei Alarmname einen Namen für den Alarm ein und wählen Sie dann Erstellen aus.

AWS-Fargate-Protokollierung für Ihren Cluster starten

Amazon EKS auf Fargate bietet einen integrierten Log-Router basierend auf Fluent Bit. Dies bedeutet, dass Sie einen Fluent-Bit-Container nicht explizit als Sidecar ausführen, sondern Amazon für Sie ausführen. Alles, was Sie tun müssen, ist den Log-Router zu konfigurieren. Die Konfiguration erfolgt über ein dediziertes `ConfigMap`, das die folgenden Kriterien erfüllen muss:

- Benannte `aws-logging`
- Erstellt in einem dedizierten Namespace namens `aws-observability`
- Mehr als 5 300 Zeichen sind nicht zulässig.

Sobald Sie die `ConfigMap` erstellt haben, erkennt Amazon EKS in Fargate sie automatisch und konfiguriert den Protokollrouter damit. Fargate verwendet eine Version von AWS für Fluent Bit, eine Upstream-kompatible Distribution von Fluent Bit, die von AWS verwaltet wird. Weitere Informationen finden Sie unter AWS für Fluent Bit auf GitHub.

Mit dem Log-Router können Sie die Bandbreite der Services von AWS für Protokoll-Analysen und -Speicherung nutzen. Sie können Protokolle von Fargate direkt an Amazon CloudWatch, Amazon

OpenSearch Service streamen. Außerdem können Sie Protokolle über Amazon Kinesis Data Firehose an Ziele wie Amazon S3, Amazon Kinesis Data Streams und Partner-Tools streamen.

- Ein vorhandenes Fargate-Profil, das einen vorhandenen Kubernetes-Namespace angibt, in dem Sie Fargate-Pods bereitstellen. Weitere Informationen finden Sie unter the section called “Schritt 3: Fargate Profil für Ihren Cluster erstellen”.
- Eine vorhandene Fargate-Pod-Ausführungsrolle. Weitere Informationen finden Sie unter the section called “Schritt 2: Fargate-Pod-Ausführungsrolle erstellen”.

Router-Konfiguration protokollieren

> **Important**
>
> Damit Protokolle erfolgreich veröffentlicht werden können, muss von der VPC, in der sich Ihr Cluster befindet, Netzwerkzugriff auf das Protokollziel bestehen. Dies betrifft vor allem Benutzer, die Ausgangsregeln für ihre VPC anpassen. Ein Beispiel für die Verwendung von CloudWatch finden Sie unter Verwendung von CloudWatch-Protokollen mit Schnittstellen-VPC-Endpunkten im Benutzerhandbuch zu Amazon CloudWatch Logs.

Ersetzen Sie in den folgenden Schritten jedes *Beispielwert* durch Ihre eigenen Werte.

1. Erstellen Sie einen dedizierten Kubernetes-Namespace namens `aws-observability`.
   a. Speichern Sie den folgenden Inhalt in einer Datei mit dem Namen `aws-observability-namespace.yaml` auf Ihrem Computer. Der Wert für `name` muss `aws-observability` sein und das Label `aws-observability: enabled` ist erforderlich.

```
kind: Namespace
apiVersion: v1
metadata:
  name: aws-observability
  labels:
    aws-observability: enabled
```

   b. Erstellen Sie den Namespace.

```
kubectl apply -f aws-observability-namespace.yaml
```

2. Erstellen Sie ein `ConfigMap` mit einem `Fluent Conf`-Datenwert, um Container-Logs an ein Ziel zu versenden. Fluent Conf ist Fluent Bit, eine schnelle und leichte Konfigurationssprache für den Protokollprozessor, die verwendet wird, um Container-Protokolle an ein Protokollziel Ihrer Wahl weiterzuleiten. Weitere Informationen finden Sie unter Konfigurationsdatei in der Fluent-Bit-Dokumentation.

> **Important**
>
> In einer typischen `Fluent Conf` sind die Hauptabschnitte `Service`, `Input`, `Filter` und `Output` enthalten. Der Fargate-Protokoll-Router akzeptiert jedoch nur:
>
> - Die Abschnitte `Filter` und `Output`.
> - Ein `Parser`-Abschnitt.
>
> Wenn Sie andere Abschnitte angeben, werden diese abgelehnt.

Der Fargate-Protokollrouter verwaltet die Abschnitte `Service` und `Input`. Es verfügt über den folgenden Abschnitt `Input`, der nicht geändert werden kann und in Ihrem `ConfigMap` nicht benötigt wird. Sie können daraus jedoch Erkenntnisse gewinnen, z. B. die Speicherpuffergrenze und das für Protokolle angewendete Tag.

```
[INPUT]
    Name tail
    Buffer_Max_Size 66KB
    DB /var/log/flb_kube.db
    Mem_Buf_Limit 45MB
    Path /var/log/containers/*.log
    Read_From_Head On
    Refresh_Interval 10
    Rotate_Wait 30
    Skip_Long_Lines On
    Tag kube.*
```

Berücksichtigen Sie beim Erstellen des `ConfigMap`, die folgenden Regeln, die Fargate verwendet, um Felder zu validieren:

- `[FILTER]`, `[OUTPUT]`, und `[PARSER]` sollen unter jedem entsprechenden Schlüssel angegeben werden. `[FILTER]` muss beispielsweise unter `filters.conf` liegen. Sie können ein oder mehrere `[FILTER]` in der `filters.conf` haben. Die Abschnitte `[OUTPUT]` und `[PARSER]` sollten sich auch unter den entsprechenden Schlüssel befinden. Durch die Angabe

mehrerer `[OUTPUT]`-Abschnitte können Sie Ihre Protokolle gleichzeitig an verschiedene Ziele weiterleiten.

- Fargate validiert die erforderlichen Schlüssel für jeden Abschnitt. `Name` und `matchs`ind für jedes `[FILTER]` und `[OUTPUT]` erforderlich. `Name` und `format` werden jeweils für jeden `[PARSER]` benötigt. Bei den Schlüssel wird nicht zwischen Groß- und Kleinschreibung unterschieden.
- Umgebungsvariablen wie `${ENV_VAR}` sind im `ConfigMap` nicht erlaubt.
- Die Einrückung muss für die Direktive oder das Schlüssel-Wert-Paar innerhalb von `filters.conf`, `output.conf` und `parsers.conf` gleich sein. Schlüssel-Wert-Paare müssen stärker eingerückt werden als Direktiven.
- Fargate validiert gegen die folgenden unterstützten Filter: `grep`, `parser`, `record_modifier`, `rewrite_tag`, `throttle`, `nest`, `modify`, und `kubernetes`.
- Fargate validiert mit der folgenden unterstützten Ausgabe: `es`, `firehose`, `kinesis_firehose`, `cloudwatch`, `cloudwatch_logs`, und `kinesis`.
- Mindestens ein unterstütztes `Output`-Plugin muss im `ConfigMap` bereitgestellt werden, um die Protokollierung zu ermöglichen. `Filter` und `Parser` sind nicht erforderlich, um die Protokollierung zu aktivieren.

Sie können Fluent Bit auch auf Amazon EC2 mit der gewünschten Konfiguration ausführen, um alle Probleme zu beheben, die sich aus der Validierung ergeben. Erstellen Sie Ihr `ConfigMap` mit einem der folgenden Beispiele.

> **Important**
>
> Die Amazon-EKS-Fargate-Protokollierung unterstützt keine dynamische Konfiguration einer `ConfigMap`. Alle Änderungen an eine `ConfigMap` werden nur auf neue Pods angewendet. Änderungen werden nicht auf vorhandene Pods angewendet.

Erstellen Sie anhand des Beispiels ein `ConfigMap` für Ihr gewünschtes Protokollziel.

> **Note**
>
> Sie können auch Amazon Kinesis Data Streams als Protokollziel verwenden. Stellen Sie bei der Nutzung von Kinesis Data Streams sicher, dass der Pod-Ausführungsrolle die Berechtigung `kinesis:PutRecords` erteilt wurde. Weitere Informationen finden

Sie unter Berechtigungen für Amazon Kinesis Data Streams im offiziellen Handbuch zu Fluent Bit.

Example

CloudWatch

Bei der Verwendung von CloudWatch haben Sie zwei Ausgabeoptionen:

- Ein Ausgabe-Plug-In in C geschrieben
- Ein in Golang geschriebenes Ausgabe-Plug-In

Das folgende Beispiel zeigt Ihnen, wie Sie das `cloudwatch_logs`-Plug-in verwenden, um Protokolle an CloudWatch zu senden.

a. Speichern Sie den folgenden Inhalt in einer Datei namens `aws-logging-cloudwatch-configmap.yaml`. Ersetzen Sie den *`region-code`* durch die AWS-Region, in der sich Ihr Cluster befindet. Die Parameter unter `[OUTPUT]` sind erforderlich.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
data:
  flb_log_cw: "false"  # Set to true to ship Fluent Bit process logs to
 CloudWatch.
  filters.conf: |
    [FILTER]
        Name parser
        Match *
        Key_name log
        Parser crio
    [FILTER]
        Name kubernetes
        Match kube.*
        Merge_Log On
        Keep_Log Off
        Buffer_Size 0
        Kube_Meta_Cache_TTL 300s
  output.conf: |
    [OUTPUT]
        Name cloudwatch_logs
```

```
        Match   kube.*
        region region-code
        log_group_name my-logs
        log_stream_prefix from-fluent-bit-
        log_retention_days 60
        auto_create_group true
  parsers.conf: |
    [PARSER]
        Name crio
        Format Regex
        Regex ^(?<time>[^ ]+) (?<stream>stdout|stderr) (?<logtag>P|F) (?
<log>.*)$
        Time_Key    time
        Time_Format %Y-%m-%dT%H:%M:%S.%L%z
```

b. Wenden Sie das Manifest auf Ihren Cluster an.

```
kubectl apply -f aws-logging-cloudwatch-configmap.yaml
```

Amazon OpenSearch Service

Wenn Sie Protokolle an Amazon OpenSearch Service senden möchten, können Sie die es-Ausgabe verwenden, ein in C geschriebenes Plug-In. Das folgende Beispiel zeigt, wie Sie mit dem Plug-In Protokolle an OpenSearch senden können.

a. Speichern Sie den folgenden Inhalt in einer Datei mit dem Namen `aws-logging-opensearch-configmap.yaml`. Ersetzen Sie jeden *`Beispielwert`* durch Ihre eigenen Werte.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
data:
  output.conf: |
    [OUTPUT]
      Name  es
      Match *
      Host  search-example-gjxdcilagiprbglqn42jsty66y.region-
code.es.amazonaws.com
      Port  443
      Index example
```

```
      Type  example_type
      AWS_Auth On
      AWS_Region region-code
      tls   On
```

b. Wenden Sie das Manifest auf Ihren Cluster an.

```
kubectl apply -f aws-logging-opensearch-configmap.yaml
```

Firehose

Beim Senden von Protokollen an Firehose haben Sie zwei Ausgabeoptionen:

- kinesis_firehose – Ein in C geschriebenes Ausgabe-Plugin.
- firehose – Ein in Golang geschriebenes Ausgabe-Plugin.

Das folgende Beispiel zeigt Ihnen, wie Sie das `kinesis_firehose`-Plugin verwenden, um Protokolle an Firehose zu senden.

a. Speichern Sie den folgenden Inhalt in einer Datei mit dem Namen `aws-logging-firehose-configmap.yaml`. Ersetzen Sie den *region-code* durch die AWS-Region, in der sich Ihr Cluster befindet.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
data:
  output.conf: |
    [OUTPUT]
     Name  kinesis_firehose
     Match *
     region region-code
     delivery_stream my-stream-firehose
```

b. Wenden Sie das Manifest auf Ihren Cluster an.

```
kubectl apply -f aws-logging-firehose-configmap.yaml
```

3. Richten Sie Berechtigungen für die Fargate-Pod-Ausführungsrolle ein, um Protokolle an Ihr Ziel zu senden.

a. Laden Sie die IAM-Richtlinie für Ihr Ziel auf Ihren Computer herunter.

Example

CloudWatch

Laden Sie die CloudWatch IAM-Richtlinie auf Ihren Computer herunter. Sie können die Richtlinie auch auf GitHub anzeigen.

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-fluent-
logging-examples/mainline/examples/fargate/cloudwatchlogs/permissions.json
```

Amazon OpenSearch Service

Laden Sie die OpenSearch-IAM-Richtlinie auf Ihren Computer herunter. Sie können die Richtlinie auch auf GitHub anzeigen.

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-
fluent-logging-examples/mainline/examples/fargate/amazon-elasticsearch/
permissions.json
```

Stellen Sie sicher, dass die Zugriffssteuerung von OpenSearch Dashboards richtig konfiguriert ist. Dem `all_access role` in OpenSearch-Dashboards muss die Fargate-Pod-Ausführungsrolle und die IAM-Rolle zugeordnet sein. Das gleiche Mapping muss für die `security_manager`-Rolle durchgeführt werden. Sie können die vorherigen Zuordnungen hinzufügen, indem Sie `Menu`, dann `Security`, dann `Roles` und dann die entsprechenden Rollen auswählen. Weitere Informationen finden Sie unter Wie behebe ich Probleme mit CloudWatch Logs, damit es an meine Amazon-ES-Domain gestreamt wird?.

Firehose

Laden Sie die Firehose-IAM-Richtlinie auf Ihren Computer herunter. Sie können die Richtlinie auch auf GitHub anzeigen.

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-fluent-
logging-examples/mainline/examples/fargate/kinesis-firehose/permissions.json
```

b. Erstellen Sie eine IAM-Richtlinie aus der heruntergeladenen Richtliniendatei.

```
aws iam create-policy --policy-name eks-fargate-logging-policy --policy-document
 file://permissions.json
```

c. Hängen Sie die IAM-Richtlinie an die Pod-Ausführungsrolle an, die für Ihr Fargate-Profil angegeben ist, mit dem folgenden Befehl. Ersetzen Sie *111122223333* durch Ihre Konto-ID. Ersetzen Sie *AmazonEKSFargatePodExecutionRole* durch Ihre Pod-Ausführungsrolle (weitere Informationen finden Sie unter the section called “Schritt 2: Fargate-Pod-Ausführungsrolle erstellen”).

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::111122223333:policy/eks-fargate-logging-policy \
  --role-name AmazonEKSFargatePodExecutionRole
```

Support von Kubernetes-Filtern

Mit dem Fluent-Bit-Kubernetes-Filter können Sie Kubernetes-Metadaten zu Ihren Protokolldateien hinzufügen. Weitere Informationen über den Filter finden Sie unter Kubernetes in der Fluent-Bit-Dokumentation. Sie können einen Filter unter Verwendung des Endpunkts des API-Servers anwenden.

```
filters.conf: |
    [FILTER]
        Name             kubernetes
        Match            kube.*
        Merge_Log           On
        Buffer_Size         0
        Kube_Meta_Cache_TTL 300s
```

Important

- `Kube_URL`,`Kube_CA_File`,`Kube_Token_Command`, und `Kube_Token_File` sind Konfigurationsparameter im Servicebesitz und dürfen nicht angegeben werden. Amazon-EKS-Fargate füllt diese Werte aus.
- `Kube_Meta_Cache_TTL` ist die Zeit, in der Fluent Bit wartet, bis es mit dem API-Server für die neuesten Metadaten kommuniziert. Wenn `Kube_Meta_Cache_TTL` nicht angegeben wird, dann hängt Amazon EKS Fargate einen Standardwert von 30 Minuten an, um die Belastung des API-Servers zu verringern.

So übermitteln Sie Fluent-Bit-Prozessprotokolle an Ihr Konto

Sie können Fluent-Bit-Prozessprotokolle optional mit den folgenden `ConfigMap` an Amazon CloudWatch senden. Für den Versand von Fluent Bit-Prozessprotokollen an CloudWatch fallen zusätzliche Kosten für die Protokollaufnahme und Speicherung an. Ersetzen Sie den *`region-code`* durch die AWS-Region, in der sich Ihr Cluster befindet.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
  labels:
data:
  # Configuration files: server, input, filters and output
  # ======================================================
  flb_log_cw: "true"  # Ships Fluent Bit process logs to CloudWatch.

  output.conf: |
    [OUTPUT]
        Name cloudwatch
        Match kube.*
        region region-code
        log_group_name fluent-bit-cloudwatch
        log_stream_prefix from-fluent-bit-
        auto_create_group true
```

Die Protokolle befinden sich in CloudWatch in derselben AWS-Region wie der Cluster. Der Name der Protokollgruppe lautet *`my-cluster`*`-fluent-bit-logs` und der Name des Fluent-Bit-Logstreams lautet `fluent-bit-`*`podname`*`-`*`pod-namespace`* .

> **Note**
>
> - Die Prozessprotokolle werden nur ausgeliefert, wenn der Fluent-Bit-Prozess erfolgreich gestartet wird. Wenn beim Starten von Fluent Bit ein Fehler auftritt, werden die Prozessprotokolle verpasst. Sie können nur Prozessprotokolle an CloudWatch senden.
> - Um Versandprozessprotokolle auf Ihr Konto zu debuggen, können Sie die vorherigen `ConfigMap` anwenden, um die Prozessprotokolle zu erhalten. Dass Fluent Bit nicht

gestartet werden kann, liegt normalerweise daran, dass Ihr `ConfigMap` beim Starten nicht von Fluent Bit analysiert oder akzeptiert wird.

So beenden Sie die Übermittlung von Fluent-Bit-Prozessprotokollen

Für den Versand von Fluent Bit-Prozessprotokollen an CloudWatch fallen zusätzliche Kosten für die Protokollaufnahme und Speicherung an. Gehen Sie wie folgt vor, um Prozessprotokolle in einem vorhandenen `ConfigMap`-Setup auszuschließen.

1. Suchen Sie die CloudWatch-Protokollgruppe, die automatisch für die Fluent-Bit-Prozessprotokolle Ihres Amazon-EKS-Clusters erstellt wurde, nachdem Sie die Fargate-Protokollierung aktiviert haben. Es folgt dem Format *`my-cluster`*`-fluent-bit-logs`.
2. Löschen Sie die vorhandenen CloudWatch-Protokollstreams, die für die Prozessprotokolle der einzelnen Pods in der CloudWatch-Protokollgruppe erstellt wurden.
3. Bearbeiten Sie die `ConfigMap` und stellen Sie `flb_log_cw: "false"` ein.
4. Starten Sie alle vorhandenen Pods im Cluster neu.

Testanwendung

1. Stellen Sie einen Beispiel-Pod bereit.
   a. Speichern Sie den folgenden Inhalt in einer Datei mit dem Namen `sample-app.yaml` auf Ihrem Computer.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: sample-app
  namespace: same-namespace-as-your-fargate-profile
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
```

```
      containers:
        - name: nginx
          image: nginx:latest
          ports:
            - name: http
              containerPort: 80
```

b. Wenden Sie das Manifest auf den Cluster an.

```
kubectl apply -f sample-app.yaml
```

2. Zeigen Sie die NGINX-Protokolle mit den Zielen an, die Sie im `ConfigMap`.

Größenüberlegungen

Wir empfehlen Ihnen, für den Protokoll-Router bis zu 50 MB Speicher einzuplanen. Wenn Sie erwarten, dass Ihre Anwendung Protokolle mit sehr hohem Durchsatz generiert, sollten Sie bis zu 100 MB einplanen.

Fehlersuche

Um zu überprüfen, ob das Protokollierungs-Feature aus irgendeinem Grund aktiviert oder deaktiviert ist, z. B. ein ungültiges `ConfigMap`, und warum es ungültig ist, überprüfen Sie Ihre Pod-Ereignisse mit `kubectl describe pod` *`pod-name`* . Die Ausgabe kann Pod-Ereignisse enthalten, die klarstellen, ob die Protokollierung aktiviert ist oder nicht, wie die folgende Beispielausgabe.

```
[...]
Annotations:          CapacityProvisioned: 0.25vCPU 0.5GB
                      Logging: LoggingDisabled: LOGGING_CONFIGMAP_NOT_FOUND
[...]
Events:
  Type     Reason           Age        From
               Message
  ----     ------           ----       ----
               -------
  Warning  LoggingDisabled  <unknown>  fargate-scheduler
                Disabled logging because aws-logging configmap was not found. configmap
 "aws-logging" not found
```

Die Pod-Ereignisse sind kurzlebig mit einem Zeitraum, der von den Einstellungen abhängt. Sie können die Annotationen eines Pods auch mit `kubectl describe pod` *`pod-name`* anzeigen. In

der Pod-Annotation finden Sie Informationen darüber, ob die Protokollierungs-Feature aktiviert oder deaktiviert ist, und den Grund.

Auswahl eines optimalen Amazon-EC2-Knoten-Instance-Typs

Amazon EC2 bietet eine große Auswahl an Instance-Typen für Worker-Knoten. Jeder Instance-Typ bietet andere Merkmale in Bezug auf Datenverarbeitung, Arbeitsspeicher, Speicher und Netzwerkfunktionen. Jede Instance wird abhängig von diesen Eigenschaften auch in Instance-Familien eingeordnet. Eine Liste finden Sie unter Verfügbare Instance-Typen im Amazon-EC2-Benutzerhandbuch. Amazon EKS veröffentlicht mehrere Varianten von Amazon-EC2-AMIs, um Support zu ermöglichen. Berücksichtigen Sie die folgenden Kriterien, um sicherzustellen, dass der ausgewählte Instance-Typ mit Amazon EKS kompatibel ist.

- Nicht alle Amazon-EKS-AMIs unterstützen derzeit die `mac`-Familie.
- Arm und nicht beschleunigte Amazon-EKS-AMIs unterstützen die Familien `g3`, `g4`, `inf` und `p` nicht.
- Beschleunigte Amazon-EKS-AMIs unterstützen die Familien `a`, `c`, `hpc`, `m` und `t`.
- Für Arm-basierte Instances unterstützt Amazon Linux 2023 (AL2023) ausschließlich Instance-Typen, die Graviton2-Prozessoren oder neuere Prozessoren verwenden. AL2023 unterstützt keine `A1`-Instances.

Berücksichtigen Sie bei der Auswahl zwischen Instance-Typen, die von Amazon EKS unterstützt werden, die folgenden Funktionen jedes Typs.

Anzahl der Instances in einer Knotengruppe

> Im Allgemeinen sind weniger, dafür größere Instances vorzuziehen, insbesondere wenn Sie über viele Daemonsets verfügen. Jede Instance erfordert API-Aufrufe an den API-Server. Je mehr Instances Sie haben, desto mehr Last auf dem API-Server.

Betriebssystem

> Überprüfen Sie die unterstützten Instance-Typen für Linux, Windows und Bottlerocket. Lesen Sie vor dem Erstellen von Windows-Instances den Abschnitt Bereitstellen von Windows-Knoten in EKS-Clustern.

Hardware-Architektur

Benötigen Sie x86 oder Arm? Bevor Sie Arm-Instances bereitstellen, überprüfen Sie die für Amazon-EKS-optimierten Arm-Amazon-Linux-AMIs. Benötigen Sie Instances, die auf dem Nitro-System (Linux oder Windows) basieren oder über Beschleunigte Funktionen verfügen? Wenn Sie beschleunigte Funktionen benötigen, können Sie Linux nur mit Amazon EKS verwenden.

Maximale Anzahl an Pods

Da jedem Pod eine eigene IP-Adresse zugewiesen wird, ist die Anzahl der IP-Adressen für einen Instance-Typ ein Faktor bei der Bestimmung der Anzahl der Pods, die auf der Instance ausgeführt werden können. Um manuell zu bestimmen, wie viele Pods ein Instance-Typ unterstützt, lesen Sie the section called “Von Amazon EKS empfohlene maximale Pods für jeden Amazon-EC2-Instance-Typ”.

> **Note**
>
> Wenn Sie ein Amazon-EKS-optimiertes Amazon-Linux-2-AMI der Version `v20220406` oder neuer verwenden, können Sie einen neuen Instance-Typ verwenden, ohne auf das neueste AMI zu aktualisieren. Für diese AMIs berechnet das AMI automatisch den erforderlichen `max-pods`-Wert, wenn dieser nicht in der Datei eni-max-pods.txt aufgeführt ist. Instance-Typen, die sich derzeit in der Vorschau befinden, werden möglicherweise von Amazon EKS standardmäßig nicht unterstützt. Werte für `max-pods` für solche Typen müssen noch zu `eni-max-pods.txt` in unserem AMI hinzugefügt werden.

AWS-Nitro-System-Instance-Typen unterstützen optional deutlich mehr IP-Adressen als Nicht-Nitro-System-Instance-Typen. Allerdings stehen Pods nicht alle IP-Adressen zur Verfügung, die einer Instance zugewiesen wurden. Um Ihren Instances eine deutlich größere Anzahl von IP-Adressen zuzuweisen, müssen Sie Version `1.9.0` oder höher des Amazon VPC CNI-Add-ons in Ihrem Cluster installiert und entsprechend konfiguriert haben. Weitere Informationen finden Sie unter the section called “IP-Adressen erhöhen”. Um Ihren Instances die größte Anzahl von IP-Adressen zuzuweisen, müssen Sie Version `1.10.1` oder höher des Amazon VPC CNI-Add-ons in Ihrem Cluster installiert haben und den Cluster mit der `IPv6`-Familie bereitstellen.

IP-Familie

Sie können jeden unterstützten Instance-Typ verwenden, wenn Sie die `IPv4`-Familie für einen Cluster nutzen, was es Ihrem Cluster ermöglicht, Ihren Pods und Services private `IPv4`-Adressen

zuzuweisen. Wenn Sie jedoch die `IPv6`-Familie für Ihren Cluster verwenden möchten, müsste Sie die AWS-Nitro-System-Instance-Typen oder Bare-Metal-Instance-Typen verwenden. Nur `IPv4` wird für Windows-Instances unterstützt. Ihr Cluster muss auf dem Version `1.10.1` oder höher des Amazon-VPC-CNI-Add-ons ausgeführt wird. Weitere Informationen zur Verwendung von `IPv6` finden Sie unter the section called “IPv6”.

Version des Amazon-VPC-CNI-Add-Ons, das Sie ausführen

Die aktuelle Version des Amazon-CNI-Plugins für Kubernetes unterstützt diese Instance-Typen. Möglicherweise müssen Sie Ihre Amazon VPC-CNI-Add-on-Version aktualisieren, um die Vorteile der neuesten unterstützten Instance-Typen zu nutzen. Weitere Informationen finden Sie unter the section called “Amazon VPC CNI”. Die neueste Version unterstützt die neuesten Features für die Verwendung mit Amazon EKS. Frühere Versionen unterstützen nicht alle Feature. Sie können die von verschiedenen Versionen unterstützten Funktionen im Änderungsverlauf auf GitHub anzeigen.

AWS-Region, in der Sie Ihre Knoten erstellen

Nicht alle Instance-Typen stehen in allen AWS-Regionen zur Verfügung.

Ob Sie Sicherheitsgruppen für Pods verwenden

Wenn Sie Sicherheitsgruppen für Pods verwenden, werden nur bestimmte Instance-Typen unterstützt. Weitere Informationen finden Sie unter the section called “Sicherheitsgruppen für Pods”.

Von Amazon EKS empfohlene maximale Pods für jeden Amazon-EC2-Instance-Typ

Da jedem Pod eine eigene IP-Adresse zugewiesen wird, ist die Anzahl der IP-Adressen für einen Instance-Typ ein Faktor bei der Bestimmung der Anzahl der Pods, die auf der Instance ausgeführt werden können. Amazon EKS stellt ein Skript bereit, das Sie herunterladen und ausführen können, um die von Amazon EKS empfohlene maximale Anzahl von Pods zur Ausführung auf jedem Instance-Typ zu bestimmen. Das Skript verwendet Hardwareattribute jeder Instance und Konfigurationsoptionen, um die maximale Pod-Anzahl zu bestimmen. Sie können die in diesen Schritten zurückgegebene Zahl verwenden, um Funktionen wie das Zuweisen von IP-Adressen zu Pods aus einem anderen Subnetz als dem der Instance zu aktivieren und die Anzahl der IP-Adressen für Ihre Instance erheblich zu erhöhen. Wenn Sie eine verwaltete Knotengruppe mit mehreren Instance-Typen verwenden, verwenden Sie einen Wert, der für alle Instance-Typen funktioniert.

1. Laden Sie ein Skript herunter, mit dem Sie die maximale Anzahl von Pods für jeden Instance-Typ berechnen können.

```
curl -O https://raw.githubusercontent.com/awslabs/amazon-eks-ami/master/templates/
al2/runtime/max-pods-calculator.sh
```

2. Markieren Sie das Skript auf Ihrem Computer als ausführbar.

```
chmod +x max-pods-calculator.sh
```

3. Führen Sie das Skript aus und ersetzen Sie *m5.large* durch den Instance-Typ, den Sie bereitstellen möchten, und *1.9.0-eksbuild.1* durch Ihre Amazon-VPC-CNI-Add-On-Version. Informationen zum Ermitteln Ihrer Add-On-Version finden Sie in den Aktualisierungsverfahren unter Zuweisen von IPs zu Pods mit dem Amazon VPC CNI.

```
./max-pods-calculator.sh --instance-type m5.large --cni-version 1.9.0-eksbuild.1
```

   Eine Beispielausgabe sieht wie folgt aus.

```
29
```

   Sie können dem Skript die folgenden Optionen hinzufügen, um die maximal unterstützten Pods anzuzeigen, wenn optionale Funktionen verwendet werden.

   - `--cni-custom-networking-enabled` – Verwenden Sie diese Option, wenn Sie IP-Adressen aus einem anderen Subnetz als der Ihrer Instance zuweisen möchten. Weitere Informationen finden Sie unter the section called “Benutzerdefinierte Netzwerke”. Das Hinzufügen dieser Option zum vorherigen Skript mit denselben Beispielwerten ergibt `20`.
   - `--cni-prefix-delegation-enabled` – Verwenden Sie diese Option, wenn Sie jeder Elastic-Network-Schnittstelle deutlich mehr IP-Adressen zuweisen möchten. Diese Funktion erfordert eine Amazon Linux-Instance, die auf dem Nitro System und der Version `1.9.0` oder höher des Amazon VPC CNI-Add-ons ausgeführt wird. Weitere Informationen finden Sie unter the section called “IP-Adressen erhöhen”. Das Hinzufügen dieser Option zum vorherigen Skript mit denselben Beispielwerten ergibt `110`.

Sie können das Skript auch mit der Option `--help` ausführen, um alle verfügbaren Optionen anzuzeigen.

> **Note**
>
> Das Berechnungsskript für maximale Pods begrenzt den Rückgabewert basierend auf den Skalierbarkeitsschwellenwerten und empfohlenen Einstellungen von Kubernetes auf `110`. Wenn Ihr Instance-Typ mehr als 30 vCPUs hat, steigt dieser Grenzwert auf `250`, eine Zahl, die auf internen Tests des Amazon-EKS-Skalierbarkeitsteams basiert. Weitere Informationen finden Sie im Blogbeitrag Amazon VPC CNI plugin increases pods per node limits (VPC-CNI-Plugin von Amazon erhöht Grenzwerte für Pods pro Knoten).

Überlegungen zu EKS Auto Mode

EKS Auto Mode begrenzt die Anzahl der Pods auf Knoten auf den niedrigeren der folgenden Werte:

- 110 Pods, feste Obergrenze
- Das Ergebnis der oben beschriebenen Berechnung der maximalen Anzahl von Pods.

Erstellung von Knoten mit vorgefertigten optimierten Images

Sie können Knoten mit vorgefertigten, für Amazon EKS optimierten Amazon Machine Images (AMIs) oder Ihren eigenen benutzerdefinierten AMIs bereitstellen, wenn Sie verwaltete Knotengruppen oder selbstverwaltete Knoten verwenden. Wenn Sie Hybridknoten ausführen, lesen Sie the section called “Betriebssystem vorbereiten”. Informationen über die einzelnen Typen von Amazon-EKS-optimierten AMIs finden Sie in den folgenden Themen. Anweisungen zum Erstellen eines eigenen benutzerdefinierten AMI finden Sie unter the section called “Erstellen Sie ein benutzerdefiniertes EKS-optimiertes Amazon Linux-AMI”.

Mit Amazon EKS Auto Mode verwaltet EKS die EC2-Instance, einschließlich der Auswahl und Aktualisierung der AMI.

Themen

- Leitfaden zu den Funktionen von EKS AL2 und AL2 -Accelerated AMIs Transition
- Knoten mit optimiertem Amazon Linux erstellen AMIs
- Erstellen Sie Knoten mit optimiertem Bottlerocket AMIs
- Erstellung von Knoten mit optimierten Ubuntu-Linux-AMIs
- Erstellung von Knoten mit optimierten Windows-AMIs

Leitfaden zu den Funktionen von EKS AL2 und AL2 -Accelerated AMIs Transition

> **Warning**
>
> Amazon EKS hat die Veröffentlichung von EKS-optimiertem Amazon Linux 2 (AL2) AMIs am 26. November 2025 eingestellt. AL2023 und Bottlerocket auf Basis von Amazon EKS sind AMIs für alle unterstützten Kubernetes-Versionen einschließlich 1.33 und höher verfügbar.

AWS wird die Unterstützung für EKS — AL2 optimiert und AL2 beschleunigt — mit Wirkung zum 26. November 2025 einstellen. AMIs Sie können EKS zwar auch AL2 AMIs nach dem end-of-support (EOS-) Datum (26. November 2025) weiter verwenden, nach diesem Datum wird EKS jedoch keine neuen Kubernetes-Versionen oder Updates mehr veröffentlichen AL2 AMIs, einschließlich kleinerer Releases, Patches und Bugfixes. Wir empfehlen ein Upgrade auf Amazon Linux 2023 (AL2023) oder AMIs Bottlerocket:

- AL2023 ermöglicht einen secure-by-default Ansatz mit vorkonfigurierten Sicherheitsrichtlinien SELinux im permissiven Modus, standardmäßig aktiviertem IMDSv2 Nur-Modus, optimierten Startzeiten und verbesserter Paketverwaltung für mehr Sicherheit und Leistung. Er eignet sich gut für Infrastrukturen, die umfangreiche Anpassungen wie direkten Zugriff auf Betriebssystemebene oder umfangreiche Knotenänderungen erfordern. Weitere Informationen finden Sie unter AL2 FAQs023 oder in unseren ausführlichen Migrationsleitlinien unter. the section called “Upgrade auf AL2 023”
- Bottlerocket bietet mit seinem speziell entwickelten, containeroptimierten Design verbesserte Sicherheit, schnellere Bootzeiten und eine kleinere Angriffsfläche für mehr Effizienz. Es eignet sich gut für containernative Ansätze mit minimalen Knotenanpassungen. Weitere Informationen finden Sie auf Bottlerocket FAQs oder in unseren ausführlichen Migrationsleitfäden unter. the section called “Bottlerocket”

Alternativ können Sie dies the section called “Erstellen Sie ein benutzerdefiniertes EKS-optimiertes Amazon Linux-AMI” bis zum EOS-Datum (26. November 2025) tun. Darüber hinaus können Sie bis zum Amazon Linux 2 EOS-Datum (30. Juni 2026) ein benutzerdefiniertes AMI mit einer Amazon Linux 2-Basis-Instance erstellen.

Migration und Support FAQs

Wie migriere ich von meinem AL2 auf ein AL2 023 AMI?

Wir empfehlen, einen Migrationsplan zu erstellen und zu implementieren, der gründliche Anwendungs-Workload-Tests und dokumentierte Rollback-Verfahren umfasst. Folgen Sie anschließend den step-by-step Anweisungen im Abschnitt Upgrade von Amazon Linux 2 auf Amazon Linux 2023 in der offiziellen EKS-Dokumentation.

Kann ich ein benutzerdefiniertes AL2 AMI erstellen, das nach dem EKS end-of-support (EOS) -Datum für EKS optimiert ist AL2 AMIs?

Wir empfehlen zwar, auf offiziell unterstütztes und veröffentlichtes EKS umzusteigen, das AMIs für AL2 023 oder Bottlerocket optimiert ist, aber Sie können AMIs bis zum AL2 AL2 AMI-EOS-Datum (26. November 2025) benutzerdefinierte EKS erstellen, die optimiert und AL2 beschleunigt sind. Alternativ können Sie bis zum EOS-Datum von Amazon Linux 2 (30. Juni 2026) eine benutzerdefinierte AMI mit einer Basis-Instance für Amazon Linux 2 erstellen. step-by-stepAnweisungen zum Erstellen eines benutzerdefinierten AL2 EKS-optimierten und AL2 -beschleunigten AMI finden Sie unter Erstellen eines benutzerdefinierten Amazon Linux-AMI in der offiziellen EKS-Dokumentation.

Ist die EKS-Kubernetes-Versionsrichtlinie für Amazon-Linux-Verteilungen anwendbar?

Nein. Das EOS-Datum für EKS AL2 — optimiert und AL2 beschleunigt — AMIs ist unabhängig von den standardmäßigen und erweiterten Supportzeiten für Kubernetes-Versionen von EKS. Sie müssen zu AL2 023 oder Bottlerocket migrieren, auch wenn Sie den erweiterten EKS-Support verwenden.

Wie wirkt sich die Umstellung von cgroupv1 auf cgroupv2 auf meine Migration aus?

Die Kubernetes-Community hat den `cgroupv1` Support (verwendet von AL2) in den Wartungsmodus versetzt, was bedeutet, dass keine neuen Funktionen hinzugefügt werden und nur kritische Sicherheits- und größere Bugfixes bereitgestellt werden. Um `cgroupv2` in Kubernetes zu implementieren, müssen Sie die Kompatibilität zwischen Betriebssystem, Kernel, Container-Laufzeit und Kubernetes-Komponenten sicherstellen. Dies erfordert eine `cgroupv2` standardmäßig aktivierte Linux-Distribution wie AL2 023, Bottlerocket, Red Hat Enterprise Linux (RHEL) 9+, Ubuntu 22.04+ oder Debian 11+. Diese Verteilungen werden mit Kernel-Versionen ≥5.8 bereitgestellt, was die Mindestanforderung für den `cgroupv2`-Support in Kubernetes darstellt. Weitere Informationen finden Sie unter Über cgroup v2.

Was mache ich, wenn ich Neuron in meinem benutzerdefinierten AL2 AMI benötige?

Sie können Ihre vollständigen Neuron-gestützten Anwendungen nicht nativ auf einem Based ausführen. AL2 AMIs Um AWS Neuron auf einem AL2 AMI zu nutzen, müssen Sie Ihre Anwendungen mithilfe eines von NEURON unterstützten Containers mit einer AL2 Nicht-Linux-Distribution (z. B. Ubuntu 22.04, Amazon Linux 2023 usw.) containerisieren und diese Container dann auf einem AL2 basierten AMI bereitstellen, auf dem der Neuron Driver () installiert ist. `aws-neuronx-dkms`

Sollte ich nach dem EKS AL2 AMI EOS-Datum (26. November 2025) zu einer bloßen Amazon Linux 2-Basisinstanz wechseln?

Beim Umstieg auf eine bloße Amazon Linux 2-Basisinstanz fehlen die spezifischen Optimierungen, Container-Laufzeitkonfigurationen und Anpassungen, die von der offiziellen AL2 EKS-Version bereitgestellt werden — optimiert und beschleunigt. AL2 AMIs Wenn Sie weiterhin eine AL2 basierte Lösung verwenden müssen, empfehlen wir stattdessen, ein benutzerdefiniertes AMI mithilfe der EKS AMI-Rezepte unter the section called “Erstellen Sie ein benutzerdefiniertes EKS-optimiertes Amazon Linux-AMI” oder der Amazon EKS AMI Build Specification zu erstellen. Dies gewährleistet die Kompatibilität mit Ihren vorhandenen Workloads und beinhaltet AL2 Kernel-Updates bis zum Amazon Linux 2 EOS-Datum (30. Juni 2026).

Welche Unterstützung ist für Pakete aus GitHub Repositorys wie amzn2-core und amzn2extra-docker verfügbar, wenn ein benutzerdefiniertes AL2 AL2 AMI mithilfe des EKS-AMI-Repositorys nach dem EKS AMI-EOS-Datum (26. November 2025) erstellt wird?

Die EKS-AMI-Anleitung unter Amazon-EKS-AMI-Entwicklungspezifikation zieht Pakete über YUM aus standardmäßiger Amazon Linux 2 Software wie amzn2-core und amzn2extra-docker. Nach dem EKS AL2 AMI EOS-Datum (26. November 2025) wird diese Software bis zum breiteren Amazon Linux 2 EOS-Datum (30. Juni 2026) weiterhin unterstützt. Beachten Sie, dass der Support während dieses Zeitraums auf Kernel-Aktualisierungen beschränkt ist. Das bedeutet, dass Sie andere Paket-Aktualisierungen, Sicherheits-Patches und alle Nicht-Kernel-Abhängigkeiten manuell verwalten und anwenden müssen, um die Sicherheit und Kompatibilität aufrechtzuerhalten.

Warum kann es bei Java-Anwendungen, die ältere Versionen von JDK8 auf Amazon EKS mit AL2 023 verwenden, zu Out-of-Memory-Ausnahmen (OOM) und Pod-Neustarts kommen, und wie kann dieses Problem behoben werden?

Bei der Ausführung auf Amazon EKS-Knoten mit AL2 023 `jdk8u372` können Java-Anwendungen, die auf früheren JDK 8-Versionen basieren, OOM-Ausnahmen und Pod-Neustarts verursachen,

da die JVM nicht kompatibel mit ist. `cgroupv2` Dieses Problem entsteht insbesondere dadurch, dass die JVM-Container-Speichergrenzen mit `cgroupv2`, dem Standard in Amazon Linux 2023, nicht erkennen kann. Daher basiert die Heap-Zuweisung auf dem Gesamtspeicher des Knotens und nicht auf dem definierten Limit des Pods. Dies liegt daran, dass `cgroupv2` den Speicherort für Speicher-Limitdaten ändert, wodurch ältere Java-Versionen den verfügbaren Speicher falsch lesen und Ressourcen auf Knotenebene annehmen. Einige mögliche Optionen sind:

- JDK-Version aktualisieren: Durch ein Upgrade auf `jdk8u372` oder höher oder auf eine neuere JDK-Version mit vollständigem `cgroupv2`-Support kann dieses Problem behoben werden. Eine Liste der kompatiblen Java-Versionen, die `cgroupv2` vollständig unterstützen, finden Sie unter Über cgroup v2.
- Erstellen Sie ein benutzerdefiniertes AMI: Wenn Sie weiterhin eine AL2 basierte Lösung verwenden müssen, können Sie ein benutzerdefiniertes AL2 AMI (bis 26. November 2025) mithilfe the section called “Erstellen Sie ein benutzerdefiniertes EKS-optimiertes Amazon Linux-AMI” der Amazon EKS AMI Build Specification erstellen. Sie können beispielsweise ein AL2 auf Version 1.33 basierendes AMI erstellen (bis 26. November 2025). Amazon EKS wird AMIs bis AL2 zum AL2 EKS EOS-Datum (26. November 2025) stationär zur Verfügung stellen. Nach dem EOS-Datum (26. November 2025) müssen Sie Ihre eigene AMI erstellen.
- Aktivieren Sie cgroupv1: Wenn Sie es weiterhin verwenden müssen`cgroupv1`, können Sie es `cgroupv1` auf einem EKS AL2 023 AMI aktivieren. Um das System zu aktivieren, führen Sie es aus `sudo grubby --update-kernel=ALL --args="systemd.unified_cgroup_hierarchy=0"` und starten Sie es neu (z. B. eine EC2 Instance oder einen Knoten, auf dem Amazon Linux 2023 ausgeführt wird). Dadurch werden die Startparameter für das System geändert (z. B. durch Hinzufügen des Kernel-Parameters „systemd.unified_cgroup_hierarchy=0” zur GRUB-Konfiguration, wodurch systemd angewiesen wird, die veraltete `cgroupv1`-Hierarchie zu verwenden) und `cgroupv1` aktiviert. Beachten Sie, dass Sie mit diesem Grubby-Befehl den Kernel so konfigurieren, dass er mit aktiviertem `cgroupv1` und deaktiviertem `cgroupv2` startet. Nur eine dieser Cgroup-Versionen wird für die aktive Ressourcenverwaltung auf einem Knoten verwendet. Dies ist nicht dasselbe wie die Ausführung von `cgroupv2` mit Abwärtskompatibilität für die `cgroupv1`-API.

> **Warning**
>
> Wir raten von der weiteren Nutzung von `cgroupv1` ab. Stattdessen empfehlen wir die Migration auf `cgroupv2`. Die Kubernetes-Community hat den `cgroupv1` Support (verwendet von AL2) in den Wartungsmodus versetzt, was bedeutet, dass keine neuen Funktionen

oder Updates hinzugefügt werden und nur kritische Sicherheits- und größere Bugfixes bereitgestellt werden. Die vollständige Einstellung des `cgroupv1`-Supports wird in einer zukünftigen Version erwartet. Ein konkretes Datum hierfür wurde jedoch noch nicht bekannt gegeben. Wenn Sie Probleme mit haben`cgroupv1`, können AWS wir Ihnen keinen Support anbieten und Ihnen empfehlen, ein Upgrade auf. `cgroupv2`

Kompatibilität und Versionen

Unterstützte Kubernetes-Versionen für AL2 AMIs

Kubernetes Version 1.32 ist die letzte Version, für die Amazon EKS veröffentlicht wird AL2 (Amazon Linux 2). AMIs Für unterstützte Kubernetes-Versionen bis 1.32 wird EKS bis zum 26. November 2025 weiterhin Versionen AL2 AMIs (AL2_ARM_64, _x86_64) und -beschleunigte Versionen (AL2_x86_64_GPU) veröffentlichen. AL2 AMIs AL2 Nach diesem Datum wird AL2 AL2 EKS AMIs die Veröffentlichung von -optimierten und -beschleunigten Versionen für alle Kubernetes-Versionen einstellen. Beachten Sie, dass das EOS-Datum für EKS AL2 — optimiert und AL2 beschleunigt — unabhängig von den standardmäßigen und erweiterten Supportzeiten für Kubernetes-Versionen von EKS AMIs ist.

Vergleich der unterstützten Treiber und Linux-Kernel-Versionen für, 023 und Bottlerocket AL2 AL2 AMIs

Komponente	AL2 EKS-AMI	EKS AL2 023 AMI	EKS Bottlerocket AMI
Grundlegende Betriebssystemkompatibilität	RHEL7/CentOS 7	Fedora/CentOS 9	–
CUDA-Benutzermodus-Treiber	12.x	12.x, 13.x	12.x, 13.x
NVIDIA-GPU-Treiber	R570	R580	R570, R580
AWS Neuron-Treiber	2,20 +	2,20+	2,20+
Linux-Kernel	5,10	6.1, 6.12	6.1, 6.12

Weitere Informationen zur NVIDIA-Treiber- und CUDA-Kompatibilität finden Sie in der NVIDIA-Dokumentation.

AWS Neuron-Kompatibilität mit AL2 AMIs

Ab AWS der Neuron-Version 2.20 unterstützt die von EKS AL-Based verwendete Neuron Runtime (`aws-neuronx-runtime-lib`) Amazon Linux 2 () nicht AMIs mehr. AL2 Der Neuron-Treiber (`aws-neuronx-dkms`) ist jetzt das einzige AWS Neuron-Paket, das Amazon Linux 2 unterstützt. Das bedeutet, dass Sie Ihre Neuron-gestützten Anwendungen nicht nativ auf einem AL2 AMI ausführen können. Informationen zur Einrichtung von Neuron auf AL2 023 AMIs finden Sie in der Neuron-Setup-Anleitung.AWS

Kubernetes-Kompatibilität mit AL2 AMIs

Die Kubernetes-Community hat den `cgroupv1` Support (verwendet von AL2) in den Wartungsmodus versetzt. Dies bedeutet, dass keine neuen Features hinzugefügt werden und nur kritische Sicherheitsaktualisierungen und wichtige Fehlerbehebungen bereitgestellt werden. Alle Kubernetes-Funktionen, die auf cgroupv2 basieren, wie MemoryQo S und erweiterte Ressourcenisolierung, sind unter nicht verfügbar. AL2 Darüber hinaus war Amazon EKS Kubernetes Version 1.32 die letzte Version, die unterstützt wurde. AL2 AMIs Um die Kompatibilität mit den neuesten Kubernetes-Versionen aufrechtzuerhalten, empfehlen wir die Migration zu AL2 023 oder Bottlerocket, die standardmäßig aktiviert sind. `cgroupv2`

Kompatibilität der Linux-Version mit AL2 AMIs

Amazon Linux 2 (AL2) wird AWS bis zu seinem end-of-support (EOS) Datum am 30. Juni 2026 unterstützt. Mit zunehmendem Alter AL2 wurde der Support für neue Anwendungen und Funktionen durch die breitere Linux-Community jedoch immer eingeschränkter. AL2 AMIs basieren auf dem Linux-Kernel 5.10, während AL2 023 den Linux-Kernel 6.1 verwendet. Im Gegensatz zu AL2 023 wird es AL2 von der breiteren Linux-Community nur begrenzt unterstützt. Das bedeutet, dass viele Upstream-Linux-Pakete und -Tools zurückportiert werden müssen, um mit AL2 der älteren Kernel-Version zu arbeiten, einige moderne Linux-Funktionen und Sicherheitsverbesserungen sind aufgrund des älteren Kernels nicht verfügbar, und viele Open-Source-Projekte unterstützen ältere Kernelversionen wie 5.10 nicht mehr oder nur eingeschränkt.

Veraltete Pakete, die nicht in 0.23 enthalten sind AL2

Zu den gängigsten Paketen, die nicht enthalten sind oder die in Version AL2 023 geändert wurden, gehören:

- Einige Quell-Binärpakete in Amazon Linux 2 sind in Amazon Linux 2023 nicht mehr verfügbar

- Änderungen in der Art und Weise, wie Amazon Linux verschiedene Versionen von Paketen (z. B. amazon-linux-extras System) in AL2 023 unterstützt
- Zusätzliche Pakete für Enterprise Linux (EPEL) werden in Version 023 nicht unterstützt AL2
- 32-Bit-Anwendungen werden in 023 nicht unterstützt AL2

Weitere Informationen finden Sie unter Comparing AL2 und AL2 023.

Vergleich der FIPS-Validierung zwischen AL2 023 AL2 und Bottlerocket

Amazon Linux 2 (AL2), Amazon Linux 2023 (AL2023) und Bottlerocket unterstützen die Einhaltung der Federal Information Processing Standards (FIPS).

- AL2 ist nach FIPS 140-2 zertifiziert und 023 ist nach FIPS 140-3 zertifiziert. AL2 Um den FIPS-Modus auf AL2 023 zu aktivieren, installieren Sie die erforderlichen Pakete auf Ihrer EC2 Amazon-Instance und befolgen Sie die Konfigurationsschritte anhand der Anweisungen unter FIPS-Modus aktivieren auf 023. AL2 Weitere Informationen finden Sie unter 023. AL2 FAQs
- Bottlerocket bietet speziell für FIPS entwickelte Varianten, die den Kernel und die Userspace-Komponenten auf die Verwendung von Kryptografie-Modulen beschränken, die dem FIPS 140-3 Cryptographic Module Validation Program unterzogen wurden.

EKS-AMI-Treiber und Versions-Änderungsprotokoll

Eine vollständige Liste aller EKS AMI-Komponenten und ihrer Versionen finden Sie in den Amazon EKS AMI-Versionshinweisen unter GitHub.

Knoten mit optimiertem Amazon Linux erstellen AMIs

Amazon Elastic Kubernetes Service (Amazon EKS) bietet spezielle Amazon Machine Images (AMIs), die für die Ausführung von Kubernetes-Worker-Knoten optimiert sind. Diese EKS-optimierten Amazon Linux (AL) AMIs sind mit wichtigen Komponenten wie dem AWS IAM Authenticator und vorkonfiguriert`kubelet`, um eine nahtlose Integration und Sicherheit innerhalb Ihrer `containerd` Cluster zu gewährleisten. In diesem Handbuch werden die verfügbaren AMI-Versionen beschrieben und spezielle Optionen für beschleunigtes Rechnen und ARM-basierte Architekturen beschrieben.

Überlegungen

- Sie können Sicherheits- oder Datenschutzereignisse für Amazon Linux im Amazon-Linux-Sicherheitscenter verfolgen, indem Sie die Registerkarte für die gewünschte Version

auswählen. Sie können auch den entsprechenden RSS-Feed abonnieren. Sicherheits- oder Datenschutzereignisse enthalten eine Übersicht über das Problem, welche Pakete betroffen sind und wie Sie Ihre Instances aktualisieren, um das Problem zu beheben.

- Bevor Sie ein beschleunigtes AMI oder ein ARM-AMI bereitstellen, lesen Sie die Informationen in Amazon EKS-optimiertem beschleunigtem Amazon Linux AMIs und. the section called “Amazon EKS-optimierter Arm Amazon Linux AMIs”
- EC2 P2Amazon-Instances werden auf Amazon EKS nicht unterstützt, da sie `NVIDIA` Treiberversion 470 oder früher benötigen.
- Alle neu erstellten verwalteten Knotengruppen in Clustern der Version `1.30` oder neuer verwenden automatisch standardmäßig AL2 023 als Knotenbetriebssystem.

Amazon EKS-optimiertes beschleunigtes Amazon Linux AMIs

Amazon EKS-optimiertes beschleunigtes Amazon Linux (AL) AMIs baut auf dem standardmäßigen EKS-optimierten Amazon Linux auf. AMIs Sie sind so konfiguriert, dass sie als optionale Images für Amazon-EKS-Knoten dienen, um GPU-, Inferentia- und Trainium-basierte Workloads zu unterstützen.

Weitere Informationen finden Sie unter the section called “Verwenden Sie die EKS-Linux-GPU AMIs”.

Amazon EKS-optimierter Arm Amazon Linux AMIs

Arm-Instances führen zu deutlichen Kosteneinsparungen für skalierbare und Arm-basierte Anwendungen wie Webserver, Container-Microservices, Zwischenspeicherflotten und verteilte Datenspeicher. Beachten Sie beim Hinzufügen von Arm-Knoten zu Ihrem Cluster die folgenden Überlegungen.

- Wenn Ihr Cluster vor dem 17. August 2020 bereitgestellt wurde, müssen Sie ein einmaliges Upgrade kritischer Cluster-Add-on-Manifeste durchführen. Dies ist so, dass Kubernetes für jede Hardwarearchitektur, die in Ihrem Cluster verwendet wird, das richtige Image abrufen kann. Weitere Informationen zum Aktualisieren von Cluster-Add-ons finden Sie unter the section called “Schritt 1: Upgrade vorbereiten”. Wenn Sie Ihren Cluster am oder nach dem 17. August 2020 bereitgestellt haben, sind Ihr CoreDNS-, `kube-proxy`- und Amazon VPC-CNI-Plug-in für Kubernetes-Add-Ons bereits Multi-Architektur-fähig.
- Anwendungen, die auf Arm-Knoten bereitgestellt werden, müssen für Arm kompiliert werden.
- Wenn Sie DaemonSets diese in einem vorhandenen Cluster bereitgestellt haben oder sie in einem neuen Cluster bereitstellen möchten, in dem Sie auch ARM-Knoten bereitstellen möchten,

stellen Sie sicher, dass Sie auf allen Hardwarearchitekturen in Ihrem Cluster ausgeführt werden DaemonSet können.

- Sie können Arm-Knotengruppen und x86-Knotengruppen im selben Cluster ausführen. In diesem Fall sollten Sie Container-Images mit mehreren Architekturen in einem Container-Repository wie Amazon Elastic Container Registry bereitstellen und dann Knoten-Selektoren zu Ihren Manifesten hinzufügen, damit Kubernetes weiß, auf welcher Hardwarearchitektur ein Pod bereitgestellt werden kann. Weitere Informationen finden Sie unter Übertragen eines Multi-Architektur-Images im Amazon-ECR-Benutzerhandbuch und im Blogbeitrag Einführung in Multi-Architektur-Container-Images für Amazon ECR.

Weitere Informationen

Weitere Informationen zur Verwendung von Amazon EKS-optimiertem Amazon Linux AMIs finden Sie in den folgenden Abschnitten:

- Informationen zur Verwendung von Amazon Linux mit verwalteten Knotengruppen finden Sie unter the section called “Verwaltete Knotengruppen”.
- Informationen zum Starten selbstverwalteter Amazon-Linux-Knoten finden Sie unter the section called “Holen Sie sich die neuesten IDs”.
- Versionsinformationen finden Sie unter the section called “Versionsinformationen abrufen”.
- Informationen zum Abrufen der neuesten Version IDs des für Amazon EKS optimierten Amazon Linux AMIs finden Sie unter. the section called “Holen Sie sich die neuesten IDs”
- Informationen zu Open-Source-Skripten, die zur Erstellung der Amazon EKS-optimierten AMIs Version verwendet werden, finden Sie unter. the section called “Erstellen Sie ein benutzerdefiniertes EKS-optimiertes Amazon Linux-AMI”

Upgrade von Amazon Linux 2 zu Amazon Linux 2023

> **Warning**
>
> Amazon EKS hat die Veröffentlichung von EKS-optimiertem Amazon Linux 2 (AL2) AMIs am 26. November 2025 eingestellt. AL2023 und Bottlerocket auf Basis von Amazon EKS sind AMIs für alle unterstützten Kubernetes-Versionen einschließlich 1.33 und höher verfügbar.

AL2023 ist ein Linux-basiertes Betriebssystem, das entwickelt wurde, um eine sichere, stabile und leistungsstarke Umgebung für Ihre Cloud-Anwendungen bereitzustellen. Es handelt sich um die nächste Generation von Amazon Linux von Amazon Web Services und ist für alle unterstützten Amazon-EKS-Versionen verfügbar.

AL2023 bietet mehrere Verbesserungen gegenüber. AL2 Einen vollständigen Vergleich finden Sie unter AL2 Comparing and Amazon Linux 2023 im Amazon Linux 2023-Benutzerhandbuch. Es wurden mehrere Pakete hinzugefügt, aktualisiert und entfernt AL2. Es wird dringend empfohlen, Ihre Anwendungen vor dem Upgrade mit AL2 023 zu testen. Eine Liste aller Paketänderungen in Version AL2 023 finden Sie unter Paketänderungen in Amazon Linux 2023 in den Versionshinweisen zu Amazon Linux 2023.

Zusätzlich zu diesen Änderungen sollten Sie Folgendes beachten:

- AL2023 führt einen neuen Knoteninitialisierungsprozess ein`nodeadm`, der ein YAML-Konfigurationsschema verwendet. Wenn Sie selbstverwaltete Knotengruppen oder eine AMI mit einer Startvorlage verwenden, müssen Sie nun beim Erstellen einer neuen Knotengruppe explizit zusätzliche Cluster-Metadaten angeben. Ein Beispiel für die mindestens erforderlichen Parameter ist wie folgt, wobei jetzt `apiServerEndpoint`, `certificateAuthority` und Service-`cidr` erforderlich sind:

  ```
  ---
  apiVersion: node.eks.aws/v1alpha1
  kind: NodeConfig
  spec:
    cluster:
      name: my-cluster
      apiServerEndpoint: https://example.com
      certificateAuthority: Y2VydGlmaWNhdGVBdXRob3JpdHk=
      cidr: 10.100.0.0/16
  ```

  AL2In wurden die Metadaten dieser Parameter aus dem Amazon `DescribeCluster` EKS-API-Aufruf ermittelt. Mit AL2 023 hat sich dieses Verhalten geändert, da durch den zusätzlichen API-Aufruf die Gefahr einer Drosselung bei der Skalierung großer Knoten besteht. Diese Änderung betrifft Sie nicht, wenn Sie verwaltete Knotengruppen ohne Startvorlage verwenden oder wenn Sie Karpenter verwenden. Weitere Informationen zu `certificateAuthority` und Service `cidr` finden Sie unter `DescribeCluster` in der API-Referenz von Amazon EKS.
- Für AL2 023 wird `nodeadm` auch das Format geändert, in dem Parameter `kubelet` für jeden verwendeten Knoten angewendet werden. `NodeConfigSpec` In AL2, das wurde mit dem

`--kubelet-extra-args` Parameter gemacht. Dies wird häufig verwendet, um Knoten mit Beschriftungen und Taints zu versehen. Ein Beispiel unten zeigt die Anwendung von `maxPods` und `--node-labels` auf den Knoten.

```
---
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name: test-cluster
    apiServerEndpoint: https://example.com
    certificateAuthority: Y2VydGlmaWNhdGVBdXRob3JpdHk=
    cidr: 10.100.0.0/16
  kubelet:
    config:
      maxPods: 110
    flags:
      - --node-labels=karpenter.sh/capacity-type=on-demand,karpenter.sh/nodepool=test
```

- Amazon VPC CNI-Version `1.16.2` oder höher ist für AL2 023 erforderlich.
- AL2023 erfordert standardmäßig. `IMDSv2` `IMDSv2`hat mehrere Vorteile, die zur Verbesserung der Sicherheitslage beitragen. Das Programm verwendet eine sitzungsorientierte Authentifizierungsmethode, welche die Erstellung eines geheimen Tokens in einer einfachen HTTP-PUT-Anfrage zum Starten der Sitzung erfordert. Die Gültigkeitsdauer eines Sitzungs-Tokens kann zwischen 1 Sekunde und 6 Stunden variieren. Weitere Informationen zum Übergang von `IMDSv1` zu `IMDSv2` finden Sie unter Umstellung auf Instance Metadata Service Version 2 und Nutzen Sie alle Vorteile IMDSv2 und deaktivieren Sie Ihre IMDSv1 gesamte AWS Infrastruktur. Wenn Sie `IMDSv1` verwenden möchten, können Sie dies dennoch tun, indem Sie die Einstellungen mithilfe der Starteigenschaften der Instance-Metadatenoption manuell überschreiben.

> **Note**
>
> `IMDSv2`Bei Version AL2 023 kann die Standard-Hop-Anzahl für verwaltete Knotengruppen variieren:
>
> - Wenn keine Startvorlage verwendet wird, ist der Standardwert auf 1 eingestellt. Dies bedeutet, dass Container über IMDS keinen Zugriff auf die Anmeldeinformationen des Knotens haben. Wenn Sie Container-Zugriff auf die Anmeldeinformationen des Knotens benötigen, können Sie dies dennoch tun, indem Sie eine benutzerdefinierte EC2 Amazon-Startvorlage verwenden.

- Wenn Sie in einer Startvorlage ein benutzerdefiniertes AMI verwenden, wird das Standard-`HttpPutResponseHopLimit` auf 2 gesetzt. Sie können `HttpPutResponseHopLimit` in der Startvorlage manuell überschreiben.

Alternativ können Sie anstelle von `IMDSv2` Amazon EKS Pod Identity verwenden, um Anmeldeinformationen bereitzustellen.

- AL2023 bietet die nächste Generation einheitlicher Kontrollgruppenhierarchien (`cgroupv2`). `cgroupv2`wird verwendet, um eine Container-Laufzeit zu implementieren, und von`systemd`. AL2023 enthält zwar immer noch Code, mit dem das System ausgeführt werden kann`cgroupv1`, dies ist jedoch keine empfohlene oder unterstützte Konfiguration. Diese Konfiguration wird in einer zukünftigen Hauptversion von Amazon Linux vollständig entfernt.
- `eksctl`Version `0.176.0` oder höher ist erforderlich, um `eksctl` AL2 023 zu unterstützen.

Für bereits bestehende verwaltete Knotengruppen können Sie entweder ein direktes Upgrade oder ein blue/green Upgrade durchführen, je nachdem, wie Sie eine Startvorlage verwenden:

- Wenn Sie ein benutzerdefiniertes AMI mit einer verwalteten Knotengruppe verwenden, können Sie ein direktes Upgrade durchführen, indem Sie die AMI-ID in der Startvorlage austauschen. Sie sollten sicherstellen, dass Ihre Anwendungen und alle Benutzerdaten zuerst auf AL2 023 übertragen werden, bevor Sie diese Upgrade-Strategie durchführen.
- Wenn Sie verwaltete Knotengruppen entweder mit der Standardstartvorlage oder mit einer benutzerdefinierten Startvorlage verwenden, die die AMI-ID nicht angibt, müssen Sie das Upgrade mithilfe einer blue/green Strategie durchführen. Ein blue/green Upgrade ist in der Regel komplexer und beinhaltet die Erstellung einer völlig neuen Knotengruppe, für die Sie AL2 023 als AMI-Typ angeben würden. Die neue Knotengruppe muss anschließend sorgfältig konfiguriert werden, um sicherzustellen, dass alle benutzerdefinierten Daten aus der AL2 Knotengruppe mit dem neuen Betriebssystem kompatibel sind. Sobald die neue Knotengruppe mit Ihren Anwendungen getestet und validiert wurde, können Pods von der alten Knotengruppe auf die neue Knotengruppe migriert werden. Nach Abschluss der Migration können Sie die alte Knotengruppe löschen.

Wenn Sie Karpenter verwenden und AL2 023 verwenden möchten, müssen Sie das `EC2NodeClass` `amiFamily` Feld mit 023 ändern. AL2 Drift ist standardmäßig in Karpenter aktiviert. Dies bedeutet, dass Karpenter nach einer Änderung des `amiFamily`-Feldes Ihre Worker-Knoten automatisch auf die neueste AMI aktualisiert, sobald diese verfügbar ist.

Zusätzliche Informationen zu nodeadm

Wenn Sie ein EKS-optimiertes Amazon Linux 2023 AMI verwenden oder ein benutzerdefiniertes EKS Amazon Linux 2023 AMI mithilfe der im offiziellen amazon-eks-ami GitHub Repository bereitgestellten Packer-Skripts erstellen, sollten Sie es vermeiden, nodeadm init explizit in EC2 Benutzerdaten oder als Teil Ihres benutzerdefinierten AMI auszuführen.

Wenn Sie NodeConfig in Ihren Benutzerdaten Dynamik erzeugen möchten, können Sie diese Konfiguration in eine Drop-In-Yaml- oder JSON-Datei schreiben. `/etc/eks/nodeadm.d` Diese Konfigurationsdateien werden zusammengeführt und auf Ihren Knoten angewendet, wenn nodeadm init später im Startvorgang automatisch gestartet wird. Beispiel:

```
cat > /etc/eks/nodeadm.d/additional-node-labels.yaml << EOF
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  kubelet:
    flags:
      - --node-labels=foo=bar
EOF
```

Das EKS-optimierte Amazon Linux 2023 führt nodeadm init AMIs automatisch in zwei Phasen über separate Systemd-Dienste aus: nodeadm-config wird vor der Ausführung der Benutzerdaten ausgeführt, während nodeadm-run danach ausgeführt wird. Der Dienst nodeadm-config richtet Basiskonfigurationen für containerd und kubelet ein, bevor Benutzerdaten ausgeführt werden. Der Dienst nodeadm-run führt ausgewählte System-Daemons aus und schließt alle endgültigen Konfigurationen nach der Ausführung der Benutzerdaten ab. Wenn der Befehl nodeadm init ein weiteres Mal über Benutzerdaten oder ein benutzerdefiniertes AMI ausgeführt wird, kann er Annahmen über die Reihenfolge der Ausführung widerlegen, was zu unerwarteten Ergebnissen, einschließlich Fehlkonfigurationen, führen kann. ENIs

Abrufen der Versionsinformationen zu Amazon-Linux-AMI

Für Amazon EKS optimierte Amazon-Linux-AMIs werden anhand der Kubernetes-Version und des Veröffentlichungsdatums des AMI in folgendem Format versioniert:

```
k8s_major_version.k8s_minor_version.k8s_patch_version-release_date
```

Jede AMI-Version enthält verschiedene Versionen von kubelet, dem Linux-Kernel und containerd. Die beschleunigten AMIs umfassen auch verschiedene Versionen des NVIDIA-Treibers. Diese Informationen finden Sie im Änderungsprotokoll in GitHub.

Rufen Sie das empfohlene Amazon Linux AMI ab IDs

Beim Bereitstellen von Knoten können Sie eine ID für ein vorkonfiguriertes, für Amazon EKS optimiertes Amazon Machine Image (AMI) angeben. Um eine AMI-ID abzurufen, die Ihrer gewünschten Konfiguration entspricht, fragen Sie die AWS Systems Manager Parameter Store-API ab. Durch die Verwendung dieser API entfällt die Notwendigkeit, manuell nach Amazon EKS-optimiertem AMI zu suchen IDs. Weitere Informationen finden Sie unter GetParameter. Das von Ihnen verwendete IAM-Prinzipal muss über die `ssm:GetParameter`-IAM-Berechtigung zum Abrufen der Amazon EKS-optimierten AMI-Metadaten verfügen.

Sie können die Image-ID des aktuellen empfohlenen für Amazon EKS optimierten Amazon-Linux-AMI mit dem folgenden Befehl abrufen, der den Unterparameter `image_id` verwendet. Nehmen Sie nach Bedarf die folgenden Änderungen am Befehl vor und führen Sie anschließend den geänderten Befehl aus:

- Ersetzen Sie `<kubernetes-version>` durch eine von Amazon EKS unterstützte Version.
- *`ami-type`*Ersetzen Sie es durch eine der folgenden Optionen. Informationen zu den Typen von EC2 Amazon-Instances finden Sie unter EC2 Amazon-Instance-Typen.
  - Wird *`amazon-linux-2023/x86_64/standard`* für Amazon Linux 2023 (AL2023) `x86` - basierte Instances verwendet.
  - Wird *`amazon-linux-2023/arm64/standard`* für AL2 023 ARM-Instances verwendet, z. B. AWS Graviton-basierte Instances.
  - Wird *`amazon-linux-2023/x86_64/nvidia`* für die neuesten genehmigten AL2 023 `x86` NVIDIA-basierten Instanzen verwendet.
  - Wird *`amazon-linux-2023/arm64/nvidia`* für die neuesten genehmigten AL2 023 `arm64` NVIDIA-basierten Instanzen verwendet.
  - Wird *`amazon-linux-2023/x86_64/neuron`* für die neuesten AL2 023 AWS Neuron-Instanzen verwendet.
- `<region-code>`Ersetzen Sie durch eine von Amazon EKS unterstützte AWS Region, für die Sie die AMI-ID benötigen.

```
aws ssm get-parameter --name /aws/service/eks/optimized-ami/<kubernetes-version>/<ami-type>/recommended/image_id \
    --region <region-code> --query "Parameter.Value" --output text
```

Hier ist ein Beispielbefehl, nachdem Platzhalter ersetzt wurden.

```
aws ssm get-parameter --name /aws/service/eks/optimized-ami/1.31/amazon-linux-2023/x86_64/standard/recommended/image_id \
    --region us-west-2 --query "Parameter.Value" --output text
```

Eine Beispielausgabe sieht wie folgt aus.

```
ami-1234567890abcdef0
```

Erstellen Sie ein benutzerdefiniertes EKS-optimiertes Amazon Linux-AMI

> **Warning**
>
> Amazon EKS hat die Veröffentlichung von EKS-optimiertem Amazon Linux 2 (AL2) AMIs am 26. November 2025 eingestellt. AL2023 und Bottlerocket auf Basis von Amazon EKS sind AMIs für alle unterstützten Kubernetes-Versionen einschließlich 1.33 und höher verfügbar.

Amazon EKS bietet Open-Source-Build-Skripte im Amazon EKS AMI Build Specification-Repository, mit denen Sie die Konfigurationen für `kubelet` die Laufzeit und den AWS IAM Authenticator für Kubernetes anzeigen und Ihr eigenes AL-basiertes AMI von Grund auf neu erstellen können.

Dieses Repository enthält das spezielle Bootstrap-Skript für AL2 und das Nodeadm-Tool für 023, das beim Booten ausgeführt wird. AL2 Diese Skripte konfigurieren die Zertifikatsdaten Ihrer Instance, den Endpunkt der Steuerebene, den Cluster-Namen und weitere Elemente. Die Skripts gelten als Informationsquelle für Amazon EKS-optimierte AMI-Builds, sodass Sie dem GitHub Repository folgen können, um Änderungen an unseren zu überwachen. AMIs

Bei der Erstellung benutzerdefinierter Systeme AMIs mit EKS-Optimized AMIs als Basis wird die Ausführung eines Betriebssystem-Upgrades nicht empfohlen oder unterstützt (z. `dnf upgrade`) oder eines der Kubernetes- oder GPU-Pakete aktualisieren, die im EKS-optimierten Paket enthalten sind AMIs, da dadurch die Komponentenkompatibilität beeinträchtigt werden kann. Wenn Sie das

Betriebssystem oder die Pakete, die in EKS-Optimized enthalten sind, aktualisieren, wird empfohlen AMIs, vor der Bereitstellung in der Produktion gründliche Tests in einer Entwicklungs- oder Staging-Umgebung durchzuführen.

Bei der Erstellung benutzerdefinierter GPU-Instanzen empfiehlt es sich, AMIs AMIs für jeden Instance-Typ, jede Generation und Familie, die Sie ausführen möchten, separate benutzerdefinierte Instances zu erstellen. Die für EKS optimierten beschleunigten AMIs Systeme installieren Treiber und Pakete selektiv zur Laufzeit, basierend auf der Generation und Familie des zugrunde liegenden Instance-Typs. Weitere Informationen finden Sie in den EKS AMI-Skripts für Installation und Laufzeit.

Voraussetzungen

- Installieren Sie die AWS CLI
- Installieren Sie HashiCorp Packer v1.9.4+
- Installation von GNU Make

Quickstart

Dieser Schnellstart zeigt Ihnen die Befehle zum Erstellen eines benutzerdefinierten AMI in Ihrem AWS Konto. Weitere Informationen zu den verfügbaren Konfigurationen zur Anpassung Ihrer AMI finden Sie unter den Vorlagenvariablen auf der Seite Amazon Linux 2023.

Voraussetzungen

Installieren Sie das erforderliche Amazon-Plugin. Beispiel:

```
packer plugins install github.com/hashicorp/amazon
```

Schritt 1. Einrichtung Ihrer Umgebung

Klonen oder forken Sie das offizielle Amazon-EKS-AMI-Repository. Beispiel:

```
git clone https://github.com/awslabs/amazon-eks-ami.git
cd amazon-eks-ami
```

Überprüfen Sie, ob Packer installiert ist:

```
packer --version
```

Schritt 2. So erstellen Sie ein benutzerdefiniertes -AMI

Im Folgenden finden Sie Beispielbefehle für verschiedene benutzerdefinierte AMIs Befehle.

Grundlegendes AL2 NVIDIA-AMI:

```
make k8s=1.31 os_distro=al2 \
  enable_accelerator=nvidia \
  nvidia_driver_major_version=560 \
  enable_efa=true
```

Grundlegendes NVIDIA AL2 023 AMI:

```
make k8s=1.31 os_distro=al2023 \
  enable_accelerator=nvidia \
  nvidia_driver_major_version=560 \
  enable_efa=true
```

STIG-konformes Neuron 023 AMI AL2:

```
make k8s=1.31 os_distro=al2023 \
  enable_accelerator=neuron \
  enable_fips=true \
  source_ami_id=ami-0abcd1234efgh5678 \
  kms_key_id=alias/aws-stig
```

Nachdem Sie diese Befehle ausgeführt haben, geht Packer wie folgt vor: * Startet eine temporäre EC2 Amazon-Instance. * Installieren Sie Kubernetes-Komponenten, -Treiber und -Konfigurationen. * Erstellen Sie das AMI in Ihrem AWS Konto.

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
==> Wait completed after 8 minutes 42 seconds

==> Builds finished. The artifacts of successful builds are:
--> amazon-ebs: AMIs were created:
us-west-2: ami-0e139a4b1a7a9a3e9

--> amazon-ebs: AMIs were created:
us-west-2: ami-0e139a4b1a7a9a3e9

--> amazon-ebs: AMIs were created:
```

```
us-west-2: ami-0e139a4b1a7a9a3e9
```

Schritt 3. Standardwerte anzeigen

Um Standardwerte und zusätzliche Optionen anzuzeigen, führen Sie den folgenden Befehl aus:

```
make help
```

Erstellen Sie Knoten mit optimiertem Bottlerocket AMIs

Bottlerocket ist eine Open-Source-Linux-Distribution, die von gesponsert und unterstützt wird. AWS Bottlerocket wurde speziell für das Hosting von Container-Workloads entwickelt. Mit Bottlerocket können Sie die Verfügbarkeit von containerisierten Bereitstellungen verbessern und die Betriebskosten senken, indem Sie Aktualisierungen Ihrer Container-Infrastruktur automatisieren. Bottlerocket enthält ausschließlich die für den Betrieb von Containern erforderliche Software. Dies führt zu einer verbesserten Ressourcennutzung, einer Reduzierung von Sicherheitsrisiken und einem geringeren Verwaltungsaufwand. Das Bottlerocket AMI umfasst`containerd`,`kubelet`, und AWS IAM Authenticator. Neben verwalteten Knotengruppen und selbstverwalteten Knoten wird Bottlerocket auch von Karpenter unterstützt.

Vorteile

Die Verwendung von Bottlerocket mit Ihrem Amazon-EKS-Cluster hat die folgenden Vorteile:

- Höhere Verfügbarkeit bei geringeren Betriebskosten und weniger Verwaltungsaufwand – Bottlerocket benötigt weniger Ressourcen, hat kürzere Startzeiten und ist weniger anfällig für Sicherheitsbedrohungen als andere Linux-Verteilungen. Der geringere Ressourcenbedarf von Bottlerocket trägt zur Kostensenkung bei, da weniger Speicher-, Rechen- und Netzwerkressourcen benötigt werden.
- Verbesserte Sicherheit durch automatische Betriebssystemupdates – Updates für Bottlerocket werden als eine Einheit installiert, die bei Bedarf rückgängig gemacht werden kann. Dadurch wird das Risiko beschädigter oder fehlgeschlagener Updates vermieden, die das System in einen unbrauchbaren Zustand versetzen können. Mit Bottlerocket können Sicherheitsaktualisierungen automatisch angewendet werden, sobald sie verfügbar sind, und das mit minimaler Unterbrechung. Bei Ausfällen können sie zurückgesetzt werden.
- Premium-Support — AWS sofern Versionen von Bottlerocket auf Amazon EC2 verfügbar sind, fallen unter dieselben AWS Support-Pläne, die auch AWS Services wie Amazon EC2, Amazon EKS und Amazon ECR abdecken.

Überlegungen

Beachten Sie bei der Verwendung von Bottlerocket für Ihren AMI-Typ Folgendes:

- Bottlerocket unterstützt EC2 Amazon-Instances mit `x86_64` und Prozessoren. `arm64`
- Bottlerocket unterstützt EC2 Amazon-Instances mit. GPUs Weitere Informationen finden Sie unter the section called “Verwenden Sie die EKS-Linux-GPU AMIs”.
- Bottlerocket-Images enthalten weder einen SSH-Server noch eine Shell. Sie können out-of-band Zugriffsmethoden verwenden, um SSH zuzulassen. Diese Ansätze ermöglichen es dem Administrator-Container, einige Bootstrapping-Konfigurationsschritte mit Benutzerdaten zu übergeben. Weitere Informationen finden Sie in den folgenden Abschnitten von Bottlerocket OS unter: GitHub
  - Exploration (Erkundung)
  - Administrator-Container
  - Kubernetes-Einstellungen
- Bottlerocket verwendet verschiedene Container-Typen:
  - Standardmäßig ist ein Steuerungs-Container aktiviert. In diesem Container wird der AWS Systems Manager Manager-Agent ausgeführt, mit dem Sie Befehle ausführen oder Shell-Sitzungen auf Amazon EC2 Bottlerocket-Instances starten können. Weitere Informationen finden Sie unter Setting up Session Manager im AWS Systems Manager Manager-Benutzerhandbuch.
  - Wenn bei der Erstellung der Knotengruppe ein SSH-Schlüssel angegeben wird, wird ein Admin-Container aktiviert. Wir empfehlen, den Administrator-Container nur für Entwicklungs- und Testszenarien zu verwenden. Es wird nicht empfohlen, ihn in Produktionsumgebungen zu verwenden. Weitere Informationen finden Sie unter Administrator-Container auf GitHub.

Weitere Informationen

Weitere Informationen zur Verwendung des für Amazon EKS optimierten Bottlerocket AMIs finden Sie in den folgenden Abschnitten:

- Einzelheiten zu Bottlerocket finden Sie in der Bottlerocket-Dokumentation.
- Informationen zur Version finden Sie unter the section called “Versionsinformationen abrufen”.
- Informationen zur Verwendung von Bottlerocket mit verwalteten Knotengruppen finden Sie unter the section called “Verwaltete Knotengruppen”.
- Um selbstverwaltete Bottlerocket-Knoten zu starten, lesen Sie the section called “Flaschenrakete”.

- Informationen zum Abrufen der neuesten Version IDs des für Amazon EKS optimierten Bottlerocket AMIs finden Sie unter. the section called “Aktuelle IDs abrufen”
- Einzelheiten zur Compliance-Unterstützung finden Sie unter the section called “Compliance-Support”.

Abrufen von Bottlerocket-AMI-Versionsinformationen

Jede Bottlerocket-AMI-Version umfasst verschiedene Versionen von kubelet, dem Bottlerocket-Kernel und containerd. Beschleunigte AMI-Varianten enthalten auch verschiedene Versionen des NVIDIA-Treibers. Sie finden diese Versionsinformationen im Abschnitt Betriebssystem der Bottlerocket-Dokumentation. Navigieren Sie von dieser Seite zum entsprechenden Unterthema Versionsinformationen.

Die Bottlerocket-Dokumentation kann gelegentlich hinter den auf GitHub verfügbaren Versionen zurückbleiben. Eine Liste der Änderungen für die neuesten Versionen finden Sie in den Versionen auf GitHub.

Empfohlene Bottlerocket-AMI-IDs abrufen

Beim Bereitstellen von Knoten können Sie eine ID für ein vorkonfiguriertes, für Amazon EKS optimiertes Amazon Machine Image (AMI) angeben. Um eine AMI-ID abzurufen, die Ihrer gewünschten Konfiguration entspricht, führen Sie eine Abfrage an die API für AWS Systems Manager Parameter Store durch. Durch die Verwendung dieser API entfällt die Notwendigkeit zur manuellen Suche nach Amazon-EKS-optimierten AMI-IDs. Weitere Informationen finden Sie unter GetParameter. Das von Ihnen verwendete IAM-Prinzipal muss über die `ssm:GetParameter`-IAM-Berechtigung zum Abrufen der Amazon EKS-optimierten AMI-Metadaten verfügen.

Sie können die Image-ID der aktuellsten empfohlenen Amazon-EKS-optimierten Bottlerocket-AMI mit dem folgenden AWS-CLI-Befehl abrufen, der den Unterparameter `image_id` verwendet. Nehmen Sie nach Bedarf die folgenden Änderungen am Befehl vor und führen Sie anschließend den geänderten Befehl aus:

- Ersetzen Sie *`kubernetes-version`* durch eine unterstützte platform-version.
- Ersetzen Sie *`-flavor`* durch eine der folgenden Optionen.
  - Entfernen Sie *`-flavor`* für Varianten ohne GPU
  - Verwenden Sie *`-nvidia`* für GPU-fähige Varianten.
  - Verwenden Sie *`-fips`* für FIPS-fähige Varianten.

- Ersetzen Sie *architecture* durch eine der folgenden Optionen.
  - Verwenden Sie *x86_64* für x86-basierte Instances.
  - Verwenden Sie *arm64* für ARM-Instances.
- Ersetzen Sie *region-code* durch eine Amazon-EKS-unterstützte AWS-Region, für die Sie die AMI-ID verwenden möchten.

```
aws ssm get-parameter --name /aws/service/bottlerocket/aws-k8s-kubernetes-version-
flavor/architecture/latest/image_id \
    --region region-code --query "Parameter.Value" --output text
```

Hier ist ein Beispielbefehl, nachdem Platzhalter ersetzt wurden.

```
aws ssm get-parameter --name /aws/service/bottlerocket/aws-k8s-1.31/x86_64/latest/
image_id \
    --region us-west-2 --query "Parameter.Value" --output text
```

Eine Beispielausgabe sieht wie folgt aus.

```
ami-1234567890abcdef0
```

Compliance-Anforderungen mit Bottlerocket erfüllen

Bottlerocket entspricht den von verschiedenen Organisationen festgelegten Empfehlungen:

- Für Bottlerocket ist ein CIS-Benchmark definiert. In einer Standardkonfiguration verfügt das Bottlerocket-Image über die meisten Steuerelemente, die für das CIS-Level-1-Konfigurationsprofil erforderlich sind. Sie können die für ein CIS-Level-2-Konfigurationsprofil erforderlichen Steuerungen implementieren. Weitere Informationen finden Sie unter Validieren des für Amazon EKS optimierten Bottlerocket-AMIs anhand des CIS-Benchmarks im AWS-Blog.
- Der optimierte Feature-Umfang und die reduzierte Angriffsfläche bedeuten, dass Bottlerocket-Instances weniger Konfiguration erfordern, um die PCI-DSS-Anforderungen zu erfüllen. Der CIS-Benchmark für Bottlerocket ist eine hervorragende Quelle für Hardening-Richtlinien und unterstützt Ihre Anforderungen an sichere Konfigurationsstandards gemäß der PCI-DSS-Anforderung 2.2. Sie können Fluent Bit auch nutzen, um Ihre Anforderungen an die Auditprotokollierung auf Betriebssystemebene gemäß der PCI-DSS-Anforderung 10.2 zu erfüllen. AWS veröffentlicht

regelmäßig neue (gepatchte) Bottlerocket-Instances, damit Sie die PCI-DSS-Anforderungen 6.2 (für v3.2.1) und 6.3.3 (für v4.0) erfüllen können.

- Bottlerocket ist ein HIPAA-fähiges Feature, die für die Verwendung mit regulierten Workloads sowohl für Amazon EC2 als auch für Amazon EKS autorisiert ist. Weitere Informationen finden Sie in der Referenz für HIPAA-berechtigte Services.
- Es sind Bottlerocket-AMIs verfügbar, die für die Verwendung von FIPS 140-3-validierten kryptografischen Modulen vorkonfiguriert sind. Dazu gehören das Amazon Linux 2023 Kernel Crypto API Cryptographic Module und das AWS-LC Cryptographic Module. Weitere Informationen finden Sie unter the section called “Bottlerocket FIPS AMIs”.

Machen Sie Ihre Worker-Nodes FIPS-fähig mit Bottlerocket FIPS AMIs

Die Veröffentlichung 140-3 des Federal Information Processing Standard (FIPS) ist ein Sicherheitsstandard der US- und der kanadischen Regierung, mit dem die Sicherheitsanforderungen für Verschlüsselungsmodule angegeben werden, die vertrauliche Informationen schützen. Bottlerocket erleichtert die Einhaltung von FIPS, indem es mit einem FIPS-Kernel anbietet. AMIs

Diese AMIs sind für die Verwendung von FIPS 140-3-validierten kryptografischen Modulen vorkonfiguriert. Dazu gehören das Amazon Linux 2023 Kernel Cryptographic API Cryptographic Module und das AWS-LC Cryptographic Module.

Durch die Verwendung von Bottlerocket FIPS sind Ihre Worker-Knoten AMIs zwar „FIPS-fähig“, aber nicht automatisch „FIPS-konform“. Weitere Informationen finden Sie unter Federal Information Processing Standard (FIPS) 140-3.

Überlegungen

- Wenn Ihr Cluster isolierte Subnetze verwendet, ist der Amazon-ECR-FIPS-Endpunkt möglicherweise nicht erreichbar. Dies kann dazu führen, dass der Knoten-Bootstrap fehlschlägt. Stellen Sie sicher, dass Ihre Netzwerkkonfiguration den Zugriff auf die erforderlichen FIPS-Endpunkte zulässt. Weitere Informationen finden Sie im AWS PrivateLink Handbuch unter Zugreifen auf eine Ressource über einen VPC-Endpunkt für Ressourcen.
- Wenn Ihr Cluster ein Subnetz mit verwendet PrivateLink, schlagen Image-Pulls fehl, weil Amazon ECR FIPS-Endpunkte nicht über verfügbar sind. PrivateLink

Erstellung einer verwalteten Knotengruppe mit einer Bottlerocket-FIPS-AMI

Das Bottlerocket FIPS AMI ist in vier Varianten erhältlich, um Ihre Workloads zu unterstützen:

- `BOTTLEROCKET_x86_64_FIPS`
- `BOTTLEROCKET_ARM_64_FIPS`
- `BOTTLEROCKET_x86_64_NVIDIA_FIPS`
- `BOTTLEROCKET_ARM_64_NVIDIA_FIPS`

Um eine verwaltete Knotengruppe mit einer Bottlerocket-FIPS-AMI zu erstellen, wählen Sie während des Erstellungsprozesses den entsprechenden AMI-Typ aus. Weitere Informationen finden Sie unter the section called “Create”.

Weitere Informationen zur Auswahl von FIPS-fähigen Varianten finden Sie unter the section called “Aktuelle IDs abrufen”.

Deaktivieren Sie den FIPS-Endpunkt für nicht unterstützte Regionen AWS

Bottlerocket FIPS AMIs werden in den Vereinigte Staaten, einschließlich AWS GovCloud (US-) Regionen, direkt unterstützt. AWS In Regionen, in denen sie verfügbar AMIs sind, aber nicht direkt unterstützt werden, können Sie sie trotzdem verwenden, AMIs indem Sie eine verwaltete Knotengruppe mit einer Startvorlage erstellen.

Das Bottlerocket-FIPS-AMI basiert während des Bootstraps auf dem Amazon ECR-FIPS-Endpunkt, der außerhalb der Vereinigten Staaten in der Regel nicht verfügbar ist. Um das AMI für seinen FIPS-Kernel in AWS Regionen zu verwenden, in denen der Amazon ECR-FIPS-Endpunkt nicht verfügbar ist, gehen Sie wie folgt vor, um den FIPS-Endpunkt zu deaktivieren:

1. Erstellen Sie eine neue Konfigurationsdatei mit dem nachfolgenden Inhalt oder integrieren Sie den Inhalt in Ihre bestehende Konfigurationsdatei.

```
[default]
use_fips_endpoint=false
```

1. Codieren Sie den Dateiinhalt im Base64-Format.
2. Fügen Sie in Ihrer Startvorlage `UserData` die folgende codierte Zeichenfolge im TOML-Format hinzu:

```
[settings.aws]
config = "<your-base64-encoded-string>"
```

Weitere Einstellungen finden Sie in der Beschreibung der Einstellungen von Bottlerocket unter. GitHub

Hier ist ein Beispiel von `UserData` für eine Startvorlage:

```
[settings]
motd = "Hello from eksctl!"
[settings.aws]
config = "W2RlZmF1bHRdCnVzZV9maXBzX2VuZHBvaW50PWZhbHNlCg==" # Base64-encoded string.
[settings.kubernetes]
api-server = "<api-server-endpoint>"
cluster-certificate = "<cluster-certificate-authority>"
cluster-name = "<cluster-name>"
...<other-settings>
```

Weitere Informationen zum Erstellen einer Startvorlage mit Benutzerdaten finden Sie unter the section called “Startvorlagen”.

Erstellung von Knoten mit optimierten Ubuntu-Linux-AMIs

Canonical hat in enger Zusammenarbeit mit Amazon EKS Knoten-AMIs erstellt, die Sie in Ihren Clustern verwenden können.

Canonical stellt ein zu diesem Zweck erstelltes Kubernetes-Knoten-OS-Image bereit. Dieses minimierte Ubuntu-Image ist für Amazon EKS optimiert und enthält den benutzerdefinierten AWS-Kernel, der gemeinsam mit AWS entwickelt wurde. Weitere Informationen finden Sie unter Ubuntu in Amazon Elastic Kubernetes Service (EKS) und the section called “Ubuntu Linux”. Informationen zum Support finden Sie im Abschnitt Software von Drittanbietern in den Häufig gestellten Fragen zum AWS-Premium-Support.

Erstellung von Knoten mit optimierten Windows-AMIs

Amazon-EKS-optimierte Windows-AMIs bauen auf Windows Server 2019 und Windows Server 2022 auf. Sie sind so konfiguriert, dass sie als Basis-Image für Amazon-EKS-Knoten dienen. Standardmäßig enthalten die AMIs die folgenden Komponenten:

- kubelet
- kube-proxy
- AWS IAM Authenticator für Kubernetes
- csi-proxy

- containerd

> **Note**
>
> Sie können Sicherheits- oder Datenschutzereignisse für Windows Server mit dem Microsoft Security Update Guide verfolgen.

Amazon EKS bietet AMIs, die für Windows-Container optimiert sind, in den folgenden Varianten an.

- Amazon-EKS-optimierte Windows-Server-2019-Core-AMI
- Amazon-EKS-optimierte Windows-Server-2019-Full-AMI
- Amazon-EKS-optimierte Windows Server 2022 Core AMI
- Amazon-EKS-optimierte Windows Server 2022 Full AMI

> **Important**
>
> - Die Amazon-EKS-optimierte Windows Server 20H2 Core AMI ist veraltet. Es werden keine neuen Versionen dieses AMI veröffentlicht.
> - Um sicherzustellen, dass Sie standardmäßig über die neuesten Sicherheitsaktualisierungen verfügen, verwaltet Amazon-EKS-optimierte Windows-AMIs für die letzten 4 Monate. Jede neue AMI wird ab dem Zeitpunkt der ersten Veröffentlichung für 4 Monate verfügbar sein. Nach diesem Zeitraum werden ältere AMIs privat und sind nicht mehr zugänglich. Wir empfehlen die Verwendung der neuesten AMIs, um Schwachstellen in der Sicherheit zu vermeiden und den Zugriff auf ältere AMIs, deren Support-Lebensdauer abgelaufen ist, nicht zu verlieren. Wir können zwar nicht garantieren, dass wir Zugriff auf privat gemachte AMIs gewähren können, Sie können jedoch Zugriff anfordern, indem Sie beim AWS Support ein Ticket einreichen.

Veröffentlichungskalender

In der folgenden Tabelle sind die Veröffentlichungs- und Daten für das Ende der Unterstützung für Windows-Versionen auf Amazon EKS aufgeführt. Wenn ein Enddatum leer ist, liegt dies daran, dass die Version weiterhin unterstützt wird.

Windows-Version	Amazon-EKS-Version	Ende der Unterstützung für Amazon EKS
Windows Server 2022 Kern	17.10.2022	
Windows Server 2022 Voll	17.10.2022	
Windows Server 20H2 Core	12.08.2021	09.08.2022
Windows Server 2004 Core	19.08.2020	14.12.2021
Windows Server 2019 Kern	07.10.2019	
Windows Server 2019 Full	07.10.2019	
Windows Server 1909 Core	07.10.2019	08.12.2020

Bootstrap-Skript-Konfigurationsparameter

Wenn Sie einen Windows-Knoten erstellen, gibt es auf dem Knoten ein Skript, mit dem unterschiedliche Parameter konfiguriert werden können. Abhängig von Ihrem Setup kann dieses Skript auf dem Knoten an einem Ort gefunden werden, der ähnlich ist wie: `C:\Program Files\Amazon\EKS\Start-EKSBootstrap.ps1`. Sie können benutzerdefinierte Parameterwerte angeben, indem Sie sie als Argumente für das Bootstrap-Skript angeben. So können Sie beispielsweise die Benutzerdaten in der Startvorlage aktualisieren. Weitere Informationen finden Sie unter the section called “ EC2 Amazon-Benutzerdaten”.

Das Skript enthält die folgenden Befehlszeilenparameter:

- `-EKSClusterName` - Gibt den Namen des Amazon-EKS-Clusters an, dem dieser Worker-Knoten beitreten soll.
- `-KubeletExtraArgs` - Gibt zusätzliche Argumente für `kubelet` an (optional).
- `-KubeProxyExtraArgs` - Gibt zusätzliche Argumente für `kube-proxy` an (optional).
- `-APIServerEndpoint` - Gibt den Amazon-EKS-Cluster-API-Server-Endpunkt an (optional). Nur gültig bei Verwendung mit `-Base64ClusterCA`. Umgehung des Anrufs `Get-EKSCluster`.
- `-Base64ClusterCA` - Gibt den base64-codierten Cluster-CA-Inhalt an (optional). Nur gültig bei Verwendung mit `-APIServerEndpoint`. Umgehung des Anrufs `Get-EKSCluster`.

- `-DNSClusterIP` - Überschreibt die IP-Adresse, die für DNS-Abfragen innerhalb des Clusters verwendet werden soll (optional). Der Standardwert ist `10.100.0.10` oder `172.20.0.10`, basierend auf der IP-Adresse der primären Schnittstelle.
- `-ServiceCIDR` – Überschreibt den Kubernetes-Service-IP-Adressbereich, aus dem Cluster adressiert werden. Der Standardwert ist `172.20.0.0/16` oder `10.100.0.0/16`, basierend auf der IP-Adresse der primären Schnittstelle.
- `-ExcludedSnatCIDRs` – Eine Liste von `IPv4`-CIDRs, die von der Source Network Address Translation (SNAT) ausgeschlossen werden sollen. Das bedeutet, dass die private Pod-IP, die über VPC adressierbar ist, nicht in die IP-Adresse der primären `IPv4`-Adresse der ENI der Instance für ausgehenden Datenverkehr übersetzt wird. Standardmäßig wird das `IPv4`-CIDR der VPC für den Amazon-EKS-Windows-Knoten hinzugefügt. Durch die Angabe von CIDRs für diesen Parameter werden auch die angegebenen CIDRs zusätzlich ausgeschlossen. Weitere Informationen finden Sie unter the section called “Ausgehenden Datenverkehr”.

Zusätzlich zu den Befehlszeilenparametern können Sie auch einige Parameter für Umgebungsvariablen angeben. Wenn Sie einen Befehlszeilenparameter angeben, hat dieser Vorrang vor der jeweiligen Umgebungsvariablen. Die Umgebungsvariablen sollten als Maschinen- (oder System-) Bereich definiert werden, da das Bootstrap-Skript nur maschinenspezifische Variablen liest.

Das Skript berücksichtigt die folgenden Umgebungsvariablen:

- `SERVICE_IPV4_CIDR` – Die Definition finden Sie im `ServiceCIDR`-Befehlszeilenparameter.
- `EXCLUDED_SNAT_CIDRS` – Sollte eine durch Kommas getrennte Zeichenfolge sein. Die Definition finden Sie im `ExcludedSnatCIDRs`-Befehlszeilenparameter.

Support für gMSA-Authentifizierung

Amazon EKS Windows Pods ermöglichen verschiedene Arten der Authentifizierung von Group Managed Service Accounts (gMSA).

- Amazon EKS unterstützt Active-Directory-Domain-Identitäten für die Authentifizierung. Weitere Informationen zu domaingebundenen gMSA finden Sie im AWS-Blogbeitrag Windows-Authentifizierung in Amazon-EKS-Windows-Pods.
- Amazon EKS bietet ein Plugin, das es nicht domaingebundenen Windows-Knoten ermöglicht, gMSA-Anmeldeinformationen mit einer portablen Benutzeridentität abzurufen. Weitere Informationen zu domainlosen gMSA finden Sie im AWS-Blogbeitrag Domainlose Windows-Authentifizierung für Amazon-EKS-Windows-Pods.

Zwischengespeicherte Container-Images

Amazon-EKS-Windows-optimierte AMIs verfügen über bestimmte Container-Images, die für die `containerd`-Laufzeit zwischengespeichert werden. Container-Images werden zwischengespeichert, wenn benutzerdefinierte AMIs mit von Amazon verwalteten Build-Komponenten erstellt werden. Weitere Informationen finden Sie unter the section called “Verwendung der von Amazon verwalteten Entwicklungskomponente”.

Die folgenden zwischengespeicherten Container-Images sind für die `containerd`-Laufzeit bestimmt:

- `amazonaws.com/eks/pause-windows`
- `mcr.microsoft.com/windows/nanoserver`
- `mcr.microsoft.com/windows/servercore`

Weitere Informationen

Weitere Informationen zu Amazon-EKS-optimierten-Windows-AMIs finden Sie in den folgenden Abschnitten:

- Weitere Informationen zum Ausführen von Workloads auf für Amazon EKS optimierten beschleunigten Windows-AMIs finden Sie unter the section called “Windows-GPU-AMIs einrichten”.
- Informationen zur Verwendung von Windows mit verwalteten Knotengruppen finden Sie unter the section called “Verwaltete Knotengruppen”.
- Informationen zum Starten von selbstverwalteten Windows-Knoten finden Sie unter the section called “Windows”.
- Versionsinformationen finden Sie unter the section called “Versionsinformationen abrufen”.
- Informationen zum Abrufen der neuesten IDs der für Amazon EKS optimierten Windows-AMIs finden Sie unter the section called “Aktuelle IDs abrufen”.
- Informationen zum Erstellen benutzerdefinierter, für Amazon EKS optimierter Windows-AMIs mit Amazon EC2 Image Builder finden Sie unter the section called “Benutzerdefinierte Entwicklungen”.
- Bewährte Methoden finden Sie unter Amazon-EKS-optimierte Windows-AMI-Verwaltung im EKS-Leitfaden für bewährte Methoden.

Selbstverwaltete Knoten für Windows Server 2022 mit **`eksctl`** erstellen

Sie können das folgende `test-windows-2022.yaml` als Referenz für die Erstellung selbstverwalteter Knoten für Windows Server 2022 verwenden. Ersetzen Sie jeden *`Beispielwert`* durch Ihre eigenen Werte.

> **Note**
>
> Sie müssen `eksctl`-Version 0.116.0 oder höher verwenden, um selbstverwaltete Knoten für Windows Server 2022 auszuführen.

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: windows-2022-cluster
  region: region-code
  version: '1.33'

nodeGroups:
  - name: windows-ng
    instanceType: m5.2xlarge
    amiFamily: WindowsServer2022FullContainer
    volumeSize: 100
    minSize: 2
    maxSize: 3
  - name: linux-ng
    amiFamily: AmazonLinux2
    minSize: 2
    maxSize: 3
```

Die Knotengruppen können dann mit dem folgenden Befehl erstellt werden.

```
eksctl create cluster -f test-windows-2022.yaml
```

Abrufen der Windows-AMI-Versionsinformationen

In diesem Thema werden Versionen des für Amazon EKS optimierten Windows AMIs und die entsprechenden Versionen von Kubelet, Containerd und csi-proxy aufgeführt.

Die Amazon-EKS-optimierten AMI-Metadaten, einschließlich der AMI-ID für jede Variante, können programmgesteuert abgerufen werden. Weitere Informationen finden Sie unter the section called “Aktuelle IDs abrufen”.

AMIs werden nach Kubernetes-Version und Veröffentlichungsdatum des AMI im folgenden Format versioniert:

```
k8s_major_version.k8s_minor_version-release_date
```

> **Note**
>
> Von Amazon EKS verwaltete Knotengruppen unterstützen die Windows-Versionen November 2022 und spätere Versionen AMIs.

Um Benachrichtigungen über alle Änderungen an den Quelldateien dieser spezifischen Dokumentationsseite zu erhalten, können Sie die folgende URL mit einem RSS-Reader abonnieren:

```
https://github.com/awsdocs/amazon-eks-user-guide/commits/mainline/latest/ug/nodes/eks-ami-versions-windows.adoc.atom
```

Amazon-EKS-optimierte Windows-Server-2022-Core-AMI

In den folgenden Tabellen sind die aktuellen und früheren Versionen des Amazon-EKS-optimierten Windows-Server-2022-Core-AMI aufgeführt.

Example

Kubernetes Version 1.34

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.34-2025.12.15	1.34.2	2.1.4	1.2.1	
1.34-2025.11.14	1.34.1	2.1.4	1.2.1	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.34-2025.10.18	1.34.1	2.1.4	1.2.1	
1.34-2025.09.13	1.34.0	2.1.4	1.2.1	

Kubernetes Version 1.33

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.33-2025.12.15	1.33.5	1.7.28	1.2.1	
1.33-2025.11.14	1.33.5	1.7.28	1.2.1	
1.33-2025.10.18	1.33.5	1.7.28	1.2.1	
1.33-2025.09.13	1.33.4	1.7.28	1.2.1	GMSA-Plug in-Protokolle in Windows-Ereignisse geändert
1.33-2025.08.18	1.33.3	1.7.27	1.2.1	
1.33-2025.07.16	1.33.1	1.7.27	1.2.1	
1.33-2025.06.13	1.33.1	1.7.27	1.2.1	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.33-2025.05.17	1.33.1	1.7.27	1.2.1	

Kubernetes versie 1.32

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.32-2025.12.15	1.32.9	1.7.28	1.2.1	
1.32-2025.11.14	1.32.9	1.7.28	1.2.1	
1.32-2025.10.18	1.32.9	1.7.28	1.2.1	
1.32-2025.09.13	1.32.8	1.7.28	1.2.1	GMSA-Plug in-Protokolle in Windows-Ereignisse geändert
1.32-2025.08.18	1.32.7	1.7.27	1.2.1	
1.32-2025.07.16	1.32.5	1.7.27	1.2.1	
1.32-2025.06.13	1.32.5	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.32-2025.05.17	1.32.5	1.7.20	1.2.1	
1.32-2025.04.14	1.32.1	1.7.20	1.1.3	
1.32-2025.03.14	1.32.1	1.7.20	1.1.3	
1.32-2025.02.18	1.32.1	1.7.20	1.1.3	
1.32-2025.01.15	1.32.0	1.7.20	1.1.3	

Kubernetes versie 1.31

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.31-2025.12.15	1.31.13	1.7.28	1.2.1	
1.31-2025.11.14	1.31.13	1.7.28	1.2.1	
1.31-2025.10.18	1.31.13	1.7.28	1.2.1	
1.31-2025.09.13	1.31.12	1.7.28	1.2.1	GMSA-Plug-in-Protokolle in Windows-Ereignisse geändert

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.31-2025.08.18	1.31.11	1.7.27	1.2.1	
1.31-2025.07.16	1.31.9	1.7.27	1.2.1	
1.31-2025.06.13	1.31.9	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.31-2025.05.17	1.31.9	1.7.20	1.2.1	
1.31-2025.04.14	1.31.5	1.7.20	1.1.3	
1.31-2025.03.14	1.31.5	1.7.20	1.1.3	
1.31-2025.02.15	1.31.5	1.7.20	1.1.3	
1.31-2025.01.15	1.31.4	1.7.20	1.1.3	
1.31-2025.01.01	1.31.4	1.7.20	1.1.3	Beinhaltet Patches für CVE-2024-9042 .
1.31-2024.12.13	1.31.3	1.7.20	1.1.3	
1.31-2024.11.12	1.31.1	1.7.20	1.1.3	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.31-2024.10.08	1.31.1	1.7.20	1.1.3	
1.31-2024.10.01	1.31.1	1.7.20	1.1.3	
1.31-2024.09.10	1.31.0	1.7.20	1.1.3	

Kubernetes versie 1.30

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.30-2025.12.15	1.30.14	1.7.28	1.2.1	
1.30-2025.11.21	1.30.14	1.7.28	1.2.1	
1.30-2025.10.18	1.30.14	1.7.28	1.2.1	
1.30-2025.09.13	1.30.14	1.7.28	1.2.1	GMSA-Plug in-Protokolle in Windows-Ereignisse geändert
1.30-2025.08.18	1.30.14	1.7.27	1.2.1	
1.30-2025.07.16	1.30.13	1.7.27	1.2.1	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.30-2025.06.13	1.30.13	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.30-2025.05.17	1.30.13	1.7.20	1.2.1	
1.30-2025.04.14	1.30.9	1.7.20	1.1.3	
1.30-2025.03.14	1.30.9	1.7.20	1.1.3	containerd wurde auf 1.7.20 aktualisiert.
1.30-2025.02.15	1.30.9	1.7.14	1.1.3	
1.30-2025.01.15	1.30.8	1.7.14	1.1.3	
1.30-2025.01.01	1.30.8	1.7.14	1.1.3	Beinhaltet Patches für CVE-2024-9042 .
1.30-2024.12.11	1.30.7	1.7.14	1.1.3	
1.30-2024.11.12	1.30.4	1.7.14	1.1.3	
1.30-2024.10.08	1.30.4	1.7.14	1.1.3	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.30-2024.09.10	1.30.2	1.7.14	1.1.3	
1.30-2024.08.13	1.30.2	1.7.14	1.1.3	
1.30-2024.07.10	1.30.2	1.7.14	1.1.2	Beinhaltet Patches für CVE-2024-5321 .
1.30-2024.06.17	1.30.0	1.7.14	1.1.2	containerd wurde auf 1.7.14 aktualisiert.
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Kubernetes versie 1.29

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.29-2025.12.15	1.29.15	1.7.28	1.2.1	
1.29-2025.11.14	1.29.15	1.7.28	1.2.1	
1.29-2025.10.18	1.29.15	1.7.28	1.2.1	
1.29-2025.09.13	1.29.15	1.7.28	1.2.1	GMSA-Plug in-Protokolle

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
				in Windows-Ereignisse geändert
1.29-2025.08.18	1.29.15	1.7.27	1.2.1	
1.29-2025.07.16	1.29.15	1.7.27	1.2.1	
1.29-2025.06.13	1.29.15	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.29-2025.05.17	1.29.15	1.7.20	1.2.1	
1.29-2025.04.14	1.29.13	1.7.20	1.1.3	
1.29-2025.03.14	1.29.13	1.7.20	1.1.3	containerd wurde auf 1.7.20 aktualisiert.
1.29-2025.02.15	1.29.13	1.7.14	1.1.3	
1.29-2025.01.15	1.29.12	1.7.14	1.1.3	
1.29-2025.01.01	1.29.12	1.7.14	1.1.3	Beinhaltet Patches für CVE-2024-9042 .

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.29-2024.12.11	1.29.10	1.7.14	1.1.3	
1.29-2024.11.12	1.29.8	1.7.14	1.1.3	
1.29-2024.10.08	1.29.8	1.7.14	1.1.3	
1.29-2024.09.10	1.29.6	1.7.14	1.1.3	
1.29-2024.08.13	1.29.6	1.7.14	1.1.3	
1.29-2024.07.10	1.29.6	1.7.11	1.1.2	Beinhaltet Patches für CVE-2024-5321 .
1.29-2024.06.17	1.29.3	1.7.11	1.1.2	
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	containerd wurde auf 1.7.11 aktualisiert. kubelet wurde auf 1.29.3 aktualisiert.

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	containerd wurde auf 1.6.28 aktualisiert. CNI und csi-proxy unter Verwendung von golang 1.22.1 neu erstellt.
1.29-2024.03.12	1.29.0	1.6.25	1.1.2	
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	Ein Fehler wurde behoben, bei dem das Pause-Image durch den kubelet-Garbage-Collection-Prozess fälschlicherweise gelöscht wurde.

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.29-2024.01.11	1.29.0	1.6.18	1.1.2	Ausgeschlossen ist das eigenständige Windows-Update KB5034439 auf Windows Server 2022 Core. AMIs Die KB gilt nur für Windows-Installationen mit einer separaten WinRE-Partition, die in keinem unserer für Amazon EKS optimierten Windows AMIs enthalten sind.

Kubernetes Version 1.28

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2025.11.14	1.28.15	1.7.28	1.2.1	
1.28-2025.10.18	1.28.15	1.7.28	1.2.1	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2025.09.13	1.28.15	1.7.28	1.2.1	GMSA-Plugin-Protokolle in Windows-Ereignisse geändert
1.28-2025.08.18	1.28.15	1.7.27	1.2.1	
1.28-2025.07.16	1.28.15	1.7.27	1.2.1	
1.28-2025.06.13	1.28.15	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.28-2025.05.17	1.28.15	1.7.20	1.2.1	
1.28-2025.04.14	1.28.15	1.7.20	1.1.3	
1.28-2025.03.14	1.28.15	1.7.20	1.1.3	containerd wurde auf 1.7.20 aktualisiert.
1.28-2025.02.15	1.28.15	1.7.14	1.1.3	
1.28-2025.01.15	1.28.15	1.7.14	1.1.3	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2025-01-01	1.28.15	1.7.14	1.1.3	Beinhaltet Patches für CVE-2024-9042 .
1.28-2024.12.11	1.28.15	1.7.14	1.1.3	
1.28-2024.11.12	1.28.13	1.7.14	1.1.3	
1.28-2024.10.08	1.28.13	1.7.14	1.1.3	
1.28-2024.09.10	1.28.11	1.7.14	1.1.3	
1.28-2024.08.13	1.28.11	1.7.14	1.1.3	
1.28-2024.07.10	1.28.11	1.7.11	1.1.2	Beinhaltet Patches für CVE-2024-5321 .
1.28-2024.06.17	1.28.8	1.7.11	1.1.2	containerd wurde auf 1.7.11 aktualisiert.

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	containerd wurde auf 1.6.28 aktualisiert. kubelet wurde auf 1.28.8 aktualisiert.
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	containerd wurde auf 1.6.25 aktualisiert. CNI und csi-proxy unter Verwendung von golang 1.22.1 neu erstellt.
1.28-2024.03.12	1.28.5	1.6.18	1.1.2	
1.28-2024.02.13	1.28.5	1.6.18	1.1.2	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2024.01.11	1.28.5	1.6.18	1.1.2	Ausgeschlossen ist das eigenständige Windows-Update KB5034439 auf Windows Server 2022 Core. AMIs Die KB gilt nur für Windows-Installationen mit einer separaten WinRE-Partition, die in keinem unserer für Amazon EKS optimierten Windows AMIs enthalten sind.
1.28-2023.12.12	1.28.3	1.6.18	1.1.2	
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	Beinhaltet Patches für CVE-2023-5528 .

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	containerd wurde auf 1.6.18 aktualisiert. Neue Umgebungsvariablen für Bootstrap-Skripte hinzugefügt (SERVICE_IPV4_CIDR und EXCLUDED_SNAT_CIDRS).
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	Ein Security Advisory wurde in kubelet behoben.
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

Amazon-EKS-optimierte Windows-Server-2022-Full-AMI

In den folgenden Tabellen sind die aktuellen und früheren Versionen des Amazon-EKS-optimierten Windows-Server-2022-Full-AMI aufgeführt.

Example

Kubernetes Version 1.34

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.34-2025.12.15	1.34.2	2.1.4	1.2.1	
1.34-2025.11.14	1.34.1	2.1.4	1.2.1	
1.34-2025.10.18	1.34.1	2.1.4	1.2.1	
1.34-2025.09.13	1.34.0	2.1.4	1.2.1	

Kubernetes Version 1.33

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.33-2025.12.15	1.33.5	1.7.28	1.2.1	
1.33-2025.11.14	1.33.5	1.7.28	1.2.1	
1.33-2025.10.18	1.33.5	1.7.28	1.2.1	
1.33-2025.09.13	1.33.4	1.7.28	1.2.1	GMSA-Plug-in-Protokolle in Windows-Ereignisse geändert

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.33-2025.08.18	1.33.3	1.7.27	1.2.1	
1.33-2025.07.16	1.33.1	1.7.27	1.2.1	
1.33-2025.06.13	1.33.1	1.7.27	1.2.1	
1.33-2025.05.17	1.33.1	1.7.27	1.2.1	

Kubernetes versie 1.32

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.32-2025.12.15	1.32.9	1.7.28	1.2.1	
1.32-2025.11.14	1.32.9	1.7.28	1.2.1	
1.32-2025.10.18	1.32.9	1.7.28	1.2.1	
1.32-2025.09.13	1.32.8	1.7.28	1.2.1	GMSA-Plug-in-Protokolle in Windows-Ereignisse geändert
1.32-2025.08.18	1.32.7	1.7.27	1.2.1	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.32-2025.07.16	1.32.5	1.7.27	1.2.1	
1.32-2025.06.13	1.32.5	1.7.27	1.2.1	containrd auf 1.7.27 aktualisiert
1.32-2025.05.17	1.32.5	1.7.20	1.2.1	
1.32-2025.04.14	1.32.1	1.7.20	1.1.3	
1.32-2025.03.14	1.32.1	1.7.20	1.1.3	
1.32-2025.02.18	1.32.1	1.7.20	1.1.3	
1.32-2025.01.01	1.32.0	1.7.20	1.1.3	

Kubernetes versie 1.31

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.31-2025.12.15	1.31.13	1.7.28	1.2.1	
1.31-2025.11.14	1.31.13	1.7.28	1.2.1	
1.31-2025.10.18	1.31.13	1.7.28	1.2.1	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.31-2025.09.13	1.31.12	1.7.28	1.2.1	GMSA-Plugin-Protokolle in Windows-Ereignisse geändert
1.31-2025.08.18	1.31.11	1.7.27	1.2.1	
1.31-2025.07.16	1.31.9	1.7.27	1.2.1	
1.31-2025.06.13	1.31.9	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.31-2025.05.17	1.31.9	1.7.20	1.2.1	
1.31-2025.04.14	1.31.5	1.7.20	1.1.3	
1.31-2025.03.14	1.31.5	1.7.20	1.1.3	
1.31-2025.02.15	1.31.5	1.7.20	1.1.3	
1.31-2025.01.15	1.31.4	1.7.20	1.1.3	
1.31-2025.01.01	1.31.4	1.7.20	1.1.3	Beinhaltet Patches für CVE-2024-9042 .

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.31-2024.12.13	1.31.3	1.7.20	1.1.3	
1.31-2024.11.12	1.31.1	1.7.20	1.1.3	
1.31-2024.10.08	1.31.1	1.7.20	1.1.3	
1.31-2024.10.01	1.31.1	1.7.20	1.1.3	
1.31-2024.09.10	1.31.0	1.7.20	1.1.3	

Kubernetes versie 1.30

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.30-2025.12.15	1.30.14	1.7.28	1.2.1	
1.30-2025.11.21	1.30.14	1.7.28	1.2.1	
1.30-2025.10.18	1.30.14	1.7.28	1.2.1	
1.30-2025.09.13	1.30.14	1.7.28	1.2.1	GMSA-Plug in-Protokolle in Windows-Ereignisse geändert

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.30-2025.08.18	1.30.14	1.7.27	1.2.1	
1.30-2025.07.16	1.30.13	1.7.27	1.2.1	
1.30-2025.06.13	1.30.13	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.30-2025.05.17	1.30.13	1.7.20	1.2.1	
1.30-2025.04.14	1.30.9	1.7.20	1.1.3	
1.30-2025.03.14	1.30.9	1.7.20	1.1.3	contianerd wurde auf 1.7.20 aktualisiert.
1.30-2025.02.15	1.30.9	1.7.14	1.1.3	
1.30-2025.01.15	1.30.8	1.7.14	1.1.3	
1.30-2025.01.01	1.30.8	1.7.14	1.1.3	Beinhaltet Patches für CVE-2024-9042 .
1.30-2024.12.11	1.30.7	1.7.14	1.1.3	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.30-2024.11.12	1.30.4	1.7.14	1.1.3	
1.30-2024.10.08	1.30.4	1.7.14	1.1.3	
1.30-2024.09.10	1.30.2	1.7.14	1.1.3	
1.30-2024.08.13	1.30.2	1.7.14	1.1.3	
1.30-2024.07.10	1.30.2	1.7.14	1.1.2	Beinhaltet Patches für CVE-2024-5321 .
1.30-2024.06.17	1.30.0	1.7.14	1.1.2	containerd wurde auf 1.7.14 aktualisiert.
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Kubernetes versie 1.29

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.29-2025.12.15	1.29.15	1.7.28	1.2.1	
1.29-2025.11.14	1.29.15	1.7.28	1.2.1	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.29-2025.10.18	1.29.15	1.7.28	1.2.1	
1.29-2025.09.13	1.29.15	1.7.28	1.2.1	GMSA-Plug in-Protokolle in Windows-Ereignisse geändert
1.29-2025.08.18	1.29.15	1.7.27	1.2.1	
1.29-2025.07.16	1.29.15	1.7.27	1.2.1	
1.29-2025.06.13	1.29.15	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.29-2025.05.17	1.29.15	1.7.20	1.2.1	
1.29-2025.04.14	1.29.13	1.7.20	1.1.3	
1.29-2025.03.14	1.29.13	1.7.20	1.1.3	containerd wurde auf 1.7.20 aktualisiert.
1.29-2025.02.15	1.29.13	1.7.14	1.1.3	
1.29-2025.01.15	1.29.12	1.7.14	1.1.3	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.29-2025.01.01	1.29.12	1.7.14	1.1.3	Beinhaltet Patches für CVE-2024-9042 .
1.29-2024.12.11	1.29.10	1.7.14	1.1.3	
1.29-2024.11.12	1.29.8	1.7.14	1.1.3	
1.29-2024.10.08	1.29.8	1.7.14	1.1.3	
1.29-2024.09.10	1.29.6	1.7.14	1.1.3	
1.29-2024.08.13	1.29.6	1.7.14	1.1.3	
1.29-2024.07.10	1.29.6	1.7.11	1.1.2	Beinhaltet Patches für CVE-2024-5321 .
1.29-2024.06.17	1.29.3	1.7.11	1.1.2	
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	containerd wurde auf 1.7.11 aktualisiert. kubelet wurde auf 1.29.3 aktualisiert.

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	containerd wurde auf 1.6.28 aktualisiert. CNI und csi-proxy unter Verwendung von golang 1.22.1 neu erstellt.
1.29-2024.03.12	1.29.0	1.6.25	1.1.2	
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	Ein Fehler wurde behoben, bei dem das Pause-Image durch den kubelet-Garbage-Collection-Prozess fälschlicherweise gelöscht wurde.
1.29-2024.01.09	1.29.0	1.6.18	1.1.2	

Kubernetes versie 1.28

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2025.11.14	1.28.15	1.7.28	1.2.1	
1.28-2025.10.18	1.28.15	1.7.28	1.2.1	
1.28-2025.09.13	1.28.15	1.7.28	1.2.1	GMSA-Plug in-Protokolle in Windows-Ereignisse geändert
1.28-2025.08.18	1.28.15	1.7.27	1.2.1	
1.28-2025.07.16	1.28.15	1.7.27	1.2.1	
1.28-2025.06.13	1.28.15	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.28-2025.05.17	1.28.15	1.7.20	1.2.1	
1.28-2025.04.14	1.28.15	1.7.20	1.1.3	
1.28-2025.03.14	1.28.15	1.7.20	1.1.3	containerd wurde auf 1.7.20 aktualisiert.

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2025.02.15	1.28.15	1.7.14	1.1.3	
1.28-2025.01.15	1.28.15	1.7.14	1.1.3	
1.28-2025.01.01	1.28.15	1.7.14	1.1.3	Beinhaltet Patches für CVE-2024-9042 .
1.28-2024.12.11	1.28.15	1.7.14	1.1.3	
1.28-2024.11.12	1.28.13	1.7.14	1.1.3	
1.28-2024.10.08	1.28.13	1.7.14	1.1.3	
1.28-2024.09.10	1.28.11	1.7.14	1.1.3	
1.28-2024.08.13	1.28.11	1.7.14	1.1.3	
1.28-2024.07.10	1.28.11	1.7.11	1.1.2	Beinhaltet Patches für CVE-2024-5321 .
1.28-2024.06.17	1.28.8	1.7.11	1.1.2	containerd wurde auf 1.7.11 aktualisiert.

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	containerd wurde auf 1.6.28 aktualisiert. kubelet wurde auf 1.28.8 aktualisiert.
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	containerd wurde auf 1.6.25 aktualisiert. CNI und csi-proxy unter Verwendung von golang 1.22.1 neu erstellt.
1.28-2024.03.12	1.28.5	1.6.18	1.1.2	
1.28-2024.02.13	1.28.5	1.6.18	1.1.2	
1.28-2024.01.09	1.28.5	1.6.18	1.1.2	
1.28-2023.12.12	1.28.3	1.6.18	1.1.2	
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	Beinhaltet Patches für CVE-2023-5528 .

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	containerd wurde auf 1.6.18 aktualisiert. Neue Umgebungsvariablen für Bootstrap-Skripte hinzugefügt (SERVICE_IPV4_CIDR und EXCLUDED_SNAT_CIDRS).
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	Ein Security Advisory wurde in kubelet behoben.
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

Amazon-EKS-optimierte Windows-Server-2019-Core-AMI

In den folgenden Tabellen sind die aktuellen und früheren Versionen des Amazon-EKS-optimierten Windows-Server-2019-Core-AMI aufgeführt.

Example

Kubernetes versie 1.34

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.34-2025.12.15	1.34.2	2.1.4	1.2.1	
1.34-2025.11.14	1.34.1	2.1.4	1.2.1	
1.34-2025.10.18	1.34.1	2.1.4	1.2.1	
1.34-2025.09.13	1.34.0	2.1.4	1.2.1	

Kubernetes Version 1.33

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.33-2025.12.15	1.33.5	1.7.28	1.2.1	
1.33-2025.11.14	1.33.5	1.7.28	1.2.1	
1.33-2025.10.18	1.33.5	1.7.28	1.2.1	
1.33-2025.09.13	1.33.4	1.7.28	1.2.1	GMSA-Plug-in-Protokolle in Windows-Ereignisse geändert

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.33-2025.08.18	1.33.3	1.7.27	1.2.1	
1.33-2025.07.16	1.33.1	1.7.27	1.2.1	
1.33-2025.06.13	1.33.1	1.7.27	1.2.1	
1.33-2025.05.17	1.33.1	1.7.27	1.2.1	

Kubernetes versie 1.32

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.32-2025.12.15	1.32.9	1.7.28	1.2.1	
1.32-2025.11.14	1.32.9	1.7.28	1.2.1	
1.32-2025.10.18	1.32.9	1.7.28	1.2.1	
1.32-2025.09.13	1.32.8	1.7.28	1.2.1	GMSA-Plug-in-Protokolle in Windows-Ereignisse geändert
1.32-2025.08.18	1.32.7	1.7.27	1.2.1	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.32-2025.07.16	1.32.5	1.7.27	1.2.1	
1.32-2025.06.13	1.32.5	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.32-2025.05.17	1.32.5	1.7.20	1.2.1	
1.32-2025.04.14	1.32.1	1.7.20	1.1.3	
1.32-2025.03.14	1.32.1	1.7.20	1.1.3	
1.32-2025.02.18	1.32.1	1.7.20	1.1.3	
1.32-2025.01.15	1.32.4	1.7.20	1.1.3	

Kubernetes versie 1.31

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.31-2025.12.15	1.31.13	1.7.28	1.2.1	
1.31-2025.11.14	1.31.13	1.7.28	1.2.1	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.31-2025.10.18	1.31.13	1.7.28	1.2.1	
1.31-2025.09.13	1.31.12	1.7.28	1.2.1	GMSA-Plug in-Protokolle in Windows-Ereignisse geändert
1.31-2025.08.18	1.31.11	1.7.27	1.2.1	
1.31-2025.07.16	1.31.9	1.7.27	1.2.1	
1.31-2025.06.13	1.31.9	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.31-2025.05.17	1.31.9	1.7.20	1.2.1	
1.31-2025.04.14	1.31.5	1.7.20	1.1.3	
1.31-2025.03.14	1.31.5	1.7.20	1.1.3	
1.31-2025.02.15	1.31.5	1.7.20	1.1.3	
1.31-2025.01.15	1.31.4	1.7.20	1.1.3	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.31-2025.01.01	1.31.4	1.7.20	1.1.3	Beinhaltet Patches für CVE-2024-9042 .
1.31-2024.12.13	1.31.3	1.7.20	1.1.3	
1.31-2024.11.12	1.31.1	1.7.20	1.1.3	
1.31-2024.10.08	1.31.1	1.7.20	1.1.3	
1.31-2024.10.01	1.31.1	1.7.20	1.1.3	
1.31-2024.09.10	1.31.0	1.7.20	1.1.3	

Kubernetes versie 1.30

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.30-2025.12.15	1.30.14	1.7.28	1.2.1	
1.30-2025.11.21	1.30.14	1.7.28	1.2.1	
1.30-2025.10.18	1.30.14	1.7.28	1.2.1	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.30-2025.09.13	1.30.14	1.7.28	1.2.1	GMSA-Plugin-Protokolle in Windows-Ereignisse geändert
1.30-2025.08.18	1.30.14	1.7.27	1.2.1	
1.30-2025.07.16	1.30.13	1.7.27	1.2.1	
1.30-2025.06.13	1.30.13	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.30-2025.05.17	1.30.13	1.7.20	1.2.1	
1.30-2025.04.14	1.30.9	1.7.20	1.1.3	
1.30-2025.03.14	1.30.9	1.7.20	1.1.3	containerd wurde auf 1.7.20 aktualisiert.
1.30-2025-02-15	1.30.9	1.7.14	1.1.3	
1.30-2025.01.15	1.30.8	1.7.14	1.1.3	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.30-2025.01.01	1.30.8	1.7.14	1.1.3	Beinhaltet Patches für CVE-2024-9042 .
1.30-2024.12.11	1.30.7	1.7.14	1.1.3	
1.30-2024.11.12	1.30.4	1.7.14	1.1.3	
1.30-2024.10.08	1.30.4	1.7.14	1.1.3	
1.30-2024.09.10	1.30.2	1.7.14	1.1.3	
1.30-2024.08.13	1.30.2	1.7.14	1.1.3	
1.30-2024.07.10	1.30.2	1.7.14	1.1.2	Beinhaltet Patches für CVE-2024-5321 .
1.30-2024.06.17	1.30.0	1.7.14	1.1.2	containerd wurde auf 1.7.14 aktualisiert.
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Kubernetes versie 1.29

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.29-2025.12.15	1.29.15	1.7.28	1.2.1	
1.29-2025.11.14	1.29.15	1.7.28	1.2.1	
1.29-2025.10.18	1.29.15	1.7.28	1.2.1	
1.29-2025.09.13	1.29.15	1.7.28	1.2.1	GMSA-Plug in-Protokolle in Windows-Ereignisse geändert
1.29-2025.08.18	1.29.15	1.7.27	1.2.1	
1.29-2025.07.16	1.29.15	1.7.27	1.2.1	
1.29-2025.06.13	1.29.15	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.29-2025.05.17	1.29.15	1.7.20	1.2.1	
1.29-2025.04.14	1.29.13	1.7.20	1.1.3	
1.29-2025.03.14	1.29.13	1.7.20	1.1.3	containerd wurde

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
				auf `1.7.20` aktualisiert.
`1.29-2025.02.15`	`1.29.13`	`1.7.14`	`1.1.3`	
`1.29-2025.01.15`	`1.29.12`	`1.7.14`	`1.1.3`	
`1.29-2025.01.01`	`1.29.12`	`1.7.14`	`1.1.3`	Beinhaltet Patches für `CVE-2024-9042` .
`1.29-2024.12.11`	`1.29.10`	`1.7.14`	`1.1.3`	
`1.29-2024.11.12`	`1.29.8`	`1.7.14`	`1.1.3`	
`1.29-2024.10.08`	`1.29.8`	`1.7.14`	`1.1.3`	
`1.29-2024.09.10`	`1.29.6`	`1.7.14`	`1.1.3`	
`1.29-2024.08.13`	`1.29.6`	`1.7.14`	`1.1.3`	
`1.29-2024.07.10`	`1.29.6`	`1.7.11`	`1.1.2`	Beinhaltet Patches für `CVE-2024-5321` .
`1.29-2024.06.17`	`1.29.3`	`1.7.11`	`1.1.2`	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	containerd wurde auf 1.7.11 aktualisiert. kubelet wurde auf 1.29.3 aktualisiert.
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	containerd wurde auf 1.6.28 aktualisiert. CNI und csi-proxy unter Verwendung von golang 1.22.1 neu erstellt.
1.29-2024.03.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	Ein Fehler wurde behoben, bei dem das Pause-Image durch den kubelet-Garbage-Collection-Prozess fälschlicherweise gelöscht wurde.
1.29-2024.01.09	1.29.0	1.6.18	1.1.2	

Kubernetes versie 1.28

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2025.11.14	1.28.15	1.7.28	1.2.1	
1.28-2025.10.18	1.28.15	1.7.28	1.2.1	
1.28-2025.09.13	1.28.15	1.7.28	1.2.1	GMSA-Plugin-Protokolle in Windows-Ereignisse geändert

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2025.08.18	1.28.15	1.7.27	1.2.1	
1.28-2025.07.16	1.28.15	1.7.27	1.2.1	
1.28-2025.06.13	1.28.15	1.7.20	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.28-2025.05.17	1.28.15	1.7.20	1.2.1	
1.28-2025.04.14	1.28.15	1.7.20	1.1.3	
1.28-2025.03.14	1.28.15	1.7.20	1.1.3	containerd wurde auf 1.7.20 aktualisiert.
1.28-2025.02.15	1.28.15	1.7.14	1.1.3	
1.28-2025-01-15	1.28.15	1.7.14	1.1.3	
1.28-2025-01-01	1.28.15	1.7.14	1.1.3	Beinhaltet Patches für CVE-2024-9042 .
1.28-2024.12.11	1.28.15	1.7.14	1.1.3	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2024.11.12	1.28.13	1.7.14	1.1.3	
1.28-2024.10.08	1.28.13	1.7.14	1.1.3	
1.28-2024.09.10	1.28.11	1.7.14	1.1.3	
1.28-2024.08.13	1.28.11	1.7.14	1.1.3	
1.28-2024.07.10	1.28.11	1.7.11	1.1.2	Beinhaltet Patches für CVE-2024-5321 .
1.28-2024.06.17	1.28.8	1.7.11	1.1.2	containerd wurde auf 1.7.11 aktualisiert.
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	containerd wurde auf 1.6.28 aktualisiert. kubelet wurde auf 1.28.8 aktualisiert.

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	containerd wurde auf 1.6.25 aktualisiert. CNI und csi-proxy unter Verwendung von golang 1.22.1 neu erstellt.
1.28-2024.03.13	1.28.5	1.6.18	1.1.2	
1.28-2024.02.13	1.28.5	1.6.18	1.1.2	
1.28-2024.01.09	1.28.5	1.6.18	1.1.2	
1.28-2023.12.12	1.28.3	1.6.18	1.1.2	
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	Beinhaltet Patches für CVE-2023-5528 .

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	containerd wurde auf 1.6.18 aktualisiert. Neue Umgebungsvariablen für Bootstrap-Skripte hinzugefügt (SERVICE_IPV4_CIDR und EXCLUDED_SNAT_CIDRS).
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	Ein Security Advisory wurde in kubelet behoben.
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

Amazon-EKS-optimierte Windows-Server-2019-Full-AMI

In den folgenden Tabellen sind die aktuellen und früheren Versionen des Amazon-EKS-optimierten Windows-Server-2019-Full-AMI aufgeführt.

Example

Kubernetes versie 1.34

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.34-2025.12.15	1.34.2	2.1.4	1.2.1	
1.34-2025.11.14	1.34.1	2.1.4	1.2.1	
1.34-2025.10.18	1.34.1	2.1.4	1.2.1	
1.34-2025.09.13	1.34.0	2.1.4	1.2.1	

Kubernetes Version 1.33

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.33-2025.12.15	1.33.5	1.7.28	1.2.1	
1.33-2025.11.14	1.33.5	1.7.28	1.2.1	
1.33-2025.10.18	1.33.5	1.7.28	1.2.1	
1.33-2025.09.13	1.33.4	1.7.28	1.2.1	GMSA-Plug-in-Protokolle in Windows-Ereignisse geändert

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.33-2025.08.18	1.33.3	1.7.27	1.2.1	
1.33-2025.07.16	1.33.1	1.7.27	1.2.1	
1.33-2025.06.13	1.33.1	1.7.27	1.2.1	
1.33-2025.05.17	1.33.1	1.7.27	1.2.1	

Kubernetes versie 1.32

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.32-2025.12.15	1.32.9	1.7.28	1.2.1	
1.32-2025.11.14	1.32.9	1.7.28	1.2.1	
1.32-2025.10.18	1.32.9	1.7.28	1.2.1	
1.32-2025.09.13	1.32.8	1.7.28	1.2.1	GMSA-Plug-in-Protokolle in Windows-Ereignisse geändert
1.32-2025.08.18	1.32.7	1.7.27	1.2.1	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.32-2025.07.16	1.32.5	1.7.27	1.2.1	
1.32-2025.06.13	1.32.5	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.32-2025.05.17	1.32.5	1.7.20	1.2.1	
1.32-2025.04.14	1.32.1	1.7.20	1.1.3	
1.32-2025.03.14	1.32.1	1.7.20	1.1.3	
1.32-2025.02.18	1.32.1	1.7.20	1.1.3	
1.32-2025.01.15	1.32.0	1.7.20	1.1.3	

Kubernetes versie 1.31

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.31-2025.12.15	1.31.13	1.7.28	1.2.1	
1.31-2025.11.14	1.31.13	1.7.28	1.2.1	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.31-2025.10.18	1.31.13	1.7.28	1.2.1	
1.31-2025.09.13	1.31.12	1.7.28	1.2.1	GMSA-Plug in-Protokolle in Windows-Ereignisse geändert
1.31-2025.08.18	1.31.11	1.7.27	1.2.1	
1.31-2025.07.16	1.31.9	1.7.27	1.2.1	
1.31-2025.06.13	1.31.9	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.31-2025.05.17	1.31.9	1.7.20	1.2.1	
1.31-2025.04.14	1.31.5	1.7.20	1.1.3	
1.31-2025.03.14	1.31.5	1.7.20	1.1.3	
1.31-2025.02.15	1.31.5	1.7.20	1.1.3	
1.31-2025.01.15	1.31.4	1.7.20	1.1.3	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.31-2025.01.01	1.31.4	1.7.20	1.1.3	Beinhaltet Patches für CVE-2024-9042 .
1.31-2024.12.13	1.31.3	1.7.20	1.1.3	
1.31-2024.11.12	1.31.1	1.7.20	1.1.3	
1.31-2024.10.08	1.31.1	1.7.20	1.1.3	
1.31-2024.10.01	1.31.1	1.7.20	1.1.3	
1.31-2024.09.10	1.31.0	1.7.20	1.1.3	

Kubernetes versie 1.30

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.30-2025.12.15	1.30.14	1.7.28	1.2.1	
1.30-2025.11.21	1.30.14	1.7.28	1.2.1	
1.30-2025.10.18	1.30.14	1.7.28	1.2.1	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.30-2025.09.13	1.30.14	1.7.28	1.2.1	GMSA-Plugin-Protokolle in Windows-Ereignisse geändert
1.30-2025.08.18	1.30.14	1.7.27	1.2.1	
1.30-2025.07.16	1.30.13	1.7.27	1.2.1	
1.30-2025.06.13	1.30.13	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.30-2025.05.17	1.30.13	1.7.20	1.2.1	
1.30-2025.04.14	1.30.9	1.7.20	1.1.3	
1.30-2025.03.14	1.30.9	1.7.20	1.1.3	containerd wurde auf 1.7.20 aktualisiert.
1.30-2025.02.15	1.30.9	1.7.14	1.1.3	
1.30-2025.01.15	1.30.8	1.7.14	1.1.3	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.30-2025.01.01	1.30.8	1.7.14	1.1.3	Beinhaltet Patches für CVE-2024-9042 .
1.30-2024.12.11	1.30.7	1.7.14	1.1.3	
1.30-2024.11.12	1.30.4	1.7.14	1.1.3	
1.30-2024.10.08	1.30.4	1.7.14	1.1.3	
1.30-2024.09.10	1.30.2	1.7.14	1.1.3	
1.30-2024.08.13	1.30.2	1.7.14	1.1.3	
1.30-2024.07.10	1.30.2	1.7.14	1.1.2	Beinhaltet Patches für CVE-2024-5321 .
1.30-2024.06.17	1.30.0	1.7.14	1.1.2	containerd wurde auf 1.7.14 aktualisiert.
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Kubernetes versie 1.29

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.29-2025.12.15	1.29.15	1.7.28	1.2.1	
1.29-2025.11.14	1.29.15	1.7.28	1.2.1	
1.29-2025.10.18	1.29.15	1.7.28	1.2.1	
1.29-2025.09.13	1.29.15	1.7.28	1.2.1	GMSA-Plug in-Protokolle in Windows-Ereignisse geändert
1.29-2025.08.18	1.29.15	1.7.27	1.2.1	
1.29-2025.07.16	1.29.15	1.7.27	1.2.1	
1.29-2025.06.13	1.29.15	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.29-2025.05.17	1.29.15	1.7.20	1.2.1	
1.29-2025.04.14	1.29.13	1.7.20	1.1.3	
1.29-2025.03.14	1.29.13	1.7.20	1.1.3	containerd wurde

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
				auf `1.7.20` aktualisiert.
`1.29-2025.02.15`	`1.29.13`	`1.7.14`	`1.1.3`	
`1.29-2025.01.15`	`1.29.12`	`1.7.14`	`1.1.3`	
`1.29-2025.01.01`	`1.29.12`	`1.7.14`	`1.1.3`	Beinhaltet Patches für `CVE-2024-9042` .
`1.29-2024.12.11`	`1.29.10`	`1.7.14`	`1.1.3`	
`1.29-2024.11.12`	`1.29.8`	`1.7.14`	`1.1.3`	
`1.29-2024.10.08`	`1.29.8`	`1.7.14`	`1.1.3`	
`1.29-2024.09.10`	`1.29.6`	`1.7.14`	`1.1.3`	
`1.29-2024.08.13`	`1.29.6`	`1.7.14`	`1.1.3`	
`1.29-2024.07.10`	`1.29.6`	`1.7.11`	`1.1.2`	Beinhaltet Patches für `CVE-2024-5321` .
`1.29-2024.06.17`	`1.29.3`	`1.7.11`	`1.1.2`	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	containerd wurde auf 1.7.11 aktualisiert. kubelet wurde auf 1.29.3 aktualisiert.
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	containerd wurde auf 1.6.28 aktualisiert. CNI und csi-proxy unter Verwendung von golang 1.22.1 neu erstellt.
1.29-2024.03.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	Ein Fehler wurde behoben, bei dem das Pause-Image durch den kubelet-Garbage-Collection-Prozess fälschlicherweise gelöscht wurde.
1.29-2024.01.09	1.29.0	1.6.18	1.1.2	

Kubernetes versie 1.28

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2025.11.14	1.28.15	1.7.28	1.2.1	
1.28-2025.10.18	1.28.15	1.7.28	1.2.1	
1.28-2025.09.13	1.28.15	1.7.28	1.2.1	GMSA-Plugin-Protokolle in Windows-Ereignisse geändert

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2025.08.18	1.28.15	1.7.27	1.2.1	
1.28-2025.07.16	1.28.15	1.7.27	1.2.1	
1.28-2025.06.13	1.28.15	1.7.27	1.2.1	containerd wurde auf 1.7.27 aktualisiert.
1.28-2025.05.17	1.28.15	1.7.20	1.2.1	
1.28-2025.04.14	1.28.15	1.7.20	1.1.3	
1.28-2025.03.14	1.28.15	1.7.20	1.1.3	containerd wurde auf 1.7.20 aktualisiert.
1.28-2025.02.15	1.28.15	1.7.14	1.1.3	
1.28-2025-01-15	1.28.15	1.7.14	1.1.3	
1.28-2025-01-01	1.28.15	1.7.14	1.1.3	Beinhaltet Patches für CVE-2024-9042 .
1.28-2024.12.11	1.28.15	1.7.14	1.1.3	

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2024.11.12	1.28.13	1.7.14	1.1.3	
1.28-2024.10.08	1.28.13	1.7.14	1.1.3	
1.28-2024.09.10	1.28.11	1.7.14	1.1.3	
1.28-2024.08.13	1.28.11	1.7.14	1.1.3	
1.28-2024.07.10	1.28.11	1.7.11	1.1.2	Beinhaltet Patches für CVE-2024-5321 .
1.28-2024.06.17	1.28.8	1.7.11	1.1.2	containerd wurde auf 1.7.11 aktualisiert.
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	containerd wurde auf 1.6.28 aktualisiert. kubelet wurde auf 1.28.8 aktualisiert.

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	containerd wurde auf 1.6.25 aktualisiert. CNI und csi-proxy unter Verwendung von golang 1.22.1 neu erstellt.
1.28-2024.03.13	1.28.5	1.6.18	1.1.2	
1.28-2024.02.13	1.28.5	1.6.18	1.1.2	
1.28-2024.01.09	1.28.5	1.6.18	1.1.2	
1.28-2023.12.12	1.28.3	1.6.18	1.1.2	
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	Beinhaltet Patches für CVE-2023-5528 .

AMI-Version	kubelet-Version	containerd-Version	csi-proxy-Version	Versionshinweise
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	containerd wurde auf 1.6.18 aktualisiert. Neue Umgebungsvariablen für Bootstrap-Skripte hinzugefügt (SERVICE_IPV4_CIDR und EXCLUDED_SNAT_CIDRS).
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	Ein Security Advisory wurde in kubelet behoben.
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

Empfohlene Microsoft-Windows-AMI-IDs abrufen

Beim Bereitstellen von Knoten können Sie eine ID für ein vorkonfiguriertes, für Amazon EKS optimiertes Amazon Machine Image (AMI) angeben. Um eine AMI-ID abzurufen, die Ihrer gewünschten Konfiguration entspricht, führen Sie eine Abfrage an die API für AWS Systems Manager Parameter Store durch. Durch die Verwendung dieser API entfällt die Notwendigkeit zur manuellen Suche nach Amazon-EKS-optimierten AMI-IDs. Weitere Informationen finden Sie unter GetParameter. Das von Ihnen verwendete IAM-Prinzipal muss über die `ssm:GetParameter`-IAM-Berechtigung zum Abrufen der Amazon EKS-optimierten AMI-Metadaten verfügen.

Sie können die Image-ID des neuesten empfohlenen, für Amazon EKS optimierten Windows-AMI mit dem folgenden Befehl abrufen, der den Unterparameter `image_id` verwendet. Nehmen Sie nach Bedarf die folgenden Änderungen am Befehl vor und führen Sie anschließend den geänderten Befehl aus:

- Ersetzen Sie *`release`* durch eine der folgenden Optionen.
  - Verwenden Sie *`2022`* für Windows Server 2022.
  - Verwenden Sie *`2019`* für Windows Server 2019.
- Ersetzen Sie *`installation-option`* durch eine der folgenden Optionen. Weitere Informationen finden Sie unter Was ist die Server-Core-Installationsoption in Windows Server?.
  - Verwenden Sie *`Core`* für eine minimale Installation mit einer geringeren Angriffsfläche.
  - Verwenden Sie *`Full`*, um die Windows-Desktop-Erfahrung einzuschließen.
- Ersetzen Sie *`kubernetes-version`* durch eine unterstützte platform-version.
- Ersetzen Sie *`region-code`* durch eine Amazon-EKS-unterstützte AWS-Region, für die Sie die AMI-ID verwenden möchten.

```
aws ssm get-parameter --name /aws/service/ami-windows-latest/Windows_Server-release-
English-installation-option-EKS_Optimized-kubernetes-version/image_id \
    --region region-code --query "Parameter.Value" --output text
```

Hier ist ein Beispielbefehl, nachdem Platzhalter ersetzt wurden.

```
aws ssm get-parameter --name /aws/service/ami-windows-latest/Windows_Server-2022-
English-Core-EKS_Optimized-k8s-n-2/image_id \
    --region us-west-2 --query "Parameter.Value" --output text
```

Eine Beispielausgabe sieht wie folgt aus.

```
ami-1234567890abcdef0
```

Entwicklung einer benutzerdefinierten Windows-AMI mit Image Builder

Sie können EC2 Image Builder verwenden, um benutzerdefinierte, für Amazon EKS optimierte Windows AMIs mit einer der folgenden Optionen zu erstellen:

- Verwendung einer für Amazon EKS optimierten Windows-AMI als Grundlage

- Verwendung der von Amazon verwalteten Entwicklungskomponente

Bei beiden Methoden müssen Sie Ihr eigenes Image-Builder-Rezept erstellen. Weitere Informationen finden Sie unter Erstellen einer neuen Version eines Bildrezeptes im Image-Builder-Benutzerhandbuch.

> **Important**
>
> Die folgenden von Amazon verwalteten Komponenten für `eks` enthalten Patches für `CVE-2024-5321`.
>
> - `1.28.2` und höher
> - `1.29.2` und höher
> - `1.30.1` und höher
> - Alle Versionen für Kubernetes 1.31 und höher

Verwendung einer für Amazon EKS optimierten Windows-AMI als Grundlage

Diese Option ist die empfohlene Methode, um Ihr benutzerdefiniertes Windows zu erstellen AMIs. Das von AMIs uns bereitgestellte Amazon EKS-optimierte Windows wird häufiger aktualisiert als die von Amazon verwaltete Build-Komponente.

1. Starten Sie ein neues Image-Builder-Rezept.
   a. Öffnen Sie die EC2 Image Builder Builder-Konsole unter https://console.aws.amazon.com / imagebuilder.
   b. Wählen Sie im linken Navigationsbereich Image recipes (Image-Rezepte) aus.
   c. Wählen Sie Create image recipe (Image-Rezept erstellen) aus.
2. Geben Sie im Abschnitt Recipe details (Rezeptdetails) einen Namen und eine Version ein.
3. Geben Sie die ID der für Amazon EKS optimierten Windows-AMI im Abschnitt Basis-Image an.
   a. Wählen Sie Enter custom AMI ID (Benutzerdefinierte AMI-ID eingeben) aus.
   b. Rufen Sie die AMI-ID für die benötigte Windows-Betriebssystemversion ab. Weitere Informationen finden Sie unter the section called “Aktuelle IDs abrufen”.
   c. Geben Sie die benutzerdefinierte AMI-ID ein. Wenn die AMI-ID nicht gefunden wird, stellen Sie sicher, dass die AWS Region für die AMI-ID mit der AWS Region übereinstimmt, die oben rechts auf Ihrer Konsole angezeigt wird.

4. (Optional) Um die neuesten Sicherheitsupdates zu erhalten, fügen Sie die `update-windows`-Komponente im Abschnitt Build components – (Build-Komponenten –) hinzu.
   a. Wählen Sie in der Dropdownliste rechts neben dem Suchfeld Find components by name (Komponenten nach Namen suchen) Amazon-managed (Von Amazon verwaltet) aus.
   b. Geben Sie im Suchfeld Find components by name (Komponenten nach Namen suchen) `update-windows` ein.
   c. Markieren Sie das Kontrollkästchen des **`update-windows`**-Suchergebnisses. Diese Komponente schließt die neuesten Windows-Patches für das Betriebssystem ein.
5. Füllen Sie die verbleibenden Image-Rezepteingaben mit Ihren erforderlichen Konfigurationen aus. Weitere Informationen finden Sie unter Erstellen einer neuen Version eines Image-Rezeptes (Konsole) im Image-Builder-Benutzerhandbuch.
6. Wählen Sie Create Recipe (Rezept erstellen) aus.
7. Verwenden Sie das neue Image-Rezept in einer neuen oder vorhandenen Image-Pipeline. Sobald Ihre Image-Pipeline erfolgreich ausgeführt wurde, wird Ihr benutzerdefiniertes AMI als Ausgabe-Image aufgeführt und ist einsatzbereit. Weitere Informationen finden Sie unter Erstellen einer Image-Pipeline mit dem EC2 Image Builder Builder-Konsolenassistenten.

Verwendung der von Amazon verwalteten Entwicklungskomponente

Wenn die Verwendung eines für Amazon EKS optimierten Windows-AMI als Grundlage nicht praktikabel ist, können Sie stattdessen die von Amazon verwaltete Entwicklungskomponente verwenden. Diese Option kann hinter den aktuellsten unterstützten Kubernetes-Versionen zurückbleiben.

1. Starten Sie ein neues Image-Builder-Rezept.
   a. Öffnen Sie die EC2 Image Builder Builder-Konsole unter https://console.aws.amazon.com / imagebuilder.
   b. Wählen Sie im linken Navigationsbereich Image recipes (Image-Rezepte) aus.
   c. Wählen Sie Create image recipe (Image-Rezept erstellen) aus.
2. Geben Sie im Abschnitt Recipe details (Rezeptdetails) einen Namen und eine Version ein.
3. Legen Sie im Abschnitt Base image (Basis-Image) fest, welche Option Sie verwenden werden, um Ihr benutzerdefiniertes AMI zu erstellen:
   - Select managed images (Verwaltete Images auswählen) – Wählen Sie Windows als Ihr Image Operating System (OS) (Image-Betriebssystem (OS)) aus. Wählen Sie dann eine der folgenden Optionen für Image origin (Image-Ursprung) aus:

- Schnellstart (von Amazon verwaltet) – Wählen Sie in der Dropdown-Liste für den Image-Namen eine von Amazon EKS unterstützte Windows-Server-Version aus. Weitere Informationen finden Sie unter the section called “Windows”.
- Images owned by me (Bilder in meinem Besitz) – Wählen Sie für Image-Name den ARN Ihres eigenen Image mit Ihrer eigenen Lizenz aus. Auf dem von Ihnen bereitgestellten Image können Amazon-EKS-Komponenten nicht bereits installiert sein.

- Enter custom AMI ID (Benutzerdefinierte AMI-ID eingeben) – Geben Sie für AMI-ID die ID für Ihr AMI mit Ihrer eigenen Lizenz ein. Auf dem von Ihnen bereitgestellten Image können Amazon-EKS-Komponenten nicht bereits installiert sein.

4. Gehen Sie im Abschnitt Build components – Windows (Komponenten erstellen – Windows) wie folgt vor:
   a. Wählen Sie in der Dropdownliste rechts neben dem Suchfeld Find components by name (Komponenten nach Namen suchen) Amazon-managed (Von Amazon verwaltet) aus.
   b. Geben Sie im Suchfeld Find components by name (Komponenten nach Namen suchen) `eks` ein.
   c. Markieren Sie das Kästchen des Suchergebnisses **`eks-optimized-ami-windows`**, auch wenn das zurückgegebene Ergebnis möglicherweise nicht die gewünschte Version ist.
   d. Geben Sie im Suchfeld Find components by name (Komponenten nach Namen suchen) `update-windows` ein.
   e. Markieren Sie das Kontrollkästchen des Suchergebnisses update-windows. Diese Komponente schließt die neuesten Windows-Patches für das Betriebssystem ein.
5. Gehen Sie im Abschnitt Selected components (Ausgewählte Komponenten) wie folgt vor:
   a. Wählen Sie Optionen zur Versionsverwaltung für **`eks-optimized-ami-windows`** aus.
   b. Wählen Sie Specify component versio (Komponentenversion angeben) aus.
   c. Geben Sie im Feld Komponentenversion Folgendes ein: Ersetzen durch eine *`version.x`* unterstützte *`version`* Kubernetes-Version. Wenn Sie *`x`* für einen Teil der Versionsnummer eingeben, wird die neueste Komponentenversion verwendet, die auch mit dem Teil der Version übereinstimmt, den Sie explizit definieren. Achten Sie auf die Konsolenausgabe, da sie Sie darüber informiert, ob Ihre gewünschte Version als verwaltete Komponente verfügbar ist. Beachten Sie, dass die neuesten Kubernetes-Versionen möglicherweise nicht für die Entwicklungskomponente verfügbar sind. Weitere Informationen zu den verfügbaren Versionen finden Sie unter the section called “Abrufen von Informationen zu `eks-optimized-ami-windows`-Komponentenversionen”.

6. Füllen Sie die verbleibenden Image-Rezepteingaben mit Ihren erforderlichen Konfigurationen aus. Weitere Informationen finden Sie unter Erstellen einer neuen Version eines Image-Rezeptes (Konsole) im Image-Builder-Benutzerhandbuch.
7. Wählen Sie Create Recipe (Rezept erstellen) aus.
8. Verwenden Sie das neue Image-Rezept in einer neuen oder vorhandenen Image-Pipeline. Sobald Ihre Image-Pipeline erfolgreich ausgeführt wurde, wird Ihr benutzerdefiniertes AMI als Ausgabe-Image aufgeführt und ist einsatzbereit. Weitere Informationen finden Sie unter Erstellen einer Image-Pipeline mit dem EC2 Image Builder Builder-Konsolenassistenten.

Abrufen von Informationen zu **`eks-optimized-ami-windows`**-Komponentenversionen

Sie können spezifische Informationen darüber abrufen, was mit den einzelnen Komponenten installiert ist. Sie können beispielsweise überprüfen, welche `kubelet`-Version installiert ist. Die Komponenten durchlaufen Funktionstests auf den von Amazon EKS unterstützten Windows-Betriebssystemversionen. Weitere Informationen finden Sie unter the section called “Veröffentlichungskalender”. Alle anderen Windows-Betriebssystemversionen, die nicht aufgelistet sind oder deren Support eingestellt wird, sind möglicherweise nicht mit der Komponente kompatibel.

1. Öffnen Sie die EC2 Image Builder Builder-Konsole unter https://console.aws.amazon.com / imagebuilder.
2. Wählen Sie im linken Navigationsbereich Components (Komponenten) aus.
3. Ändern Sie in der Dropdownliste rechts neben dem Suchfeld Find components by name (Komponenten nach Namen suchen) Owned by me (In meinem Besitz) zu Quick start (Amazon-managed) (Schnellstart (Von Amazon verwaltet)).
4. Geben Sie im Feld Komponenten nach Name suchen `eks` ein.
5. (Optional) Wenn Sie eine aktuelle Version verwenden, sortieren Sie die Spalte Version in absteigender Reihenfolge, indem Sie sie zweimal auswählen.
6. Wählen Sie den Link **`eks-optimized-ami-windows`** mit der gewünschten Version aus.

Unter Description (Beschreibung) auf der resultierenden Seite werden die spezifischen Informationen angezeigt.

Aktivieren der automatischen Knotenreparatur und Untersuchen von Problemen mit dem Zustand des Knotens

Der Knotenzustand bezieht sich auf den Betriebsstatus und die Fähigkeit eines Knotens, Workloads effektiv auszuführen. Ein fehlerfreier Knoten verfügt über die erwartete Konnektivität, ausreichende Ressourcen und kann Pods ohne Unterbrechungen erfolgreich ausführen. Informationen zum Abrufen von Details zu Ihren Knoten finden Sie unter the section called “Knotenzustand anzeigen” und the section called “Knotenprotokolle abrufen”.

Um die Aufrechterhaltung eines einwandfreien Zustands der Knoten zu unterstützen, bietet Amazon EKS den Knoten-Überwachungsagenten und die automatische Knotenreparatur an.

> **Important**
>
> Der Knoten-Überwachungsagente und die Knotenreparatur sind nur in Linux verfügbar. Diese Features sind unter Windows nicht verfügbar.

Knotenüberwachungsagent

Der Knoten-Überwachungsagent liest automatisch Knotenprotokolle, um Zustandsprobleme zu erkennen. Er analysiert Knotenprotokolle, um Fehler zu erkennen und verschiedene Statusinformationen über Worker-Knoten anzuzeigen. Für jede Kategorie der erkannten Probleme, wie beispielsweise Speicher- und Netzwerkprobleme, wird eine dedizierte `NodeCondition` auf die Worker-Knoten angewendet. Beschreibungen der erkannten Statusprobleme werden im Beobachtbarkeits-Dashboard zur Verfügung gestellt. Weitere Informationen finden Sie unter the section called “Probleme mit dem Zustand des Knotens”.

Der Knoten-Überwachungsagent ist als Funktion für alle Cluster von Amazon EKS Auto Mode enthalten. Für andere Cluster-Typen können Sie den Überwachungsagenten als Amazon-EKS-Add-On hinzufügen. Weitere Informationen finden Sie unter the section called “Add-On erstellen”.

Automatische Knotenreparatur

Die automatische Knotenreparatur ist ein zusätzliches Feature, das den Zustand der Knoten kontinuierlich überwacht, automatisch auf erkannte Probleme reagiert und Knoten nach Möglichkeit ersetzt. Dies unterstützt die Gesamtverfügbarkeit des Clusters mit minimalen manuellen Eingriffen.

Wenn eine Zustandsprüfung fehlschlägt, wird der Knoten automatisch gesperrt, sodass keine neuen Pods auf dem Knoten geplant werden.

Die automatische Knotenreparatur kann von sich aus auf den `Ready`-Zustand des `kubelet` und aller manuell gelöschten Knotenobjekte reagieren. In Kombination mit dem Knoten-Überwachungsagenten kann die automatische Knotenreparatur auf mehr Zustände reagieren, die anderenfalls nicht erkannt würden. Zu diesen zusätzlichen Zuständen gehören `KernelReady`, `NetworkingReady` und `StorageReady`.

Diese automatische Knotenwiederherstellung behebt automatisch intermittierende Knotenprobleme wie Fehler beim Beitritt zum Cluster, nicht reagierende Kubelets und eine zunehmende Zahl von Accelerator-Fehlern (Gerätefehler). Die erhöhte Zuverlässigkeit trägt dazu bei, Ausfallzeiten von Anwendungen zu reduzieren und den Cluster-Betrieb zu verbessern. Die automatische Knotenreparatur kann bestimmte gemeldete Probleme wie `DiskPressure`, `MemoryPressure` und `PIDPressure` nicht verarbeiten. Amazon EKS wartet 10 Minuten, bevor es auf `AcceleratedHardwareReady` `NodeConditions` reagiert, und 30 Minuten bei allen anderen Bedingungen.

Verwaltete Knotengruppen deaktivieren Knotenreparaturen aus Sicherheitsgründen in zwei Szenarien automatisch. Alle bereits laufenden Reparaturarbeiten werden in beiden Fällen fortgesetzt.

- Wenn eine Zonenverschiebung für Ihren Cluster über den Application Recovery Controller (ARC) ausgelöst wurde, werden alle nachfolgenden Reparaturvorgänge angehalten.
- Wenn Ihre Knotengruppe mehr als fünf Knoten umfasst und mehr als 20 % der Knoten in Ihrer Knotengruppe sich in einem fehlerhaften Zustand befinden, werden die Reparaturvorgänge angehalten.

Sie können die automatische Knotenreparatur beim Erstellen oder Bearbeiten einer verwalteten Knotengruppe aktivieren.

- Bei Verwendung der Amazon-EKS-Konsole aktivieren Sie das Kontrollkästchen Automatische Reparatur von Knoten aktivieren für die verwaltete Knotengruppe. Weitere Informationen finden Sie unter the section called “Create”.
- Wenn Sie die AWS CLI verwenden, fügen Sie `--node-repair-config enabled=true` dem `eks update-nodegroup-config`Befehl `eks create nodegroup`or den hinzu.
- Ein Beispiel `eksctlClusterConfig`, das eine verwaltete Knotengruppe mit auto Knotenreparatur verwendet, finden Sie unter 44-node-repair.yaml on. GitHub

Amazon EKS bietet eine differenziertere Kontrolle über das Verhalten der automatischen Knotenreparatur durch Folgendes:

- `maxUnhealthyNodeThresholdCount` und `maxUnhealthyNodeThresholdPercentage`
  - In diesen Feldern können Sie eine Anzahl oder einen Prozentsatz als Schwellenwert für fehlerhafte Knoten festlegen, ab dem die automatische Knotenreparatur beendet wird. Dadurch erhalten Sie mehr Kontrolle über den „Explosionsradius“ der automatischen Knotenreparatur.
  - Sie können entweder die absolute Anzahl oder den Prozentsatz festlegen, jedoch nicht beides.
- `maxParallelNodesRepairedCount` und `maxParallelNodesRepairedPercentage`
  - In diesen Feldern können Sie die maximale Anzahl von Knoten angeben, die gleichzeitig oder parallel repariert werden können, ausgedrückt als Anzahl oder Prozentsatz aller fehlerhaften Knoten. Dadurch erhalten Sie eine differenzierte Kontrolle über das Tempo des Knotenaustauschs.
  - Ähnlich wie beim Schwellenwert für fehlerhafte Knoten können Sie entweder die absolute Anzahl oder den Prozentsatz festlegen, jedoch nicht beides.
- `nodeRepairConfigOverrides`
  - Es handelt sich hierbei um eine komplexe Struktur, die es Ihnen ermöglicht, differenzierte Überschreibungen für bestimmte Reparaturmaßnahmen festzulegen. Diese Überschreibungen steuern die Reparaturmaßnahme und die Reparaturverzögerungszeit, bevor ein Knoten als reparaturbedürftig eingestuft wird.
  - Die spezifischen Felder in dieser Struktur sind:
    - `nodeMonitoringCondition`: Der vom Knoten-Überwachungsagent gemeldete fehlerhafte Zustand.
    - `nodeUnhealthyReason`: Der Grund, warum der Knoten-Überwachungsagent den Knoten als fehlerhaft identifiziert hat.
    - `minRepairWaitTimeMins`: Die Mindestzeit (in Minuten), die der Reparaturzustand und der Grund für den fehlerhaften Zustand bestehen müssen, bevor der Knoten für eine Reparatur in Frage kommt.
    - `repairAction`: Die Maßnahme, die das Reparatursystem ergreifen soll, wenn die oben genannten Bedingungen erfüllt sind.
  - Wenn Sie dieses Feld verwenden, müssen Sie alle Felder in der Struktur angeben. Sie können auch eine Liste dieser Überschreibungen bereitstellen.

- `nodeMonitoringCondition` und `nodeUnhealthyReason` sind manuelle Texteingaben, die Sie festlegen, um anzugeben, dass Sie vom Standardverhalten des Systems abweichen möchten.
- Mit den Feldern `minRepairWaitTimeMins` und `repairAction` können Sie Abweichungen vom Standardverhalten des Systems angeben.

Probleme mit dem Zustand des Knotens

In den folgenden Tabellen werden Knotenintegritätsprobleme beschrieben, die vom Knoten-Überwachungsagenten erkannt werden können. Es gibt zwei Arten von Problemen:

- Zustand – Ein schwerwiegendes Problem, das eine Behebungsmaßnahme wie den Austausch einer Instance oder einen Neustart erfordert. Wenn die automatische Reparatur aktiviert ist, führt Amazon EKS eine Reparaturmaßnahme durch, entweder einen Knotenaustausch oder einen Neustart. Weitere Informationen finden Sie unter the section called “Knotenzustände”.
- Ereignis – Ein vorübergehendes Problem oder eine suboptimale Knotenkonfiguration. Es erfolgt keine automatische Reparatur. Weitere Informationen finden Sie unter the section called “Knotenereignisse”.

AcceleratedHardware Probleme mit dem Zustand des Knotens

Der Überwachungszustand ist `AcceleratedHardwareReady` für Probleme in der folgenden Tabelle mit dem Schweregrad „Zustand“.

Wenn die automatische Reparatur aktiviert ist, beginnen die aufgeführten Reparaturaktionen 10 Minuten, nachdem das Problem erkannt wurde. Weitere Informationen zu XID-Fehlern finden Sie unter XID-Fehler in der Dokumentation zur Bereitstellung und Verwaltung von NVIDIA-GPUs. Weitere Informationen zu den einzelnen XID-Meldungen finden Sie unter XID-Meldungen verstehen in der Dokumentation zur Bereitstellung und Verwaltung von NVIDIA-GPUs.

Name	Schweregrad	Description
DCGMDiagnosticFehlschlag	Bedingung	Ein Testfall aus der Test-Suite für die aktive DCGM-Diagnose ist fehlgeschlagen.

Name	Schweregrad	Description
DCGMError	Bedingung	Die Verbindung zum DCGM-Hostprozess wurde unterbrochen oder konnte nicht hergestellt werden.
DCGMFieldFehler [Code]	Veranstaltung	DCGM hat anhand einer Feld-ID eine Verschlechterung der GPU festgestellt.
DCGMHealthCode [Kode]	Veranstaltung	Ein DCGM-Gesundheitscheck schlug ohne schwerwiegenden Fehler fehl.
DCGMHealthKode [Kodierung]	Bedingung	Ein DCGM-Gesundheitscheck ist auf fatale Weise fehlgeschlagen.
Neuron DMAError	Bedingung	Bei einer DMA-Engine ist ein nicht behebbarer Fehler aufgetreten.
Neuronenfehler HBMUncorrectable	Bedingung	Bei einem HBM ist ein nicht korrigierbarer Fehler aufgetreten und es wurden falsche Ergebnisse ausgegeben.
Neuronenfehler NCUncorrectable	Bedingung	Ein nicht korrigierbarer Neuron Core-Speicherfehler wurde erkannt.
Neuronenfehler SRAMUncorrectable	Bedingung	Ein On-Chip-SRAM hat einen Paritätsfehler festgestellt und unkorrekte Ergebnisse erzeugt.

Name	Schweregrad	Description
NvidiaDeviceCountMismatch	Veranstaltung	Die Anzahl der durch NVML GPUs sichtbaren Geräte stimmt nicht mit der Anzahl der NVIDIA-Geräte im Dateisystem überein.
NvidiaDoubleBitError	Bedingung	Der GPU-Treiber hat einen doppelten Bit-Fehler erzeugt.
Nvidia NCCLError	Veranstaltung	In der NVIDIA-Bibliothek für kollektive Kommunikation (`libnccl`) ist ein Sicherhei tsfehler aufgetreten.
Nvidia-Fehler NVLink	Bedingung	NVLink Fehler wurden vom GPU-Treiber gemeldet.
PCIeNvidia-Fehler	Veranstaltung	PCIe Wiederholungen wurden ausgelöst, um Übertragu ngsfehler zu beheben.
NvidiaPageRetirement	Veranstaltung	Der GPU-Treiber hat eine Speicherseite zur Außerbetr iebnahme markiert. Dies kann auftreten, wenn ein einzelner Doppel-Bit-Fehler oder zwei Einzel-Bit-Fehler an derselben Adresse auftreten.
NvidiaPowerError	Veranstaltung	Der Stromverbrauch von hat die zulässigen GPUs Grenzwerte überschritten.
NvidiaThermalError	Veranstaltung	Thermischer Status hat die GPUs zulässigen Grenzwerte überschritten.

Name	Schweregrad	Description
NVIDIAxID [Code] -Fehler	Bedingung	Ein kritischer GPU-Fehler ist aufgetreten.
NvidiaXID[Code]Warning	Veranstaltung	Ein unkritischer GPU-Fehler ist aufgetreten.

ContainerRuntime Probleme mit dem Zustand des Knotens

Der Überwachungszustand ist `ContainerRuntimeReady` für Probleme in der folgenden Tabelle mit dem Schweregrad „Zustand“.

Name	Schweregrad	Description
ContainerRuntimeFailed	Veranstaltung	Die Container-Laufzeit konnte keinen Container erstellen . Dies hängt wahrscheinlich mit einem der gemeldeten Probleme zusammen, wenn es wiederholt auftritt.
DeprecatedContainerdConfiguration	Veranstaltung	Ein Container-Image, das die veraltete Image-Manifest-Version 2, Schema 1 verwendet, wurde kürzlich auf den Knoten übertragen. `containerd`
KubeletFailed	Veranstaltung	Der kubelet ist in einen fehlerhaften Zustand übergegangen.
LivenessProbeFailures	Veranstaltung	Es wurde ein Fehler bei der Aktivitätsprüfung erkannt, der bei wiederholtem Auftreten möglicherweise auf Probleme mit dem Anwendungscode

Name	Schweregrad	Description
		oder unzureichende Timeout-Werte hinweist.
PodStuckTerminating	Bedingung	Ein Pod ist oder war über einen längeren Zeitraum hinweg blockiert. Dies kann auf CRI-Fehler zurückzuführen sein, die den Fortschritt des Pod-Status verhindern.
ReadinessProbeFailures	Veranstaltung	Es wurde ein Fehler bei der Bereitschaftsprüfung erkannt, der bei wiederholtem Auftreten möglicherweise auf Probleme mit dem Anwendungscode oder unzureichende Timeout-Werte hinweist.
[Name] RepeatedRestart	Veranstaltung	Eine Systemd-Einheit wird häufig neu gestartet.
ServiceFailedToStart	Veranstaltung	Eine systemd-Einheit konnte nicht gestartet werden

Integritätsprobleme des Kernelknotens

Der Überwachungszustand ist `KernelReady` für Probleme in der folgenden Tabelle mit dem Schweregrad „Zustand“.

Name	Schweregrad	Description
AppBlocked	Veranstaltung	Die Aufgabe wurde für einen längeren Zeitraum aus der Planung ausgeschlossen, was in der Regel darauf zurückzuführen ist, dass sie

Name	Schweregrad	Description
		bei der Eingabe oder Ausgabe blockiert wurde.
AppCrash	Veranstaltung	Eine Anwendung auf dem Knoten ist abgestürzt.
ApproachingKernelPidMax	Veranstaltung	Die Anzahl der Prozesse nähert sich der maximalen Anzahl PIDs , die gemäß der aktuellen `kernel.pid_max` Einstellung verfügbar sind. Danach können keine Prozesse mehr gestartet werden.
ApproachingMaxOpenFiles	Veranstaltung	Die Anzahl der geöffneten Dateien nähert sich der maximalen Anzahl möglicher geöffneter Dateien gemäß den aktuellen Kernel-Einstellungen. Danach wird das Öffnen neuer Dateien fehlschlagen
ConntrackExceededKernel	Veranstaltung	Die Verbindungsverfolgung hat das Maximum für den Kernel überschritten, und es konnten keine neuen Verbindungen hergestellt werden, was zu Paketverlusten führen kann.

Name	Schweregrad	Description
ExcessiveZombieProcesses	Veranstaltung	Es sammeln sich zahlreiche Prozesse, die nicht vollständig wiederhergestellt werden können. Dies weist auf Anwendungsprobleme hin und kann dazu führen, dass die Grenzen der Systemprozesse erreicht werden.
ForkFailedOutOfPIDs	Bedingung	Ein Fork- oder Exec-Aufruf ist fehlgeschlagen, weil das System keinen Prozess IDs - oder Speicherzugriff mehr hatte. Dies kann auf Zombie-Prozesse oder physische Speichererschöpfung zurückzuführen sein.
KernelBug	Veranstaltung	Ein Kernel-Fehler wurde vom Linux-Kernel selbst erkannt und gemeldet. Dieser kann jedoch manchmal durch Knoten mit hoher CPU- oder Speicherauslastung verursacht werden, was zu einer verzögerten Ereignisverarbeitung führt.

Name	Schweregrad	Description
LargeEnvironment	Veranstaltung	Die Anzahl der Umgebungsvariablen für diesen Prozess ist größer als erwartet, was möglicherweise darauf zurückzuführen ist, dass viele Dienste auf true `enableServiceLinks` gesetzt sind, was zu Leistungsproblemen führen kann.
RapidCron	Veranstaltung	Ein Cron-Auftrag wird auf diesem Knoten schneller als alle fünf Minuten ausgeführt. Dies kann die Leistung beeinträchtigen, wenn der Auftrag erhebliche Ressourcen verbraucht.
SoftLockup	Veranstaltung	Die CPU ist für eine bestimmte Zeit blockiert.

Probleme mit der Integrität von Netzwerkknoten

Der Überwachungszustand ist `NetworkingReady` für Probleme in der folgenden Tabelle mit dem Schweregrad „Zustand“.

Name	Schweregrad	Description
BandwidthInExceeded	Veranstaltung	Pakete wurden in die Warteschlange gestellt oder verworfen, da die eingehende Gesamtbandbreite das Maximum für die Instance überschritten hat.

Name	Schweregrad	Description
BandwidthOutExceeded	Veranstaltung	Pakete wurden in die Warteschlange gestellt oder verworfen, da die ausgehende Gesamtbandbreite das Maximum für die Instance überschritten hat.
ConntrackExceeded	Veranstaltung	Die Verbindungsverfolgung hat das Maximum für die Instance überschritten und es konnten keine neuen Verbindungen hergestellt werden, was zu Paketverlusten führen kann.
IPAMDInconsistentStatus	Veranstaltung	Der Status des IPAMD-Checkpoints auf der Festplatte entspricht nicht der Laufzeit IPs in der Container-Laufzeit.
IPAMDNoIPs	Veranstaltung	IPAMD hat keine IP-Adressen mehr.
IPAMDNotBereit	Bedingung	IPAMD kann keine Verbindung zum API-Server herstellen.
IPAMDNotLäuft	Bedingung	Es wurde festgestellt, dass der Amazon VPC CNI-Prozess nicht läuft.
IPAMDRepeatedlyStarten Sie neu	Veranstaltung	Es sind mehrere Neustarts des IPAMD-Services aufgetreten.

Name	Schweregrad	Description
InterfaceNotRunning	Bedingung	Diese Schnittstelle scheint nicht zu funktionieren oder es liegen Netzwerkprobleme vor.
InterfaceNotUp	Bedingung	Diese Schnittstelle scheint derzeit nicht verfügbar zu sein oder es liegen Netzwerkp robleme vor.
KubeProxyNotReady	Veranstaltung	Kube-proxy konnte die Ressourcen nicht überwachen oder auflisten.
LinkLocalExceeded	Veranstaltung	Pakete wurden verworfen, da die PPS des Datenverkehrs zu lokalen Proxy- Services das Maximum der Netzwerks chnittstelle überschritten hat.
MACAddressPolicyMisconfigur ed	Veranstaltung	Die Verbindungskonfiguration systemd-networkd hat den falschen Wert. `MACAddres sPolicy`
MissingDefaultRoutes	Veranstaltung	Es fehlen Standard- Weiterlei tungsregeln.
Fehlt IPRoutes	Veranstaltung	Es fehlen Routen für Pod IPs.
Es fehlt IPRules	Veranstaltung	Es fehlen Regeln für Pod IPs.
MissingLoopbackInterface	Bedingung	Die Loopback-Schnittstelle fehlt in dieser Instance, was zu einem Ausfall von Services führt, die von der lokalen Konnektivität abhängig sind.

Name	Schweregrad	Description
NetworkSysctl	Veranstaltung	Die `sysctl` Netzwerke instellungen dieses Knotens sind möglicherweise falsch.
PPSExceeded	Veranstaltung	Pakete wurden in die Warteschlange gestellt oder verworfen, da die bidirekti onale PPS das Maximum für die Instance überschritten hat.
PortConflict	Veranstaltung	Wenn ein Pod HostPort verwendet, kann er `iptables` Regeln schreiben, die die bereits gebundenen Ports des Hosts außer Kraft setzen und so möglicherweise den API-Serverzugriff `kubelet` verhindern.
UnexpectedRejectRule	Veranstaltung	Im wurde eine unerwartete `REJECT` `DROP` Oder-Rege l gefunden`iptables`, die möglicherweise den erwartete n Datenverkehr blockiert.

Probleme mit der Integrität von Speicherknoten

Der Überwachungszustand ist `StorageReady` für Probleme in der folgenden Tabelle mit dem Schweregrad „Zustand“.

Name	Schweregrad	Description
EBSInstanceIOPSExceeded	Veranstaltung	Die maximale Anzahl an IOPS für die Instanz wurde überschri tten.

Name	Schweregrad	Description
EBSInstanceThroughputExceeded	Veranstaltung	Der maximale Durchsatz für die Instanz wurde überschritten.
EBSVolumeIOPSExceeded	Veranstaltung	Die maximale Anzahl an IOPS für ein bestimmtes EBS-Volume wurde überschritten.
EBSVolumeThroughputExceeded	Veranstaltung	Der maximale Durchsatz für ein bestimmtes Amazon EBS-Volume wurde überschritten.
EtcHostsMountFailed	Veranstaltung	Das Mounten des generierten Kubelets ist `/etc/hosts` fehlgeschlagen, da die Benutzerdaten während des Betriebs erneut bereitgestellt wurden. `/var/lib/kubelet/pods` `kubelet-container`
IODelays	Veranstaltung	In einem Prozess wurde eine Eingabe- oder Ausgabeverzögerung festgestellt, die bei übermäßiger Dauer auf eine unzureichende Eingabe-Ausgabe-Bereitstellung hindeuten könnte.

Name	Schweregrad	Description
KubeletDiskUsageSlow	Veranstaltung	Der meldet `kubelet` eine langsame Festplattennutzung beim Versuch, auf das Dateisystem zuzugreifen. Dies deutet möglicherweise auf eine unzureichende Eingabe- und Ausgabekapazität der Festplatte oder auf Probleme mit dem Dateisystem hin.
XFSSmallAverageClusterSize	Veranstaltung	Die durchschnittliche XFS-Clustergröße ist klein, was auf eine übermäßige Fragmenti erung des freien Speicherp latzes hindeutet. Dies kann die Erstellung von Dateien trotz verfügbarer Inodes oder freiem Speicherplatz verhinder n.

Integritätsstatus Ihrer Knoten anzeigen

In diesem Thema werden die Tools und Methoden beschrieben, die zur Überwachung des Zustands von Knoten in Amazon-EKS-Clustern verfügbar sind. Die Informationen umfassen Knotenbedingungen, Ereignisse und Erkennungsfälle, die Sie bei der Identifizierung und Diagnose von Problemen auf Knotenebene unterstützen. Verwenden Sie die hier beschriebenen Befehle und Muster, um die Ressourcen für den Knotenstatus zu überprüfen, Statusbedingungen zu interpretieren und Knotenereignisse für die Fehlerbehebung im Betrieb zu analysieren.

Mit Kubernetes-Befehlen können Sie einige Informationen zum Knotenzustand für alle Knoten abrufen. Wenn Sie den Knoten-Überwachungsagent über den Amazon EKS Auto Mode oder das verwaltete Amazon-EKS-Add-On verwenden, erhalten Sie eine größere Auswahl an Knotensignalen, die Ihnen bei der Fehlerbehebung helfen. Beschreibungen der vom Knoten-Überwachungsagent erkannten Integritätsprobleme werden auch im Dashboard zur Beobachtbarkeit angezeigt. Weitere Informationen finden Sie unter the section called “Knotenintegrität”.

Knotenzustände

Knotenzustände stellen terminale Probleme dar, die Abhilfemaßnahmen wie den Austausch oder Neustart einer Instance erfordern.

So rufen Sie die Zustände für alle Knoten ab:

```
kubectl get nodes -o 'custom-
columns=NAME:.metadata.name,CONDITIONS:.status.conditions[*].type,STATUS:.status.conditions[*].
```

So erhalten Sie detaillierte Bedingungen für einen bestimmten Knoten

```
kubectl describe node node-name
```

Beispiel für die Ausgabe des Zustands eines fehlerfreien Knotens:

```
  - lastHeartbeatTime: "2024-11-21T19:07:40Z"
    lastTransitionTime: "2024-11-08T03:57:40Z"
    message: Monitoring for the Networking system is active
    reason: NetworkingIsReady
    status: "True"
    type: NetworkingReady
```

Beispiel für den Zustand eines fehlerhaften Knotens mit einem Netzwerkproblem:

```
  - lastHeartbeatTime: "2024-11-21T19:12:29Z"
    lastTransitionTime: "2024-11-08T17:04:17Z"
    message: IPAM-D has failed to connect to API Server which could be an issue with
      IPTable rules or any other network configuration.
    reason: IPAMDNotReady
    status: "False"
    type: NetworkingReady
```

Knotenereignisse

Knotenereignisse weisen auf vorübergehende Probleme oder nicht optimale Konfigurationen hin.

So rufen Sie alle vom Knoten-Überwachungsagenten gemeldeten Ereignisse ab

Wenn der Knoten-Überwachungsagent verfügbar ist, können Sie den folgenden Befehl ausführen.

```
kubectl get events --field-selector=reportingComponent=eks-node-monitoring-agent
```

Beispielausgabe:

```
LAST SEEN   TYPE      REASON       OBJECT
 MESSAGE
4s          Warning   SoftLockup   node/ip-192-168-71-251.us-west-2.compute.internal
 CPU stuck for 23s
```

So erhalten Sie Ereignisse für alle Knoten

```
kubectl get events --field-selector involvedObject.kind=Node
```

So rufen Sie Ereignisse für einen bestimmten Knoten ab

```
kubectl get events --field-selector involvedObject.kind=Node,involvedObject.name=node-name
```

So beobachten Sie Ereignisse in Echtzeit

```
kubectl get events -w --field-selector involvedObject.kind=Node
```

Beispiel für eine Ereignisausgabe:

```
LAST SEEN   TYPE     REASON           OBJECT         MESSAGE
2m          Warning  MemoryPressure   Node/node-1    Node experiencing memory pressure
5m          Normal   NodeReady        Node/node-1    Node became ready
```

Gängige Befehle zur Fehlerbehebung

```
# Get comprehensive node status
kubectl get node node-name -o yaml

# Watch node status changes
kubectl get nodes -w

# Get node metrics
```

```
kubectl top node
```

Abrufen von Knotenprotokollen für einen verwalteten Knoten mithilfe von kubectl und S3

Erfahren Sie, wie Sie Knotenprotokolle für einen von Amazon EKS verwalteten Knoten abrufen können, auf dem der Knoten-Überwachungsagent installiert ist.

Voraussetzungen

Stellen Sie sicher, dass Sie über Folgendes verfügen:

- Ein vorhandener Amazon-EKS-Cluster mit dem Knoten-Überwachungsagent. Weitere Informationen finden Sie unter the section called “Knotenintegrität”.
- Das Befehlszeilentool `kubectl`, das für die Kommunikation mit Ihrem Cluster installiert und konfiguriert wurde.
- Die AWS CLI wurde installiert und hat sich mit ausreichenden Berechtigungen angemeldet, um S3-Buckets und -Objekte zu erstellen.
- Eine aktuelle Installation von Python 3
- Das AWS SDK für Python 3, Boto 3, ist installiert.

Schritt 1: S3-Bucket-Ziel erstellen (optional)

Falls Sie noch nicht über einen S3-Bucket zum Speichern der Protokolle verfügen, erstellen Sie einen. Verwenden Sie den folgenden AWS CLI-Befehl. Der Bucket verwendet standardmäßig die `private`-Zugriffskontroll-Liste. *`bucket-name`*Ersetzen Sie es durch den von Ihnen ausgewählten eindeutigen Bucket-Namen.

```
aws s3api create-bucket --bucket <bucket-name>
```

Schritt 2: Vorab signierte S3-URL für HTTP Put erstellen

Amazon EKS gibt die Knotenprotokolle zurück, indem es eine HTTP-PUT-Operation an eine von Ihnen angegebene URL durchführt. In diesem Tutorial erstellen wir eine vorab signierte S3-HTTP-PUT-URL.

Die Protokolle werden als gzip tarball mit der `.tar.gz`-Erweiterung zurückgegeben.

> **Note**
>
> Sie müssen die AWS API oder ein SDK verwenden, um die vorsignierte S3-Upload-URL für EKS zum Hochladen der Protokolldatei zu erstellen. Sie können mit der AWS CLI keine vorsignierte S3-Upload-URL erstellen.

1. Bestimmen Sie, wo im Bucket Sie die Protokolle speichern möchten. Sie können beispielsweise *2024-11-12/logs1.tar.gz* als Schlüssel verwenden.
2. Speichern Sie den folgenden Python-Code in der Datei *presign-upload.py*. Ersetzen Sie *<bucket-name>* und *<key>*. Der Schlüssel sollte mit `.tar.gz` enden.

```
import boto3; print(boto3.client('s3').generate_presigned_url(
   ClientMethod='put_object',
   Params={'Bucket': '[.replaceable]`<bucket-name>`', 'Key':
 '[.replaceable]`<key>`'},
   ExpiresIn=[.replaceable]`1000`
))
```

3. Skript ausführen mit

```
python presign-upload.py
```

4. Beachten Sie die URL-Ausgabe. Verwenden Sie diesen Wert im nächsten Schritt als `http-put-destination`.

Weitere Informationen finden Sie unter Generieren einer vorsignierten URL zum Hochladen einer Datei in der Dokumentation zum AWS Boto3-SDK für Python.

Schritt 3: Ressource erstellen NodeDiagnostic

Identifizieren Sie den Namen des Knotens, von dem Sie Protokolle erfassen möchten.

Erstellen Sie ein `NodeDiagnostic`-Manifest, das den Namen des Knotens als Namen der Ressource verwendet und eine HTTP-PUT-URL-Zieladresse angibt.

```
apiVersion: eks.amazonaws.com/v1alpha1
kind: NodeDiagnostic
metadata:
    name: <node-name>
```

```
spec:
    logCapture:
        destination: http-put-destination
```

Wenden Sie die Manifestdatei auf Ihren Cluster an.

```
kubectl apply -f nodediagnostic.yaml
```

Sie können den Status der Sammlung überprüfen, indem Sie die `NodeDiagnostic`-Ressource beschreiben:

- Ein Status von `Success` oder `SuccessWithErrors` zeigt an, dass die Aufgabe abgeschlossen und die Protokolle an das angegebene Ziel hochgeladen wurden (`SuccessWithErrors` zeigt an, dass möglicherweise einige Protokolle fehlen).
- Wenn der Status „Fehler“ lautet, stellen Sie sicher, dass die Upload-URL korrekt formatiert und nicht abgelaufen ist.

```
kubectl describe nodediagnostics.eks.amazonaws.com/<node-name>
```

Schritt 4: Protokolle aus S3 herunterladen

Warten Sie etwa eine Minute, bevor Sie versuchen, die Protokolle herunterzuladen. Verwenden Sie anschließend die S3-CLI, um die Protokolle herunterzuladen.

```
# Once NodeDiagnostic shows Success status, download the logs
aws s3 cp s3://<bucket-name>/key ./<path-to-node-logs>.tar.gz
```

Schritt 5: NodeDiagnostic Ressource bereinigen

- `NodeDiagnostic`-Ressourcen werden nicht automatisch gelöscht. Sie sollten diese selbst bereinigen, nachdem Sie Ihre Protokoll-Artefakte erhalten haben

```
# Delete the NodeDiagnostic resource
kubectl delete nodediagnostics.eks.amazonaws.com/<node-name>
```

Übersicht über Amazon EKS Hybrid Nodes

Mit Amazon EKS Hybrid Nodes können Sie Ihre On-Premises- und Edge-Infrastruktur als Knoten in Amazon-EKS-Clustern verwenden. AWS verwaltet die von AWS gehostete Kubernetes-Steuerebene des Amazon-EKS-Clusters, und Sie verwalten die Hybridknoten, die in Ihren On-Premises- oder Edge-Umgebungen ausgeführt werden. Dies vereinheitlicht die Kubernetes-Verwaltung in Ihren Umgebungen und lagert die Verwaltung der Kubernetes-Steuerebene für Ihre On-Premises- und Edge-Anwendungen an AWS aus.

Amazon EKS Hybrid Nodes funktioniert mit jeder On-Premises Hardware oder virtuellem Rechner und bietet die Effizienz, Skalierbarkeit und Verfügbarkeit von Amazon EKS überall dort, wo Ihre Anwendungen ausgeführt werden müssen. Mit Amazon EKS Hybrid Nodes können Sie eine Vielzahl von Amazon-EKS-Features verwenden, darunter Amazon-EKS-Add-On, Amazon EKS Pod Identity, Cluster Access Entries, Cluster Insights und erweiterten Support für Kubernetes-Versionen. Amazon EKS Hybrid Nodes lässt sich nahtlos in AWS-Services wie AWS Systems Manager, AWS IAM Roles Anywhere, Amazon Managed Service für Prometheus und Amazon CloudWatch integrieren und ermöglicht so eine zentralisierte Überwachung, Protokollierung und Identitätsverwaltung.

Bei Amazon EKS Hybrid Nodes fallen keine Vorabverpflichtungen oder Mindestgebühren an. Die Kosten werden stündlich für die vCPU-Ressourcen Ihrer Hybridknoten berechnet, wenn diese mit Ihren Amazon-EKS-Clustern verbunden sind. Weitere Preisinformationen finden Sie unter Amazon EKS – Preise.

Features

EKS-Hybridknoten bieten die folgenden allgemeinen Features:

- Verwaltete Kubernetes-Steuerebene: AWS verwaltet die von AWS gehostete Kubernetes-Steuerebene des EKS-Clusters, und Sie verwalten die Hybridknoten, die in Ihren On-Premises- oder Edge-Umgebungen ausgeführt werden. Dies vereinheitlicht die Kubernetes-Verwaltung in Ihren Umgebungen und lagert die Verwaltung der Kubernetes-Steuerebene für Ihre On-Premises- und Edge-Anwendungen an AWS aus. Durch die Verlagerung der Kubernetes-Steuerebene auf AWS können Sie On-Premises-Kapazitäten für Ihre Anwendungen einsparen und darauf vertrauen, dass die Kubernetes-Steuerebene mit Ihren Workloads skaliert
- Konsistente EKS-Erfahrung: Die meisten EKS-Features werden von EKS-Hybridknoten unterstützt, um eine konsistente EKS-Erfahrung in Ihren On-Premises- und Cloud-Umgebungen zu gewährleisten. Dazu gehören EKS-Add-Ons, EKS Pod Identity, Cluster-Zugriffseinträge,

Cluster-Erkenntnisse, erweiterter Support für Kubernetes-Versionen und vieles mehr. Weitere Informationen zu den von EKS-Hybridknoten unterstützten EKS-Add-Ons finden Sie unter the section called “Add-Ons konfigurieren”.

- Zentralisierte Beobachtbarkeit und Identitätsverwaltung: EKS-Hybridknoten lassen sich nativ in AWS-Services wie AWS Systems Manager, AWS IAM Roles Anywhere, Amazon Managed Service für Prometheus und Amazon CloudWatch integrieren und ermöglicht so eine zentralisierte Überwachung, Protokollierung und Identitätsverwaltung.
- Burst-to-Cloud oder Erweiterung der On-Premises-Kapazität: Ein einzelner EKS-Cluster kann verwendet werden, um Hybridknoten und Knoten in AWS-Regionen, AWS Local Zones oder AWS Outposts auszuführen, um Burst-to-Cloud oder die On-Premises-Kapazität Ihrer EKS-Cluster zu erweitern. Weitere Information finden Sie unter Überlegungen zu Clustern im kombinierten Modus.
- Flexible Infrastruktur: EKS-Hybridknoten verfolgen einen Bring-Your-Own-Infrastructure-Ansatz und sind unabhängig von der Infrastruktur, die Sie für Hybridknoten verwenden. Sie können Hybridknoten auf physischen oder virtuellen Rechnern sowie auf x86- und ARM-Architekturen ausführen, wodurch die Migration von On-Premises-Workloads, die in Hybridknoten ausgeführt werden, über verschiedene Infrastrukturtypen hinweg möglich wird.
- Flexible Vernetzung: Bei EKS-Hybridknoten wird die Kommunikation zwischen der EKS-Steuerebene und den Hybridknoten über die VPC und die Subnetze geleitet, die Sie bei der Cluster-Erstellung angeben. Dies baut auf dem vorhandenen Mechanismus in EKS für die Vernetzung zwischen Steuerebene und Knoten auf. Dies ist flexibel und entspricht Ihrer bevorzugten Methode zum Verbinden Ihrer On-Premises-Netzwerke mit einem VPC in AWS. Es stehen mehrere dokumentierte Optionen zur Verfügung, darunter AWS Site-to-Site-VPN, AWS Direct Connect oder Ihre eigene VPN-Lösung. Sie können die Methode auswählen, die am besten zu Ihrem Anwendungsfall passt.

Grenzwerte

- Pro Cluster werden bis zu 15 CIDRs für Fern-Knoten-Netzwerke und 15 CIDRs für Fern-Pod-Netzwerke unterstützt.

Überlegungen

- EKS-Hybridknoten können mit neuen oder vorhandenen EKS-Clustern verwendet werden.
- EKS-Hybridknoten ist in allen AWS-Regionen verfügbar, mit Ausnahme der AWS GovCloud (US)-Regionen und der AWS-China-Regionen.

- EKS-Hybridknoten erfordern, dass zwischen Ihrer On-Premises-Umgebung und AWS eine zuverlässige Verbindung hergestellt wird. EKS-Hybridknoten sind nicht für getrennte, unterbrochene, intermittierende oder eingeschränkte (DDIL) Umgebungen geeignet. Wenn Sie in einer DDIL-Umgebung ausführen, empfehlen wir Ihnen Amazon EKS Anywhere. Informationen zum Verhalten von Hybridknoten bei Netzwerkunterbrechungen finden Sie in den Bewährten Methoden für EKS-Hybridknoten.
- Die Ausführung von EKS-Hybridknoten in Cloud-Infrastrukturen, einschließlich AWS-Regionen, AWS Local Zones, AWS Outposts oder anderen Clouds, wird nicht unterstützt. Wenn Sie Hybridknoten in Amazon-EC2-Instances ausführen, wird Ihnen die Gebühr für Hybridknoten in Rechnung gestellt.
- Die Fakturierung für Hybridknoten beginnt, wenn die Knoten dem EKS-Cluster beitreten, und endet, wenn die Knoten aus dem Cluster entfernt werden. Stellen Sie sicher, dass Sie Ihre Hybridknoten aus Ihrem EKS-Cluster entfernen, wenn Sie diese nicht verwenden.

Weitere Ressourcen

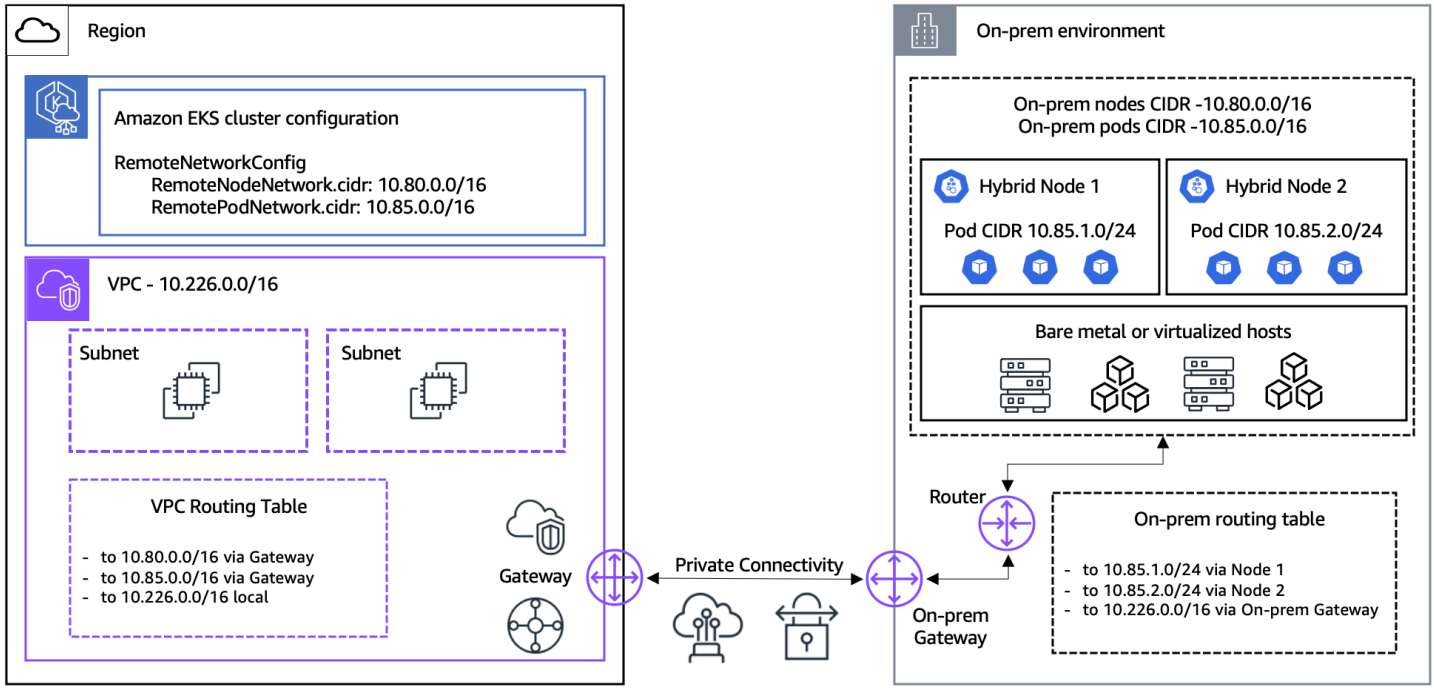
- EKS-Hybridknoten-Workshop: Schritt-für-Schritt-Anleitung zum Bereitstellen von EKS-Hybridknoten in einer Demo-Umgebung.
- AWS re:Invent: EKS-Hybridknoten: AWS-re:Invent-Sitzung zur Einführung der EKS-Hybridknoten, in der ein Kunde demonstriert, wie er EKS-Hybridknoten in seiner Umgebung verwendet.
- AWS re:Post: Cluster-Netzwerk für EKS-Hybridknoten: Dieser Artikel erläutert verschiedene Methoden, mit denen ein Netzwerk für EKS-Hybridknoten eingerichtet werden kann.
- AWS-Blog: Ausführung von GenAI-Inferenz über Umgebungen hinweg mit EKS-Hybridknoten: Dieser Blog-Beitrag zeigt, wie Sie GenAI-Inferenz über Umgebungen hinweg mit EKS-Hybridknoten ausführen können.

Voraussetzungen für die Einrichtung von Hybridknoten

Für die Verwendung von Amazon EKS Hybrid Nodes müssen Sie über eine private Verbindung von Ihrer On-Premises-Umgebung zu/von AWS, Bare-Metal-Servern oder virtuellen Rechnern mit einem unterstützten Betriebssystem sowie über konfigurierte AWS IAM Roles Anywhere- oder AWS Systems Manager (SSM)-Hybridaktivierungen verfügen. Sie sind für die Verwaltung dieser Voraussetzungen während des gesamten Lebenszyklus der Hybridknoten verantwortlich.

- Hybride Netzwerkkonnektivität von Ihrer On-Premises-Umgebung zu/von AWS

- Infrastruktur in Form von physischen oder virtuellen Rechnern
- Mit Hybridknoten kompatibles Betriebssystem
- On-Premises IAM-Anmeldeinformationsanbieter konfiguriert



Hybride Netzwerkkonnektivität

Die Kommunikation zwischen der Amazon-EKS-Steuerebene und den Hybridknoten wird über die VPC und Subnetze geleitet, die Sie während der Cluster-Erstellung übergeben. Dies baut auf dem vorhandenen Mechanismus in Amazon EKS für die Vernetzung von der Steuerebene zum Knoten auf. Es stehen Ihnen mehrere dokumentierte Optionen zur Verfügung, um Ihre On-Premises-Umgebung mit Ihrem VPC zu verbinden, darunter AWS-Site-to-Site-VPN, AWS Direct Connect oder Ihre eigene VPN-Verbindung. Weitere Informationen zur Verwendung dieser Lösungen für Ihre hybride Netzwerkverbindung finden Sie in den Benutzerhandbüchern zu AWS Site-to-Site VPN und AWS Direct Connect.

Für eine optimale Leistung empfehlen wir Ihnen eine zuverlässige Netzwerkverbindung mit einer Geschwindigkeit von mindestens 100 Mbit/s und einer maximalen Rundlauf-Latenz von 200 ms für die Verbindung der Hybridknoten mit der AWS-Region. Dies sind allgemeine Leitlinien, die für die meisten Anwendungsfälle geeignet sind, jedoch keine strengen Anforderungen darstellen. Die Bandbreiten- und Latenzanforderungen können je nach Anzahl der Hybridknoten und Ihren Workload-Eigenschaften variieren, z. B. Größe der Anwendungs-Images, Anwendungselastizität,

Überwachungs- und Protokollierungskonfigurationen und Anwendungsabhängigkeiten beim Zugriff auf in anderen AWS-Services gespeicherte Daten. Wir empfehlen Ihnen, vor der Bereitstellung in der Produktionsumgebung Tests mit Ihren eigenen Anwendungen und Umgebungen durchzuführen, um sicherzustellen, dass Ihre Netzwerkkonfiguration den Anforderungen Ihrer Workloads entspricht.

On-Premises-Netzwerkkonfiguration

Sie müssen den eingehenden Netzwerkzugriff von der Amazon-EKS-Steuerebene auf Ihre lokale Umgebung aktivieren, damit die Amazon-EKS-Steuerebene mit dem in Hybridknoten ausgeführten `kubelet` und optional mit in Ihren Hybridknoten ausgeführten Webhooks kommunizieren kann. Darüber hinaus müssen Sie den ausgehenden Netzwerkzugriff für Ihre Hybridknoten und die darauf ausgeführten Komponenten aktivieren, damit diese mit der Amazon-EKS-Steuerebene kommunizieren können. Sie können diese Kommunikation so konfigurieren, dass sie für Ihre AWS Direct Connect-, AWS Site-to-Site-VPN- oder Ihre eigene VPN-Verbindung vollständig privat bleibt.

Die Classless Inter-Domain Routing (CIDR)-Bereiche, die Sie für Ihre On-Premises-Knoten- und Pod-Netzwerke verwenden, müssen IPv4 RFC-1918-Adressbereiche verwenden. Ihr On-Premises Router muss mit Routen zu Ihren On-Premises Knoten und optional zu Pods konfiguriert sein. Weitere Informationen zu den Anforderungen an das On-Premises-Netzwerk, einschließlich einer vollständigen Liste der erforderlichen Ports und Protokolle, die in Ihrer Firewall und Ihrer On-Premises-Umgebung aktiviert werden müssen, finden Sie unter the section called “Konfiguration des On-Premises-Netzwerks”.

EKS-Cluster-Konfiguration

Um die Latenz zu minimieren, empfehlen wir Ihnen, Ihren Amazon-EKS-Cluster in der AWS-Region zu erstellen, die Ihrer On-Premises- oder Edge-Umgebung am nächsten liegt. Sie übergeben Ihre On-Premises-Knoten- und Pod-CIDRs während der Erstellung des Amazon-EKS-Clusters über zwei API-Felder: `RemoteNodeNetwork` und `RemotePodNetwork`. Möglicherweise müssen Sie die CIDRs Ihrer On-Premises-Knoten und Pods mit Ihrem On-Premises-Netzwerk-Team besprechen. Der Knoten-CIDR wird von Ihrem On-Premises-Netzwerk zugewiesen und der Pod-CIDR wird von der Container Network Interface (CNI) zugewiesen, die Sie verwenden, wenn Sie ein Überlagerungsnetzwerk für Ihr CNI verwenden. Cilium und Calico verwenden standardmäßig Überlagerungsnetzwerke.

Die On-Premises-Knoten- und Pod-CIDRs, die Sie über die Felder `RemoteNodeNetwork` und `RemotePodNetwork` konfigurieren, werden verwendet, um die Amazon-EKS-Steuerebene so zu konfigurieren, dass der Datenverkehr durch Ihre VPC zu `kubelet` und den auf Ihren Hybridknoten ausgeführten Pods geleitet wird. Ihre On-Premises-Knoten- und Pod-CIDRs dürfen sich nicht

gegenseitig überschneiden, ebenso wenig wie die VPC-CIDR, die Sie bei der Cluster-Erstellung übergeben, oder die IPv4-Konfiguration für Ihren Amazon-EKS-Cluster. Darüber hinaus müssen Pod-CIDRs für jeden EKS-Cluster eindeutig sein, damit Ihr On-Premises-Router den Datenverkehr weiterleiten kann.

Wir empfehlen, entweder den öffentlichen oder den privaten Endpunktzugriff für den API-Server-Endpunkt von Amazon EKS Kubernetes zu verwenden. Wenn Sie „Öffentlich und Privat“ auswählen, wird der API-Serverendpunkt für Amazon EKS Kubernetes immer auf die öffentlichen IP-Adressen für Hybridknoten aufgelöst, die außerhalb Ihrer VPC ausgeführt werden. Dies kann dazu führen, dass Ihre Hybridknoten nicht dem Cluster beitreten können. Wenn Sie den öffentlichen Endpunktzugriff verwenden, wird der Endpunkt des Kubernetes-API-Servers in öffentliche IPs aufgelöst und die Kommunikation von Hybridknoten zur Amazon-EKS-Steuerebene wird über das Internet geleitet. Wenn Sie den privaten Endpunktzugriff wählen, wird der Endpunkt des Kubernetes-API-Servers in private IPs aufgelöst und die Kommunikation von Hybridknoten zur Amazon-EKS-Steuerebene wird über Ihre private Konnektivitätsverbindung geleitet, in den meisten Fällen AWS Direct Connect oder AWS-Site-to-Site-VPN.

VPC-Konfiguration

Sie müssen die VPC, die Sie bei der Erstellung des Amazon-EKS-Clusters übergeben, mit Routen in der Routing-Tabelle für Ihren On-Premises-Knoten und optional für Pod-Netzwerke mit Ihrem Virtual Private Gateway (VGW) oder Transit Gateway (TGW) als Ziel konfigurieren. Es folgt ein Beispiel. Ersetzen Sie `REMOTE_NODE_CIDR` und `REMOTE_POD_CIDR` durch die Werte für Ihr On-Premises-Netzwerk.

Bestimmungsort	Ziel	Beschreibung
10.226.0.0/16	local	Lokaler Datenverkehr zu den VPC-Routen innerhalb der VPC
REMOTE_NODE_CIDR	tgw-abcdef123456	CIDR des On-Premises-Knotens, Weiterleitung des Datenverkehrs an den TGW
REMODE_POD_CIDR	tgw-abcdef123456	CIDR des On-Premises-Pods, Weiterleitung des Datenverk ehrs an den TGW

Sicherheitsgruppenkonfiguration

Wenn Sie einen Cluster erstellen, erstellt Amazon EKS eine Sicherheitsgruppe mit dem Namen `eks-cluster-sg-<cluster-name>-<uniqueID>`. Sie können die eingehenden Regeln dieser Cluster-Sicherheitsgruppe nicht ändern. Sie können jedoch die ausgehenden Regeln einschränken. Sie müssen Ihrem Cluster eine zusätzliche Sicherheitsgruppe hinzufügen, um das Kubelet und optional die auf Ihren Hybridknoten ausgeführten Webhooks zu aktivieren, damit sie Kontakt mit der Amazon-EKS-Steuerebene aufnehmen können. Die erforderlichen Eingangsregeln für diese zusätzliche Sicherheitsgruppe sind unten aufgeführt. Ersetzen Sie `REMOTE_NODE_CIDR` und `REMOTE_POD_CIDR` durch die Werte für Ihr On-Premises-Netzwerk.

Name	ID der Sicherheitsgruppenregel	IP-Version	Typ	Protocol (Protokoll)	Port-Bereich	Quelle
On-Premise-Knoten eingehend	sgr-abcdef123456	IPv4	HTTPS	TCP	443	REMOTE_NODE_CIDR
On-Premise-Pod eingehend	sgr-abcdef654321	IPv4	HTTPS	TCP	443	REMOTE_POD_CIDR

Infrastruktur

Sie müssen über Bare-Metal-Server oder virtuelle Rechnern verfügen, die Sie als Hybridknoten verwenden können. Hybridknoten sind unabhängig von der zugrunde liegenden Infrastruktur und unterstützen x86- und ARM-Architekturen. Amazon EKS Hybrid Nodes verfolgt einen „Bring Your Own Infrastructure“-Ansatz, bei dem Sie für die Bereitstellung und Verwaltung der Bare-Metal-Server oder virtuellen Rechnern verantwortlich sind, die Sie für Hybridknoten verwenden. Es gibt zwar keine strengen Mindestanforderungen an die Ressource, jedoch empfehlen wir, für Hybridknoten Hosts mit mindestens 1 vCPU und 1 GiB RAM zu verwenden.

Betriebssystem

Bottlerocket, Amazon Linux 2023 (AL2023), Ubuntu und RHEL werden fortlaufend für die Verwendung als Knotenbetriebssystem für Hybridknoten validiert. Bottlerocket wird ausschließlich

von AWS in VMware-vSphere-Umgebungen unterstützt. AL2023 ist nicht durch AWS-Support-Pläne abgedeckt, wenn es außerhalb von Amazon EC2 ausgeführt wird AL2023 kann nur in virtualisierten Umgebungen On-Premises verwendet werden. Weitere Informationen finden Sie im Benutzerhandbuch zu Amazon Linux 2023. AWS unterstützt die Integration von Hybridknoten mit den Betriebssystemen Ubuntu und RHEL, bietet jedoch keine Unterstützung für das Betriebssystem selbst.

Sie sind für die Bereitstellung und Verwaltung des Betriebssystems verantwortlich. Wenn Sie Hybridknoten zum ersten Mal testen, ist es am einfachsten, die CLI für Amazon EKS Hybrid Nodes (`nodeadm`) auf einem bereits bereitgestellten Host auszuführen. Für Produktionsbereitstellungen empfehlen wir, `nodeadm` in Ihre Golden-Betriebssystem-Images aufzunehmen und es so zu konfigurieren, dass es als systemd-Service ausgeführt wird, um Hosts beim Host-Startup automatisch mit Amazon-EKS-Clustern zu verbinden.

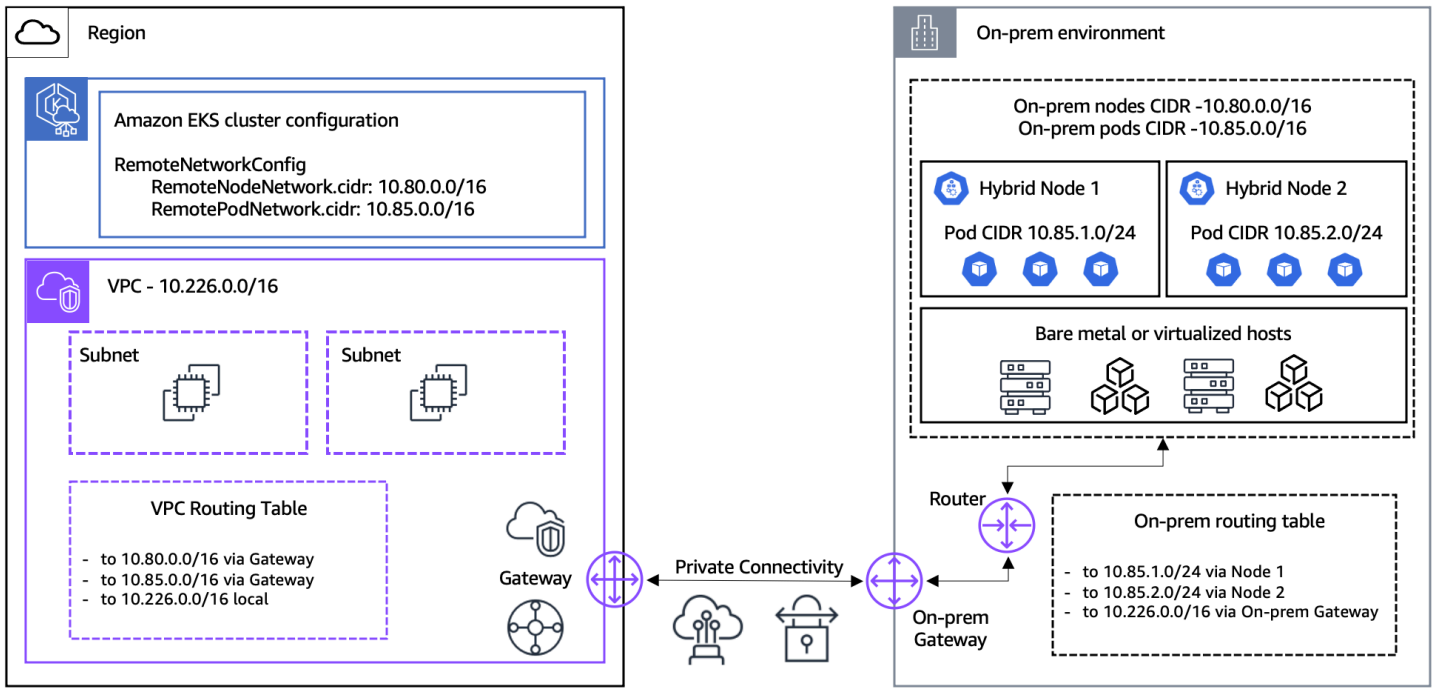
On-Premises IAM-Anmeldeinformationsanbieter

Amazon EKS Hybrid Nodes verwenden temporäre IAM-Anmeldeinformationen, die durch AWS-SSM-Hybridaktivierungen oder AWS IAM Roles Anywhere bereitgestellt werden, um sich beim Amazon-EKS-Cluster zu authentifizieren. Sie müssen entweder AWS-SSM-Hybridaktivierungen oder AWS IAM Roles Anywhere mit der Amazon EKS Hybrid Nodes CLI (`nodeadm`) verwenden Wir empfehlen Ihnen, AWS-SSM-Hybridaktivierungen zu verwenden, wenn Sie nicht über eine vorhandene Public-Key-Infrastruktur (PKI) mit einer Zertifizierungsstelle (CA) und Zertifikaten für Ihre On-Premises-Umgebungen verfügen. Wenn Sie über eine vorhandene PKI und Zertifikate On-Premises verfügen, verwenden Sie AWS IAM Roles Anywhere.

Ähnlich wie bei the section called “Knoten-IAM-Rolle” für Knoten, die auf Amazon EC2 ausgeführt werden, erstellen Sie eine IAM-Rolle für Hybridknoten mit den erforderlichen Berechtigungen, um Hybridknoten mit Amazon-EKS-Clustern zu verbinden. Wenn Sie AWS IAM Roles Anywhere verwenden, konfigurieren Sie eine Vertrauensrichtlinie, die es AWS IAM Roles Anywhere ermöglicht, die IAM-Rolle für Hybridknoten zu übernehmen, und konfigurieren Sie Ihr AWS IAM Roles Anywhere-Profil mit der IAM-Rolle für Hybridknoten als übernehmbare Rolle. Wenn Sie AWS SSM verwenden, konfigurieren Sie eine Vertrauensrichtlinie, die es AWS SSM ermöglicht, die IAM-Rolle für Hybridknoten zu übernehmen und die Hybridaktivierung mit der IAM-Rolle für Hybridknoten zu erstellen. Informationen zum Erstellen der IAM-Rolle für Hybridknoten mit den erforderlichen Berechtigungen finden Sie unter the section called “Anmeldeinformationen vorbereiten”.

Vorbereitung der Vernetzung für Hybridknoten

Dieses Thema bietet einen Überblick über die Netzwerkkonfiguration, die Sie vor der Erstellung Ihres Amazon-EKS-Clusters und dem Hinzufügen von Hybridknoten konfigurieren müssen. In diesem Leitfaden wird davon ausgegangen, dass Sie die Voraussetzungen für Hybrid-Netzwerkkonnektivität mit AWS Site-to-Site VPN, AWS Direct Connect oder Ihrer eigenen VPN-Lösung erfüllt haben.



Konfiguration des On-Premises-Netzwerks

Mindestanforderungen an das Netzwerk

Für ein optimales Erlebnis empfehlen wir eine zuverlässige Netzwerkkonnektivität von mindestens 100 Mbit/s und eine maximale Round-Trip-Latenz von 200 ms für die Verbindung der Hybridknoten mit der AWS Region. Dies sind allgemeine Leitlinien, die für die meisten Anwendungsfälle geeignet sind, jedoch keine strengen Anforderungen darstellen. Die Anforderungen an Bandbreite und Latenz können je nach Anzahl der Hybridknoten und Ihren Workload-Merkmalen wie Größe des Anwendungsimages, Anwendungselastizität, Überwachungs- und Protokollierungskonfigurationen und Anwendungsabhängigkeiten beim Zugriff auf Daten, die in anderen AWS Diensten gespeichert sind, variieren. Wir empfehlen Ihnen, vor der Bereitstellung in der Produktionsumgebung Tests mit Ihren eigenen Anwendungen und Umgebungen durchzuführen, um sicherzustellen, dass Ihre Netzwerkkonfiguration den Anforderungen Ihrer Workloads entspricht.

Lokaler Knoten und Pod CIDRs

Identifizieren Sie den Knoten und den Pod, den CIDRs Sie für Ihre Hybridknoten und die darauf ausgeführten Workloads verwenden werden. Die Knoten-CIDR wird aus Ihrem On-Premises-Netzwerk zugewiesen, und die Pod-CIDR wird aus Ihrer Container Network Interface (CNI) zugewiesen, wenn Sie ein Überlagerungsnetzwerk für Ihre CNI verwenden. Sie übergeben Ihren lokalen Knoten CIDRs und Pod CIDRs als Eingaben, wenn Sie Ihren EKS-Cluster mit den Feldern `RemoteNodeNetwork` und `RemotePodNetwork` erstellen. Ihr lokaler Knoten CIDRs muss in Ihrem lokalen Netzwerk routingfähig sein. Informationen zur CIDR-Routing-Fähigkeit des On-Premises-Pods finden Sie im folgenden Abschnitt.

Die CIDR-Blöcke für den On-Premises-Knoten und den Pod müssen die folgenden Anforderungen erfüllen:

1. Muss innerhalb eines der folgenden `IPv4`-RFC-1918-Bereiche liegen: `10.0.0.0/8`, `172.16.0.0/12` oder `192.168.0.0/16`.
2. Beachten Sie, dass sich die VPC-CIDR für Ihren EKS-Cluster und die `IPv4` CIDR Ihres Kubernetes-Service nicht überschneiden dürfen.

On-premises-Pod-Netzwerk-Routing

Wenn Sie EKS-Hybridknoten verwenden, empfehlen wir generell, Ihren lokalen Pod in Ihrem lokalen Netzwerk CIDRs routingfähig zu machen, um die vollständige Cluster-Kommunikation und Funktionalität zwischen Cloud- und lokalen Umgebungen zu ermöglichen.

Routingfähige Pod-Netzwerke

Wenn Sie Ihr Pod-Netzwerk in Ihrem On-Premises-Netzwerk routingfähig machen können, befolgen Sie die nachstehenden Anweisungen.

1. Konfigurieren Sie das `RemotePodNetwork`-Feld für Ihren EKS-Cluster mit Ihrer On-Premises-Pod-CIDR, Ihre VPC-Routing-Tabellen mit Ihrer On-Premises-Pod-CIDR und Ihre EKS-Cluster-Sicherheitsgruppe mit Ihrer On-Premises-Pod-CIDR.
2. Es gibt mehrere Techniken, mit denen Sie Ihren On-Premises-Pod CIDR in Ihrem On-Premises-Netzwerk routingfähig machen können, darunter Border Gateway Protocol (BGP), statische Routen oder andere benutzerdefinierte Routing-Lösungen. BGP ist die empfohlene Lösung, da sie skalierbarer und einfacher zu verwalten ist als alternative Lösungen, die eine benutzerdefinierte oder manuelle Routenkonfiguration erfordern. AWS unterstützt die BGP-Funktionen von Cilium

und Calico für Werbe-Pods CIDRs. Weitere Informationen finden Sie unter the section called “CNI konfigurieren” undthe section called “Routingfähige Fern-Pod-CIDRs”.

3. Webhooks können auf Hybridknoten ausgeführt werden, da die EKS-Steuerebene mit den Webhooks zugewiesenen Pod-IP-Adressen kommunizieren kann.
4. In Cloud-Knoten ausgeführte Workloads können direkt mit Workloads kommunizieren, die auf Hybridknoten im selben EKS-Cluster ausgeführt werden.
5. Andere AWS Dienste, wie AWS Application Load Balancers und Amazon Managed Service for Prometheus, können mit Workloads kommunizieren, die auf Hybridknoten ausgeführt werden, um den Netzwerkverkehr auszugleichen und Pod-Metriken auszuwerten.

Nicht routingfähige Pod-Netzwerke

Wenn Sie Ihre Pod-Netzwerke in Ihrem On-Premises-Netzwerk nicht routingfähig machen können, befolgen Sie die nachstehenden Anweisungen.

1. Webhooks können nicht in Hybridknoten ausgeführt werden, da Webhooks eine Verbindung von der EKS-Steuerebene zu den den Webhooks zugewiesenen Pod-IP-Adressen erfordern. In diesem Fall empfehlen wir, Webhooks auf Cloud-Knoten im selben EKS-Cluster wie Ihre Hybridknoten auszuführen. Weitere Informationen finden Sie unter the section called “Webhooks konfigurieren”.
2. Workloads auf Cloud-Knoten können nicht direkt mit Workloads auf Hybridknoten kommunizieren, wenn VPC CNI für Cloud-Knoten und Cilium oder Calico für Hybridknoten verwendet werden.
3. Verwenden Sie Verteilung des Service-Datenverkehrs, um den Datenverkehr auf die Zone zu beschränken, aus der er stammt. Weitere Informationen zur Verteilung des Service-Datenverkehrs finden Sie unter the section called “Verteilung des Service-Datenverkehrs konfigurieren”.
4. Konfigurieren Sie Ihr CNI so, dass es ausgehende Maskierung oder Network Address Translation (NAT) für den Pod-Datenverkehr verwendet, wenn dieser Ihre On-Premises-Hosts verlässt. Dies ist in Cilium standardmäßig aktiviert. Calico erfordert, dass `natOutgoing` auf `true` festgelegt ist.
5. Andere AWS Dienste, wie AWS Application Load Balancers und Amazon Managed Service for Prometheus, können nicht mit Workloads kommunizieren, die auf Hybridknoten ausgeführt werden.

Zugriff während der Installation und Aktualisierung von Hybridknoten erforderlich

Während der Installation, bei der Sie die Abhängigkeiten der Hybridknoten auf Ihren Hosts installieren, müssen Sie Zugriff auf die folgenden Domains haben. Dieser Vorgang kann einmalig beim Erstellen Ihrer Betriebssystem-Images oder zur Laufzeit auf jedem Host durchgeführt werden.

Dies gilt sowohl für die Erstinstallation als auch für das Upgrade der Kubernetes-Version Ihrer Hybridknoten.

Einige Pakete werden mit dem Standard-Paket-Manager des Betriebssystems installiert. Für AL2 023 und RHEL wird der `yum` Befehl verwendet, um, und zu installieren. `containerd` `ca-certificates` `iptables` `amazon-ssm-agent` Für Ubuntu, `apt` wird für die Installation von `containerd`, `ca-certificates` und `iptables` und `snap` wird für die Installation von `amazon-ssm-agent` verwendet.

Komponente	URL	Protocol (Protokoll)	Port
EKS-Knoten-Artefakte (S3)	https://hybrid-assets.eks.amazonaws.com	HTTPS	443
EKS-Service-Endpunkte	https://eks.*region*.amazonaws.com	HTTPS	443
ECR-Service-Endpunkte	https://api.ecr.*region*.amazonaws.com	HTTPS	443
EKS-ECR-Endpunkte	Informationen zu regionalen Endpunkten finden Sie unter the section called “Amazon-Image-Registries anzeigen”.	HTTPS	443
SSM-Binärendpunkt [1]	https://amazon-ssm - .s3. *region* *region*.amazonaws.com	HTTPS	443
SSM-Service-Endpunkt [1]	https://ssm.*region*.amazonaws.com	HTTPS	443

Komponente	URL	Protocol (Protokoll)	Port
IAM-Anywhere-Binärendpunkt [2]	https://rolesanywhere.amazonaws.com	HTTPS	443
IAM-Anywhere-Service-Endpunkt [2]	https://rolesanywhere.*region*.amazonaws.com	HTTPS	443
Endpunkte des Paket-Managers des Betriebssystems	Die Endpunkte des Paket-Repositorys sind betriebssystemspezifisch und können je nach geografischer Region variieren.	HTTPS	443

Note

[1] Der Zugriff auf die AWS SSM-Endpunkte ist nur erforderlich, wenn Sie SSM-Hybrid-Aktivierungen für Ihren lokalen AWS IAM-Anmeldeinformationsanbieter verwenden.
[2] Der Zugriff auf die AWS IAM-Endpunkte ist nur erforderlich, wenn Sie IAM Roles Anywhere für Ihren lokalen AWS IAM-Anmeldeinformationsanbieter verwenden.

Zugriff für den fortlaufenden Cluster-Betrieb erforderlich

Für den kontinuierlichen Cluster-Betrieb ist der folgende Netzwerkzugang für Ihre On-Premises-Firewall erforderlich.

Important

Je nach Ihrer Wahl des CNI müssen Sie zusätzliche Netzwerk-Zugriffsregeln für die CNI-Ports konfigurieren. Weitere Informationen finden Sie in der Cilium-Dokumentation und der Calico-Dokumentation.

Typ	Protocol (Protokoll)	Richtung	Port	Quelle	Ziel	Usage
HTTPS	TCP	Ausgehend	443	Fern-Knoten-CIDR(s)	EKS-Cluster 1 IPs	kubelet zum Kubernetes-API-Server
HTTPS	TCP	Ausgehend	443	Fern-Pod-CIDR(s)	EKS-Cluster IPs [1]	Pod zum Kubernetes-API-Server
HTTPS	TCP	Ausgehend	443	Fern-Knoten-CIDR(s)	SSM-Service-Endpunkt	Aktualisierung der Anmeldeinformationen für SSM-Hybridaktivierungen und SSM-Heartbeats alle 5 Minuten
HTTPS	TCP	Ausgehend	443	Fern-Knoten-CIDR(s)	IAM-Anywhere-Service-Endpunkt	Aktualisierung der Anmeldeinformationen für IAM Roles Anywhere
HTTPS	TCP	Ausgehend	443	Fern-Pod-CIDR(s)	Regionales STS-Endpunkt	Pod zum STS-Endpu

Typ	Protocol (Protokoll)	Richtung	Port	Quelle	Ziel	Usage
						nkt, nur für IRSA erforderlich
HTTPS	TCP	Ausgehend	443	Fern-Knoten-CIDR(s)	Service-Endpunk für Amazon EKS Auth	Knoten zum Amazon-EKS-Auth-Endpunkt, nur für Amazon EKS Pod Identity erforderlich
HTTPS	TCP	Eingehend	10250	EKS-Cluster IPs [1]	Fern-Knoten-CIDR(s)	Kubernetes-API-Server zu Kubelet
HTTPS	TCP	Eingehend	Webhook-Ports	EKS-Cluster IPs [1]	Fern-Pod-CIDR(s)	Kubernetes-API-Server zu Webhooks

Typ	Protocol (Protokoll)	Richtung	Port	Quelle	Ziel	Usage
HTTPS	TCP, UDP	Eingehend, ausgehend	53	Fern-Pod-CIDR(s)	Fern-Pod-CIDR(s)	Pod zu CoreDNS. Wenn Sie mindestens eine Replik von CoreDNS in der Cloud ausführen, müssen Sie DNS-Datenverkehr zum VPC zulassen, auf dem CoreDNS ausgeführt wird.
Benutzerdefiniert	Benutzerdefiniert	Eingehend, ausgehend	App-Ports	Fern-Pod-CIDR(s)	Fern-Pod-CIDR(s)	Pod zu Pod

> **Note**
>
> [1] Der IPs des EKS-Clusters. Weitere Informationen finden Sie im folgenden Abschnitt zu elastischen Netzwerkschnittstellen von Amazon EKS.

Amazon-EKS-Netzwerkschnittstellen

Amazon EKS fügt Netzwerkschnittstellen an die Subnetze in der VPC an, die Sie bei der Cluster-Erstellung übergeben, um die Kommunikation zwischen der EKS-Steuerebene und Ihrer

VPC zu ermöglichen. Die Netzwerkschnittstellen, die Amazon EKS erstellt, finden Sie nach der Clustererstellung in der EC2 Amazon-Konsole oder mit der AWS CLI. Die ursprünglichen Netzwerkschnittstellen werden gelöscht und neue Netzwerkschnittstellen erstellt, wenn Änderungen an Ihrem EKS-Cluster vorgenommen werden, z. B. Upgrades der Kubernetes-Version. Sie können den IP-Bereich für die Amazon EKS-Netzwerkschnittstellen einschränken, indem Sie eingeschränkte Subnetzgrößen für die Subnetze verwenden, die Sie bei der Clustererstellung passieren. Dies macht es einfacher, Ihre lokale Firewall so zu konfigurieren, dass inbound/outbound Konnektivität zu dieser bekannten, eingeschränkten Gruppe von Verbindungen möglich ist. IPs Um zu steuern, in welchen Subnetzen Netzwerkschnittstellen erstellt werden, können Sie die Anzahl der Subnetze begrenzen, die Sie beim Erstellen eines Clusters angeben, oder Sie können die Subnetze nach dem Erstellen des Clusters aktualisieren.

Die von Amazon EKS bereitgestellten Netzwerkschnittstellen verfügen über eine Beschreibung im Format `Amazon EKS your-cluster-name` . Im folgenden Beispiel finden Sie einen AWS CLI-Befehl, mit dem Sie die IP-Adressen der Netzwerkschnittstellen ermitteln können, die Amazon EKS bereitstellt. Ersetzen Sie `VPC_ID` durch die ID der VPC, die Sie bei der Cluster-Erstellung übergeben haben.

```
aws ec2 describe-network-interfaces \
--query 'NetworkInterfaces[?(VpcId == VPC_ID && contains(Description,Amazon
 EKS))].PrivateIpAddress'
```

AWS VPC- und Subnetz-Setup

Die vorhandenen VPC- und Subnetz-Anforderungen für Amazon EKS gelten für Cluster mit Hybridknoten. Darüber hinaus darf sich Ihr VPC-CIDR nicht mit Ihrem lokalen Knoten und Pod überschneiden. CIDRs Sie müssen Routen in Ihrer VPC-Routingtabelle für Ihren lokalen Knoten und optional den Pod konfigurieren. CIDRs Diese Routen müssen so eingerichtet werden, dass der Datenverkehr an das Gateway weitergeleitet wird, das Sie für Ihre Hybridnetzwerkkonnektivität verwenden. Dabei handelt es sich üblicherweise um ein Virtual Private Gateway (VGW) oder ein Transit Gateway (TGW). Wenn Sie TGW oder VGW verwenden, um Ihre VPC mit Ihrer On-Premises-Umgebung zu verbinden, müssen Sie eine TGW- oder VGW-Anbindung für Ihre VPC erstellen. Ihre VPC muss DNS-Hostnamen und die DNS-Auflösung unterstützen.

Die folgenden Schritte verwenden die AWS CLI. Sie können diese Ressourcen auch in AWS-Managementkonsole oder mit anderen Schnittstellen wie AWS CloudFormation AWS CDK oder Terraform erstellen.

Schritt 1: VPC erstellen

1. Führen Sie den folgenden Befehl aus, um eine VPC zu erstellen. Ersetzen Sie `VPC_CIDR` durch einen IPv4-RFC-1918-CIDR-Bereich (privat) oder einen Nicht-RFC-1918-CIDR-Bereich (öffentlich) (zum Beispiel `10.0.0.0/16`). Hinweis: Die DNS-Auflösung, eine EKS-Anforderung, ist für die VPC standardmäßig aktiviert.

```
aws ec2 create-vpc --cidr-block VPC_CIDR
```

2. Aktivieren Sie DNS-Host-Namen für Ihre VPC. Beachten Sie, dass die DNS-Auflösung für die VPC standardmäßig aktiviert ist. Ersetzen Sie `VPC_ID` durch die ID der VPC, die Sie im vorherigen Schritt erstellt haben.

```
aws ec2 modify-vpc-attribute --vpc-id VPC_ID --enable-dns-hostnames
```

Schritt 2: Subnetze erstellen

Erstellen Sie mindestens 2 Subnetze. Amazon EKS verwendet diese Subnetze für die Cluster-Netzwerkschnittstellen. Weitere Informationen finden Sie unter Anforderungen und Überlegungen zu Subnetzen.

1. Sie können die Verfügbarkeitszonen für eine AWS Region mit dem folgenden Befehl finden. Ersetzen Sie `us-west-2` durch Ihre Region.

```
aws ec2 describe-availability-zones \
     --query 'AvailabilityZones[?(RegionName == us-west-2)].ZoneName'
```

2. Erstellen Sie ein Subnetz. Ersetzen Sie `VPC_ID` durch die ID der VPC. Ersetzen Sie `SUBNET_CIDR` durch den CIDR-Block für Ihr Subnetz (z. B. 10.0.1.0/24). Ersetzen Sie `AZ` durch die Availability Zone, in der das Subnetz erstellt werden soll (z. B. „us-west-2a“). Die von Ihnen erstellten Subnetze müssen sich in mindestens zwei verschiedenen Availability Zones befinden.

```
aws ec2 create-subnet \
    --vpc-id VPC_ID \
    --cidr-block SUBNET_CIDR \
    --availability-zone AZ
```

(Optional) Schritt 3: Verbinden Sie VPC mit Amazon VPC Transit Gateway (TGW) oder AWS Direct Connect Virtual Private Gateway (VGW)

Wenn Sie ein TGW oder VGW verwenden, fügen Sie Ihr VPC an das TGW oder VGW an. Weitere Informationen finden Sie unter Amazon-VPC-Anhänge in Amazon VPC Transit Gateways oder Zuordnungen von AWS Direct Connect Virtual Private Gateway.

Transit-Gateway

Führen Sie den folgenden Befehl aus, um ein Transit Gateway anzufügen. Ersetzen Sie `VPC_ID` durch die ID der VPC. Ersetzen Sie `SUBNET_ID1` und `SUBNET_ID2` durch die Subnetze, IDs die Sie im vorherigen Schritt erstellt haben. Ersetzen Sie `TGW_ID` durch die ID Ihres TGW.

```
aws ec2 create-transit-gateway-vpc-attachment \
    --vpc-id VPC_ID \
    --subnet-ids SUBNET_ID1 SUBNET_ID2 \
    --transit-gateway-id TGW_ID
```

Virtuelles privates Gateway

Führen Sie den folgenden Befehl aus, um ein Transit Gateway anzufügen. Ersetzen Sie `VPN_ID` durch die ID Ihres VGW. Ersetzen Sie `VPC_ID` durch die ID der VPC.

```
aws ec2 attach-vpn-gateway \
    --vpn-gateway-id VPN_ID \
    --vpc-id VPC_ID
```

(Optional) Schritt 4: Routing-Tabelle erstellen

Sie können die Haupt-Routing-Tabelle für die VPC ändern oder eine benutzerdefinierte Routing-Tabelle erstellen. Mit den folgenden Schritten wird eine benutzerdefinierte Routentabelle mit den Routen zum lokalen Knoten und Pod CIDRs erstellt. Weitere Informationen finden Sie unter Subnetz-Routing-Tabellen. Ersetzen Sie `VPC_ID` durch die ID der VPC.

```
aws ec2 create-route-table --vpc-id VPC_ID
```

Schritt 5: Routen für On-Premises-Knoten und -Pods erstellen

Erstellen Sie Routen in der Routing-Tabelle für jeden Ihrer On-Premises-Fern-Knoten. Sie können die Haupt-Routing-Tabelle für die VPC ändern oder die benutzerdefinierte Routing-Tabelle verwenden, die Sie im vorherigen Schritt erstellt haben.

Die folgenden Beispiele zeigen, wie Sie Routen für Ihren lokalen Knoten und Pod erstellen. CIDRs In den Beispielen wird ein Transit Gateway (TGW) verwendet, um die VPC mit der On-Premises-Umgebung zu verbinden. Wenn Sie mehrere lokale Knoten und Pods haben CIDRs, wiederholen Sie die Schritte für jeden CIDR.

- Wenn Sie ein Internet-Gateway oder ein Virtual Private Gateway (VGW) verwenden, ersetzen Sie `--transit-gateway-id` durch `--gateway-id`.
- Ersetzen Sie `RT_ID` durch die ID der Routing-Tabelle, die Sie im vorherigen Schritt erstellt haben.
- Ersetzen Sie `REMOTE_NODE_CIDR` durch den CIDR-Bereich, den Sie für Ihre Hybridknoten verwenden werden.
- Ersetzen Sie `REMOTE_POD_CIDR` durch den CIDR-Bereich, den Sie für die auf Hybridknoten ausgeführten Pods verwenden. Der Pod-CIDR-Bereich entspricht der Container Networking Interface (CNI)-Konfiguration, die in der Regel ein Überlagerungsnetzwerk On-Premises verwendet. Weitere Informationen finden Sie unter the section called “CNI konfigurieren”.
- Ersetzen Sie `TGW_ID` durch die ID Ihres TGW.

Fern-Knoten-Netzwerk

```
aws ec2 create-route \
    --route-table-id RT_ID \
    --destination-cidr-block REMOTE_NODE_CIDR \
    --transit-gateway-id TGW_ID
```

Fern-Pod-Netzwerk

```
aws ec2 create-route \
    --route-table-id RT_ID \
    --destination-cidr-block REMOTE_POD_CIDR \
    --transit-gateway-id TGW_ID
```

(Optional) Schritt 6: Subnetze der Routing-Tabelle zuordnen

Wenn Sie im vorherigen Schritt eine benutzerdefinierte Routing-Tabelle erstellt haben, ordnen Sie jede der im vorherigen Schritt erstellten Subnetze Ihrer benutzerdefinierten Routing-Tabelle zu. Wenn Sie die VPC-Haupt-Routing-Tabelle ändern, werden die Subnetze automatisch der Haupt-Routing-Tabelle der VPC zugeordnet, und Sie können diesen Schritt überspringen.

Führen Sie den folgenden Befehl für jedes der Subnetze aus, die Sie in den vorherigen Schritten erstellt haben. Ersetzen Sie `RT_ID` durch die Routing-Tabelle, die Sie im vorherigen Schritt erstellt haben. Ersetzen Sie `SUBNET_ID` durch die ID eines Subnetzes.

```
aws ec2 associate-route-table --route-table-id RT_ID --subnet-id SUBNET_ID
```

Konfiguration der Cluster-Sicherheitsgruppe

Für den fortlaufenden Cluster-Betrieb ist der folgende Zugriff für Ihre EKS-Cluster-Sicherheitsgruppe erforderlich. Amazon EKS erstellt automatisch die erforderlichen Regeln für eingehende Sicherheitsgruppen für Hybridknoten, wenn Sie Ihren Cluster mit konfigurierten Fern-Knoten- und Pod-Netzwerken erstellen oder aktualisieren. Da Sicherheitsgruppen standardmäßig den gesamten ausgehenden Datenverkehr zulassen, ändert Amazon EKS die ausgehenden Regeln der Cluster-Sicherheitsgruppe für Hybridknoten nicht automatisch. Wenn Sie die Cluster-Sicherheitsgruppe anpassen möchten, können Sie den Datenverkehr auf die Regeln in der folgenden Tabelle beschränken.

Typ	Protocol (Protokoll)	Richtung	Port	Quelle	Ziel	Usage
HTTPS	TCP	Eingehend	443	Fern-Knoten-CIDR(s)	–	Kubelet zum Kubernetes API-Server
HTTPS	TCP	Eingehend	443	Fern-Pod-CIDR(s)	–	Pods, die Zugriff auf den K8s-API-Server erfordern

Typ	Protocol (Protokoll)	Richtung	Port	Quelle	Ziel	Usage
						, wenn das CNI kein NAT für den Pod-Datenverkehr verwendet.
HTTPS	TCP	Ausgehend	10250	–	Fern-Knoten-CIDR(s)	Kubernetes-API-Server zu Kubelet
HTTPS	TCP	Ausgehend	Webhook-Ports	–	Fern-Pod-CIDR(s)	Kubernetes-API-Server zum Webhook (beim Ausführen von Webhooks in Hybridknoten)

Important

Grenzwerte für Sicherheitsgruppenregeln: EC2 Amazon-Sicherheitsgruppen haben standardmäßig maximal 60 Regeln für eingehenden Datenverkehr. Die eingehenden Regeln der Sicherheitsgruppe werden möglicherweise nicht angewendet, wenn die Sicherheitsgruppe Ihres Clusters diese Beschränkung erreicht. In diesem Fall kann es erforderlich sein, die fehlenden eingehenden Regeln manuell hinzuzufügen. Verantwortung für die Bereinigung von CIDR: Wenn Sie Fern-Knoten- oder Pod-Netzwerke aus EKS-Clustern entfernen, löscht EKS die entsprechenden Sicherheitsgruppenregeln

nicht automatisch. Sie sind dafür verantwortlich, nicht verwendete Fern-Knoten- oder Pod-Netzwerke manuell aus Ihren Sicherheitsgruppenregeln zu entfernen.

Weitere Informationen zu der Cluster-Sicherheitsgruppe, die Amazon EKS erstellt, finden Sie unter the section called “Anforderungen an Sicherheitsgruppen”.

(Optional) Manuelle Konfiguration der Sicherheitsgruppe

Sollten Sie zusätzliche Sicherheitsgruppen erstellen oder die automatisch erstellten Regeln ändern müssen, können Sie die folgenden Befehle als Referenz verwenden. Standardmäßig erstellt der folgende Befehl eine Sicherheitsgruppe, die den gesamten ausgehenden Datenverkehr zulässt. Sie können den ausgehenden Zugriff einschränken, sodass nur die oben genannten Regeln gelten. Wenn Sie eine Einschränkung der ausgehenden Regeln in Erwägung ziehen, empfehlen wir Ihnen, alle Ihre Anwendungen und die Pod-Konnektivität gründlich zu testen, bevor Sie Ihre geänderten Regeln auf einen Produktions-Cluster anwenden.

- Ersetzen Sie im ersten Befehl `SG_NAME` durch einen Namen für Ihre Sicherheitsgruppe
- Ersetzen Sie im ersten Befehl `VPC_ID` durch die ID der VPC, die Sie im vorherigen Schritt erstellt haben.
- Ersetzen Sie im zweiten Befehl `SG_ID` durch die ID der Sicherheitsgruppe, die Sie im ersten Befehl erstellt haben
- Ersetzen Sie im zweiten Befehl `REMOTE_NODE_CIDR` und `REMOTE_POD_CIDR` durch die Werte für Ihre Hybridknoten und Ihr On-Premises-Netzwerk.

```
aws ec2 create-security-group \
    --group-name SG_NAME \
    --description "security group for hybrid nodes" \
    --vpc-id VPC_ID
```

```
aws ec2 authorize-security-group-ingress \
    --group-id SG_ID \
    --ip-permissions '[{"IpProtocol": "tcp", "FromPort": 443, "ToPort": 443,
 "IpRanges": [{"CidrIp": "REMOTE_NODE_CIDR"}, {"CidrIp": "REMOTE_POD_CIDR"}]}]'
```

Vorbereitung des Betriebssystems für Hybridknoten

Bottlerocket, Amazon Linux 2023 (AL2023), Ubuntu und RHEL werden fortlaufend für die Verwendung als Knotenbetriebssystem für Hybridknoten validiert. Bottlerocket wird nur AWS in VMware vSphere-Umgebungen unterstützt. AL2023 ist nicht durch AWS Supportpläne abgedeckt, wenn sie außerhalb von Amazon EC2 ausgeführt werden. AL2023 kann nur in lokalen virtualisierten Umgebungen verwendet werden. Weitere Informationen finden Sie im Amazon Linux 2023 User Guide. AWS unterstützt die Integration von Hybridknoten mit Ubuntu- und RHEL-Betriebssystemen, bietet jedoch keine Unterstützung für das Betriebssystem selbst.

Sie sind für die Bereitstellung und Verwaltung des Betriebssystems verantwortlich. Wenn Sie Hybridknoten zum ersten Mal testen, ist es am einfachsten, die CLI für Amazon EKS Hybrid Nodes (`nodeadm`) auf einem bereits bereitgestellten Host auszuführen. Für Produktions-Bereitstellungen empfehlen wir, `nodeadm` in Ihre Betriebssystem-Images aufzunehmen und es so zu konfigurieren, dass es als systemd-Service ausgeführt wird, um Hosts beim Host-Startup automatisch mit Amazon-EKS-Clustern zu verbinden. Wenn Sie Bottlerocket als Knotenbetriebssystem auf vSphere verwenden, müssen Sie `nodeadm` nicht verwenden, da Bottlerocket bereits die für Hybridknoten erforderlichen Abhängigkeiten enthält und beim Host-Startup automatisch eine Verbindung mit dem von Ihnen konfigurierten Cluster herstellt.

Versionskompatibilität

Die nachfolgende Tabelle enthält die Betriebssystemversionen, die kompatibel und für die Verwendung als Knoten-Betriebssystem für Hybridknoten validiert sind. Wenn Sie andere Betriebssystemvarianten oder Versionen verwenden, die nicht in dieser Tabelle enthalten sind, wird die Kompatibilität von Hybridknoten mit Ihrer Betriebssystemvariante oder -version nicht vom AWS Support abgedeckt. Hybridknoten sind unabhängig von der zugrunde liegenden Infrastruktur und unterstützen x86- und ARM-Architekturen.

Betriebssystem	Versionen
Amazon Linux	Amazon Linux 2023 (AL2023)
Flaschenrakete	v1.37.0 und höhere VMware Varianten, auf denen Kubernetes v1.28 und höher ausgeführt wird
Ubuntu	Ubuntu 20.04, Ubuntu 22.04, Ubuntu 24.04

Betriebssystem	Versionen
Red Hat Enterprise Linux	RHEL 8, RHEL 9

Überlegungen zum Betriebssystem

General

- Die Amazon EKS Hybrid Nodes CLI (`nodeadm`) kann verwendet werden, um die Installation und Konfiguration der Hybridknoten-Komponenten und -Abhängigkeiten zu vereinfachen. Sie können den `nodeadm install`-Prozess während der Erstellung Ihrer Betriebssystem-Image-Pipelines oder zur Laufzeit auf jedem On-Premises-Host ausführen. Weitere Informationen zu den von `nodeadm` installierten Komponenten finden Sie unter the section called “nodeadm für Hybridknoten”.
- Wenn Sie in Ihrer On-Premises-Umgebung einen Proxy für den Internetzugang verwenden, sind zusätzliche Betriebssystemkonfigurationen für die Installations- und Aktualisierungsprozesse erforderlich, um Ihren Paket-Manager für die Verwendung des Proxys zu konfigurieren. Detaillierte Anweisungen finden Sie unter the section called “Proxy konfigurieren”.

Bottlerocket

- Die Schritte und Tools zum Verbinden eines Bottlerocket-Knotens unterscheiden sich von den Schritten für andere Betriebssysteme und werden separat in the section called “Hybridknoten mit Bottlerocket verbinden” behandelt, anstatt in den Schritten in the section called “Hybridknoten verbinden”.
- Die Schritte für Bottlerocket verwenden nicht das CLI-Tool für Hybridknoten, `nodeadm`.
- Nur VMware Varianten von Bottlerocket Version v1.37.0 und höher werden von EKS Hybrid Nodes unterstützt. VMware Varianten von Bottlerocket sind für die Kubernetes-Versionen v1.28 und höher verfügbar. Andere Bottlerocket-Varianten werden als Betriebssystem für Hybridknoten nicht unterstützt. HINWEIS: VMware Varianten von Bottlerocket sind nur für die x86_64-Architektur verfügbar.

Containerd

- Containerd ist die Standard-Container-Laufzeit von Kubernetes und eine Abhängigkeit für Hybridknoten sowie alle Amazon-EKS-Knoten-Rechentypen. Die CLI für Amazon EKS Hybrid

Nodes (`nodeadm`) versucht während des `nodeadm install`-Vorgangs, containerd zu installieren. Sie können die containerd-Installation zur `nodeadm install`-Laufzeit mit der Befehlszeilenoption `--containerd-source` konfigurieren. Gültige Optionen sind `none`, `distro` und `docker`. Wenn Sie RHEL verwenden, ist `distro` keine gültige Option und Sie können `nodeadm` entweder so konfigurieren, dass die containerd-Entwicklung aus den Docker-Repos installiert wird. Sie können containerd auch manuell installieren. Wenn Sie AL2 023 oder Ubuntu verwenden, wird containerd `nodeadm` standardmäßig aus der Betriebssystemdistribution installiert. Wenn Sie nicht möchten, dass nodeadm containerd installiert, verwenden Sie die `--containerd-source none`-Option.

Ubuntu

- Wenn Sie Ubuntu 24.04 verwenden, müssen Sie möglicherweise Ihre Version von containerd aktualisieren oder Ihre AppArmor Konfiguration ändern, um einen Fix zu implementieren, mit dem Pods ordnungsgemäß beendet werden können, siehe Ubuntu #2065423. Ein Neustart ist erforderlich, um die Änderungen am Profil zu übernehmen. AppArmor Die neueste Version von Ubuntu 24.04 enthält in ihrem Paket-Manager eine aktualisierte containerd-Version mit der Korrektur (Containerd-Version 1.7.19+).

ARM

- Wenn Sie ARM-Hardware verwenden, ist ein ARMv8 2.2-kompatibler Prozessor mit der Cryptography Extension (ARMv8.2+Crypto) erforderlich, um Version 1.31 und höher des EKS-Kube-Proxy-Add-Ons auszuführen. Alle Raspberry-Pi-Systeme vor dem Raspberry Pi 5 sowie Cortex-A72-basierte Prozessoren erfüllen diese Anforderung nicht. Als vorübergehende Lösung können Sie die Version 1.30 des EKS-kube-proxy-Add-Ons weiterhin verwenden, bis der erweiterte Support im Juli 2026 endet (siehe Kubernetes-Versions-Kalender), oder ein benutzerdefiniertes kube-proxy-Image aus dem Upstream verwenden.
- Die folgende Fehlermeldung im kube-proxy-Protokoll weist auf diese Inkompatibilität hin:

```
Fatal glibc error: This version of Amazon Linux requires a newer ARM64 processor
 compliant with at least ARM architecture 8.2-a with Cryptographic extensions. On EC2
 this is Graviton 2 or later.
```

Erstellung von Betriebssystem-Images

Amazon EKS bietet Beispielvorlagen für Packer, mit denen Sie Betriebssystem-Images erstellen können, die `nodeadm` enthalten, und diese so konfigurieren können, dass sie beim Startup des Hosts ausgeführt werden. Dieser Prozess wird empfohlen, um zu vermeiden, dass die Abhängigkeiten der Hybridknoten einzeln auf jedem Host abgerufen werden müssen, und um den Bootstrap-Prozess der Hybridknoten zu automatisieren. Sie können die Beispielvorlagen für Packer mit einem ISO-Image von Ubuntu 22.04, Ubuntu 24.04, RHEL 8 oder RHEL 9 verwenden und Images in den folgenden Formaten ausgeben: OVA, Qcow2 oder raw.

Voraussetzungen

Bevor Sie die Beispielvorlagen für Packer verwenden können, müssen Sie die folgenden Komponenten auf dem Computer installieren, auf dem Sie Packer ausführen.

- Packer-Version 1.11.0 oder höher. Anweisungen zur Installation von Packer finden Sie unter Installation von Packer in der Packer-Dokumentation.
- Beim Erstellen das OVAs VMware vSphere-Plug-in 1.4.0 oder höher
- Bei Erstellung von `Qcow2`- oder RAW-Images: QEMU-Plugin Version 1.x.

Festlegen von Umgebungsvariablen

Legen Sie vor Ausführung der Packer-Entwicklung die folgenden Umgebungsvariablen auf dem Computer fest, von dem aus Sie Packer ausführen.

General

Die folgenden Umgebungsvariablen müssen für die Erstellung von Images mit allen Betriebssystemen und Ausgabeformaten festgelegt werden.

Umgebungsvariable	Typ	Description
PKR_SSH_PASSWORD	Zeichenfolge	Packer verwendet die Variablen `ssh_username` und `ssh_password` , um bei der Bereitstellung per SSH auf den erstellten Rechner zuzugreifen. Dies muss mit den Passwörtern übereinst

Umgebungsvariable	Typ	Description
		immen, die zur Erstellung des ersten Benutzers in den Kickstart- oder Benutzerd atendateien des jeweiligen Betriebssystems verwendet wurden. Der Standardwert ist je nach Betriebssystem „builder“ oder „ubuntu“. Achten Sie bei der Festlegung Ihres Passworts darauf, es in der entsprechenden `ks.cfg`- oder `user-data` -Datei entsprech end anzupassen.
ISO_URL	Zeichenfolge	URL der zu verwendenden ISO. Dies kann ein Weblink zum Herunterladen von einem Server oder ein absoluter Pfad zu einer lokalen Datei sein
ISO_CHECKSUM	Zeichenfolge	Zugehörige Prüfsumme für die bereitgestellte ISO-Datei.
CREDENTIAL_PROVIDER	Zeichenfolge	Anmeldeinformationsanbieter für Hybridknoten. Gültige Werte sind `ssm` (Standard) für SSM-Hybridaktivierungen und für `iam` IAM Roles Anywhere.
K8S_VERSION	Zeichenfolge	Kubernetes-Version für Hybridknoten (z. B. `1.31`). Informationen zu den unterstüt zten Kubernetes-Versionen finden Sie unter Unterstützte Amazon-EKS-Versionen.

Umgebungsvariable	Typ	Description
NODEADM_ARCH	Zeichenfolge	Architektur für `nodeadm install`: Wählen Sie `amd` oder `arm`.

RHEL

Wenn Sie RHEL verwenden, müssen die folgenden Umgebungsvariablen festgelegt werden.

Umgebungsvariable	Typ	Description
RH_USERNAME	Zeichenfolge	Benutzername des RHEL-Abonnement-Managers
RH_PASSWORD	Zeichenfolge	Kennwort für den RHEL-Abonnement-Manager
RHEL_VERSION	Zeichenfolge	Verwendete Rhel-ISO-Version. Gültige Werte sind 8 oder 9.

Ubuntu

Es sind keine Ubuntu-spezifischen Umgebungsvariablen erforderlich.

vSphere

Wenn Sie eine VMware vSphere-OVA erstellen, müssen die folgenden Umgebungsvariablen festgelegt werden.

Umgebungsvariable	Typ	Description
VSPHERE_SERVER	Zeichenfolge	vSphere-Server-Adresse
VSPHERE_USER	Zeichenfolge	vSphere-Benutzername
VSPHERE_PASSWORD	Zeichenfolge	vSphere-Kennwort

Umgebungsvariable	Typ	Description
VSPHERE_DATACENTER	Zeichenfolge	Name des vSphere-Rechenzentrums
VSPHERE_CLUSTER	Zeichenfolge	vSphere-Cluster-Name
VSPHERE_DATASTORE	Zeichenfolge	Name des vSphere-Datenspeichers
VSPHERE_NETWORK	Zeichenfolge	vSphere-Netzwerkname
VSPHERE_OUTPUT_FOLDER	Zeichenfolge	vSphere-Ausgabeordner für die Vorlagen

QEMU

Umgebungsvariable	Typ	Description
PACKER_OUTPUT_FORMAT	Zeichenfolge	Ausgabeformat für QEMU-Entwickler. Gültige Werte sind `qcow2` und `raw`.

Vorlage validieren

Bevor Sie Ihre Entwicklung ausführen, überprüfen Sie Ihre Vorlage mit dem folgenden Befehl, nachdem Sie Ihre Umgebungsvariablen festgelegt haben. Ersetzen Sie `template.pkr.hcl`, falls Sie einen anderen Namen für Ihre Vorlage verwenden.

```
packer validate template.pkr.hcl
```

Entwicklungs-Images

Erstellen Sie Ihre Images mit den folgenden Befehlen und verwenden Sie das `-only`-Flag, um das Ziel und das Betriebssystem für Ihre Images anzugeben. Ersetzen Sie `template.pkr.hcl`, falls Sie einen anderen Namen für Ihre Vorlage verwenden.

vSphere OVAs

> **Note**
>
> Wenn Sie RHEL mit vSphere verwenden, müssen Sie die Kickstart-Dateien in ein OEMDRV-Image konvertieren und es als ISO zum Booten übergeben. Weitere Informationen finden Sie in der Packer-Readme-Datei im EKS Hybrid Nodes Repository. GitHub

Ubuntu 22.04 OVA

```
packer build -only=general-build.vsphere-iso.ubuntu22 template.pkr.hcl
```

Ubuntu 24.04 OVA

```
packer build -only=general-build.vsphere-iso.ubuntu24 template.pkr.hcl
```

RHEL 8 OVA

```
packer build -only=general-build.vsphere-iso.rhel8 template.pkr.hcl
```

RHEL 9 OVA

```
packer build -only=general-build.vsphere-iso.rhel9 template.pkr.hcl
```

QEMU

> **Note**
>
> Wenn Sie ein Image für eine bestimmte Host-CPU erstellen, die nicht mit Ihrem Builder-Host übereinstimmt, suchen Sie in der QEMU-Dokumentation nach dem Namen, der Ihrer Host-CPU entspricht, und verwenden Sie das `-cpu`-Flag mit dem Namen der Host-CPU, wenn Sie die folgenden Befehle ausführen.

Ubuntu 22.04 Qcow2 / Raw

```
packer build -only=general-build.qemu.ubuntu22 template.pkr.hcl
```

Ubuntu 24.04 Qcow2 / Raw

```
packer build -only=general-build.qemu.ubuntu24 template.pkr.hcl
```

RHEL 8 Qcow2 / Raw

```
packer build -only=general-build.qemu.rhel8 template.pkr.hcl
```

RHEL 9 Qcow2 / Raw

```
packer build -only=general-build.qemu.rhel9 template.pkr.hcl
```

nodeadm-Konfiguration über Benutzerdaten übergeben

Sie können die Konfiguration für `nodeadm` in Ihren Benutzerdaten über cloud-init übergeben, um Hybridknoten zu konfigurieren und beim Startup des Hosts automatisch mit Ihrem EKS-Cluster zu verbinden. Im Folgenden finden Sie ein Beispiel dafür, wie Sie dies erreichen können, wenn Sie VMware vSphere als Infrastruktur für Ihre Hybridknoten verwenden.

1. Installieren Sie die `govc` CLI gemäß den Anweisungen in der Govc-Readme-Datei unter. GitHub
2. Nachdem Sie die Packer-Entwicklung im vorherigen Abschnitt ausgeführt und Ihre Vorlage bereitgestellt haben, können Sie Ihre Vorlage klonen, um mithilfe der folgenden Schritte mehrere verschiedene Knoten zu erstellen. Sie müssen die Vorlage für jede neue VM klonen, die Sie für Hybridknoten erstellen. Ersetzen Sie die Variablen im folgenden Befehl durch die Werte für Ihre Umgebung. Der `VM_NAME` im folgenden Befehl enthaltene Befehl wird als Ihr verwendet`NODE_NAME`, wenn Sie die Namen für Sie VMs über Ihre Datei eingeben. `metadata.yaml`

   ```
   govc vm.clone -vm "/PATH/TO/TEMPLATE" -ds="YOUR_DATASTORE" \
       -on=false -template=false -folder=/FOLDER/TO/SAVE/VM "VM_NAME"
   ```

3. Nachdem Sie die Vorlage für jedes Ihrer neuen geklont haben VMs, erstellen Sie ein `userdata.yaml` und `metadata.yaml` für Ihre. VMs Sie VMs können dasselbe teilen `userdata.yaml` `metadata.yaml` und Sie werden diese in den folgenden Schritten für jede VM einzeln ausfüllen. Die `nodeadm`-Konfiguration wird im `write_files`-Abschnitt Ihres `userdata.yaml` erstellt und definiert. Im folgenden Beispiel werden AWS SSM-Hybrid-Aktivierungen als lokaler Anbieter von Anmeldeinformationen für Hybridknoten verwendet. Weitere Informationen zur `nodeadm`-Konfiguration finden Sie unter the section called “nodeadm für Hybridknoten”.

userdata.yaml:

```
#cloud-config
users:
  - name: # username for login. Use 'builder' for RHEL or 'ubuntu' for Ubuntu.
    passwd: # password to login. Default is 'builder' for RHEL.
    groups: [adm, cdrom, dip, plugdev, lxd, sudo]
    lock-passwd: false
    sudo: ALL=(ALL) NOPASSWD:ALL
    shell: /bin/bash

write_files:
  - path: /usr/local/bin/nodeConfig.yaml
    permissions: '0644'
    content: |
      apiVersion: node.eks.aws/v1alpha1
      kind: NodeConfig
      spec:
          cluster:
              name: # Cluster Name
              region: # AWS region
          hybrid:
              ssm:
                  activationCode: # Your ssm activation code
                  activationId: # Your ssm activation id

runcmd:
  - /usr/local/bin/nodeadm init -c file:///usr/local/bin/nodeConfig.yaml >> /var/log/
nodeadm-init.log 2>&1
```

metadata.yaml:

Erstellen Sie einen `metadata.yaml` für Ihre Umgebung. Behalten Sie das Variablenformat `"$NODE_NAME"` in der Datei bei, da dieses in einem nachfolgenden Schritt mit Werten gefüllt wird.

```
instance-id: "$NODE_NAME"
local-hostname: "$NODE_NAME"
network:
  version: 2
  ethernets:
    nics:
      match:
```

```
        name: ens*
      dhcp4: yes
```

4. Fügen Sie die `userdata.yaml`- und `metadata.yaml`-Dateien als `gzip+base64`-Zeichenfolgen mit den folgenden Befehlen hinzu. Die folgenden Befehle sollten für jeden Befehl ausgeführt werden, den VMs Sie erstellen. Ersetzen Sie `VM_NAME` durch den Namen der VM, die Sie aktualisieren.

```
export NODE_NAME="VM_NAME"
export USER_DATA=$(gzip -c9 <userdata.yaml | base64)

govc vm.change -dc="YOUR_DATASTORE" -vm "$NODE_NAME" -e
 guestinfo.userdata="${USER_DATA}"
govc vm.change -dc="YOUR_DATASTORE" -vm "$NODE_NAME" -e
 guestinfo.userdata.encoding=gzip+base64

envsubst '$NODE_NAME' < metadata.yaml > metadata.yaml.tmp
export METADATA=$(gzip -c9 <metadata.yaml.tmp | base64)

govc vm.change -dc="YOUR_DATASTORE" -vm "$NODE_NAME" -e
 guestinfo.metadata="${METADATA}"
govc vm.change -dc="YOUR_DATASTORE" -vm "$NODE_NAME" -e
 guestinfo.metadata.encoding=gzip+base64
```

5. Schalten Sie Ihren neuen ein VMs, der automatisch eine Verbindung zu dem von Ihnen konfigurierten EKS-Cluster herstellen sollte.

```
govc vm.power -on "${NODE_NAME}"
```

Vorbereitung der Anmeldeinformationen für Hybridknoten

Amazon EKS-Hybridknoten verwenden temporäre IAM-Anmeldeinformationen, die durch AWS SSM-Hybrid-Aktivierungen oder AWS IAM Roles Anywhere bereitgestellt werden, um sich beim Amazon EKS-Cluster zu authentifizieren. Sie müssen entweder AWS SSM-Hybrid-Aktivierungen oder AWS IAM Roles Anywhere mit der Amazon EKS Hybrid Nodes CLI () verwenden. `nodeadm` Sie sollten nicht sowohl AWS SSM-Hybrid-Aktivierungen als auch IAM-Rollen Anywhere verwenden. AWS Wir empfehlen, AWS SSM-Hybrid-Aktivierungen zu verwenden, wenn Sie nicht über eine bestehende Public Key-Infrastruktur (PKI) mit einer Zertifizierungsstelle (CA) und Zertifikaten für Ihre lokalen Umgebungen verfügen. Wenn Sie bereits über PKI und Zertifikate vor Ort verfügen, verwenden Sie IAM Roles Anywhere. AWS

IAM-Rolle für Hybridknoten

Bevor Sie Hybridknoten mit Ihrem Amazon EKS-Cluster verbinden können, müssen Sie eine IAM-Rolle erstellen, die mit AWS SSM-Hybrid-Aktivierungen oder AWS IAM Roles Anywhere für Ihre Hybridknoten-Anmeldeinformationen verwendet wird. Nach der Clustererstellung verwenden Sie diese Rolle mit einem Amazon EKS-Zugriffseintrag oder `aws-auth` ConfigMap einem Eintrag, um die IAM-Rolle Kubernetes Role-Based Access Control (RBAC) zuzuordnen. Weitere Informationen zum Verknüpfen der IAM-Rolle für Hybridknoten mit Kubernetes RBAC finden Sie unter the section called “Cluster-Zugriff vorbereiten”.

Die IAM-Rolle für Hybridknoten muss auch über die folgenden Berechtigungen verfügen.

- Berechtigungen für die Verwendung der `eks:DescribeCluster` Aktion `nodeadm` zum Sammeln von Informationen über den Cluster, mit dem Sie Hybridknoten verbinden möchten. Wenn Sie die `eks:DescribeCluster` Aktion nicht aktivieren, müssen Sie Ihren Kubernetes-API-Endpunkt, das Cluster-CA-Bundle und den IPv4 Service-CIDR in der Knotenkonfiguration übergeben, die Sie an den Befehl übergeben. `nodeadm init`
- Berechtigungen für `nodeadm` die Verwendung der `eks:ListAccessEntries` Aktion zum Auflisten der Zugriffseinträge auf dem Cluster, mit dem Sie Hybridknoten verbinden möchten. Wenn Sie die `eks:ListAccessEntries` Aktion nicht aktivieren, müssen Sie das `--skip cluster-access-validation` Flag übergeben, wenn Sie den `nodeadm init` Befehl ausführen.
- Berechtigungen für das Kubelet zur Verwendung von Container-Images aus Amazon Elastic Container Registry (Amazon ECR), wie in der Amazon-Richtlinie definiert. EC2 ContainerRegistryPullOnly
- Bei Verwendung von AWS SSM: Berechtigungen für `nodeadm init` die Verwendung von AWS SSM-Hybrid-Aktivierungen, wie in der Richtlinie definiert. https://docs.aws.amazon.com/aws-managed-policy/latest/reference/AmazonSSMManagedInstanceCore.html
- Bei Verwendung von AWS SSM: Berechtigungen zur Verwendung der `ssm:DeregisterManagedInstance` Aktion und `ssm:DescribeInstanceInformation` Aktion für `nodeadm uninstall` zum Abmelden von Instanzen.
- (Optional) Berechtigungen für den Amazon EKS Pod Identity Agent zum Verwenden der Aktion `eks-auth:AssumeRoleForPodIdentity`, um Anmeldeinformationen für Pods abzurufen.

Richten Sie SSM-Hybrid-Aktivierungen AWS ein

Bevor Sie AWS SSM-Hybrid-Aktivierungen einrichten, müssen Sie eine IAM-Rolle für Hybrid Nodes erstellt und konfiguriert haben. Weitere Informationen finden Sie unter the section called “Erstellung der IAM-Rolle für Hybridknoten”. Folgen Sie den Anweisungen unter Hybrid-Aktivierung erstellen, um Knoten bei Systems Manager zu registrieren im AWS Systems Manager Manager-Benutzerhandbuch, um eine AWS SSM-Hybrid-Aktivierung für Ihre Hybridknoten zu erstellen. Der Aktivierungscode und die ID, die Sie erhalten, werden mit `nodeadm` verwendet, wenn Sie Ihre Hosts als Hybridknoten bei Ihrem Amazon-EKS-Cluster registrieren. Sie können zu einem späteren Zeitpunkt auf diesen Schritt zurückkommen, nachdem Sie Ihre Amazon-EKS-Cluster für Hybridknoten erstellt und vorbereitet haben.

> **Important**
>
> Systems Manager übergibt den Aktivierungscode und die ID sofort an die Konsole oder das Befehlsfenster, je nachdem, wie Sie die Aktivierung erstellt haben. Kopieren Sie diese Informationen und speichern Sie sie an einem sicheren Ort. Wenn Sie die Konsole verlassen oder das Befehlsfenster schließen, können diese Informationen verloren gehen. Wenn Sie die Informationen verlieren, müssen Sie eine neue Aktivierung erstellen.

Standardmäßig sind AWS SSM-Hybrid-Aktivierungen 24 Stunden lang aktiv. Alternativ können Sie beim Erstellen Ihrer Hybridaktivierung ein `--expiration-date` im Zeitstempelformat angeben, beispielsweise `2024-08-01T00:00:00`. Wenn Sie AWS SSM als Ihren Anmeldeinformationsanbieter verwenden, ist der Knotenname für Ihre Hybridknoten nicht konfigurierbar und wird automatisch von SSM generiert. AWS Sie können die von AWS SSM verwalteten Instanzen in der AWS Systems Manager-Konsole unter Fleet Manager anzeigen und verwalten. Sie können bis zu 1.000 standardmäßige, hybridaktivierte Knoten pro Konto und AWS Region ohne zusätzliche Kosten registrieren. Um mehr als 1 000 Hybrid-Knoten zu registrieren, müssen Sie jedoch das Advanced-Instances-Kontingent aktivieren. Für die Nutzung der Stufe „Erweiterte Instances“ wird eine Gebühr erhoben, die nicht in den Preisen für Amazon EKS Hybrid Nodes pricing enthalten ist. Weitere Informationen finden Sie unter Preise für AWS Systems Manager.

Im folgenden Beispiel erfahren Sie, wie Sie eine AWS SSM-Hybrid-Aktivierung mit Ihrer IAM-Rolle Hybrid Nodes erstellen. Wenn Sie AWS SSM-Hybrid-Aktivierungen für Ihre Hybridknoten-Anmeldeinformationen verwenden, haben die Namen Ihrer Hybridknoten das gleiche Format `mi-012345678abcdefgh` und die von AWS SSM bereitgestellten temporären

Anmeldeinformationen sind 1 Stunde lang gültig. Sie können den Knotennamen oder die Dauer der Anmeldeinformationen nicht ändern, wenn Sie AWS SSM als Ihren Anmeldeinformationsanbieter verwenden. Die temporären Anmeldeinformationen werden von AWS SSM automatisch rotiert, und die Rotation hat keine Auswirkungen auf den Status Ihrer Knoten oder Anwendungen.

Wir empfehlen, dass Sie eine AWS SSM-Hybrid-Aktivierung pro EKS-Cluster verwenden, um die AWS `ssm:DeregisterManagedInstance` SSM-Berechtigungen der IAM-Rolle Hybrid Nodes so festzulegen, dass nur Instances abgemeldet werden können, die mit Ihrer SSM-Hybrid-Aktivierung verknüpft sind. AWS In dem Beispiel auf dieser Seite wird ein Tag mit dem EKS-Cluster-ARN verwendet, das verwendet werden kann, um Ihre AWS SSM-Hybridaktivierung dem EKS-Cluster zuzuordnen. Sie können alternativ Ihr bevorzugtes Tag und Ihre bevorzugte Methode verwenden, um den Umfang der AWS SSM-Berechtigungen auf der Grundlage Ihrer Berechtigungsgrenzen und Anforderungen festzulegen. Die `REGISTRATION_LIMIT` Option im folgenden Befehl ist eine Ganzzahl, die verwendet wird, um die Anzahl der Maschinen zu begrenzen, die die AWS SSM-Hybridaktivierung verwenden können (zum Beispiel) `10`

```
aws ssm create-activation \
     --region AWS_REGION \
     --default-instance-name eks-hybrid-nodes \
     --description "Activation for EKS hybrid nodes" \
     --iam-role AmazonEKSHybridNodesRole \
     --tags Key=EKSClusterARN,Value=arn:aws: eks:AWS_REGION:AWS_ACCOUNT_ID:cluster/
CLUSTER_NAME \
     --registration-limit REGISTRATION_LIMIT
```

Weitere Informationen zu den verfügbaren Konfigurationseinstellungen für AWS SSM-Hybrid-Aktivierungen finden Sie in den Anweisungen unter Hybrid-Aktivierung erstellen, um Knoten bei Systems Manager zu registrieren.

Richten Sie AWS IAM-Rollen überall ein

Befolgen Sie die Anweisungen unter Erste Schritte mit IAM Roles Anywhere im Benutzerhandbuch zu IAM Roles Anywhere, um den Trust Anchor und das Profil einzurichten, die Sie für temporäre IAM-Anmeldeinformationen für Ihre IAM-Rolle für Hybridknoten verwenden werden. Wenn Sie Ihr Profil erstellen, können Sie dies ohne Hinzufügen von Rollen tun. Sie können dieses Profil erstellen, zu diesen Schritten zurückkehren, um Ihre IAM-Rolle für Hybridknoten zu erstellen, und dann Ihre Rolle nach der Erstellung zu Ihrem Profil hinzufügen. Sie können auch die AWS CloudFormation Schritte weiter unten auf dieser Seite verwenden, um Ihr IAM Roles Anywhere-Setup für Hybridknoten abzuschließen.

Wenn Sie Ihrem Profil die IAM-Rolle Hybrid Nodes hinzufügen, wählen Sie im Bereich Sitzungsname für benutzerdefinierte Rolle unten auf der Seite Profil bearbeiten in der AWS IAM Roles Anywhere-Konsole die Option Sitzungsname der benutzerdefinierten Rolle akzeptieren aus. Dies entspricht dem Feld „acceptRoleSessionName“ der `CreateProfile` API. Auf diese Weise können Sie in der Konfiguration, die Sie während des Bootstrap-Prozesses an `nodeadm` übergeben, einen benutzerdefinierten Knotennamen für Ihre Hybridknoten angeben. Die Übergabe eines benutzerdefinierten Knotennamens während des `nodeadm init`-Prozesses ist erforderlich. Sie können Ihr Profil aktualisieren, um einen benutzerdefinierten Rollensitzungsnamen zu akzeptieren, nachdem Sie Ihr Profil erstellt haben.

Sie können die Gültigkeitsdauer der Anmeldeinformationen mit AWS IAM Roles Anywhere über das Feld DurationSeconds Ihres AWS IAM Roles Anywhere-Profils konfigurieren. Die Standarddauer beträgt 1 Stunde, maximal 12 Stunden. Die `MaxSessionDuration` Einstellung in Ihrer IAM-Rolle für Hybrid Nodes muss größer sein als die `durationSeconds` Einstellung in Ihrem IAM Roles Anywhere-Profil. AWS Weitere Informationen dazu finden Sie in der `MaxSessionDuration` UpdateRole API-Dokumentation.

Die von Ihrer Zertifizierungsstelle (CA) generierten rechnerspezifischen Zertifikate und Schlüssel müssen in dem `/etc/iam/pki`-Verzeichnis auf jedem Hybridknoten mit den Dateinamen `server.pem` für das Zertifikat und `server.key` für den Schlüssel abgelegt werden.

Erstellung der IAM-Rolle für Hybridknoten

Um die Schritte in diesem Abschnitt auszuführen, muss der IAM-Principal, der die AWS Konsole oder AWS CLI verwendet, über die folgenden Berechtigungen verfügen.

- `iam:CreatePolicy`
- `iam:CreateRole`
- `iam:AttachRolePolicy`
- Wenn Sie AWS IAM Roles Anywhere verwenden
  - `rolesanywhere:CreateTrustAnchor`
  - `rolesanywhere:CreateProfile`
  - `iam:PassRole`

AWS CloudFormation

Installieren und konfigurieren Sie die AWS CLI, falls Sie dies noch nicht getan haben. Siehe Installation oder Aktualisierung auf die letzte Version der AWS CLI.

Schritte für AWS SSM-Hybrid-Aktivierungen

Der CloudFormation Stack erstellt die IAM-Rolle Hybrid Nodes mit den oben beschriebenen Berechtigungen. Die CloudFormation Vorlage erstellt die AWS SSM-Hybrid-Aktivierung nicht.

1. Laden Sie die AWS CloudFormation SSM-Vorlage für Hybridknoten herunter:

```
curl -OL 'https://raw.githubusercontent.com/aws/eks-hybrid/refs/heads/main/example/
hybrid-ssm-cfn.yaml'
```

2. Erstellen Sie einen `cfn-ssm-parameters.json` mit den folgenden Optionen:
   a. Ersetzen Sie `ROLE_NAME` durch den Namen Ihrer IAM-Rolle für Hybridknoten. Standardmäßig verwendet `AmazonEKSHybridNodesRole` die CloudFormation Vorlage den Namen der Rolle, die sie erstellt, wenn Sie keinen Namen angeben.
   b. `TAG_KEY`Ersetzen Sie es durch den AWS SSM-Ressourcen-Tag-Schlüssel, den Sie bei der Erstellung Ihrer AWS SSM-Hybrid-Aktivierung verwendet haben. Die Kombination aus Tag-Schlüssel und Tag-Wert wird in der Bedingung verwendet, dass nur die IAM-Rolle Hybrid Nodes die Registrierung der mit AWS SSM verwalteten Instanzen, die mit Ihrer SSM-Hybrid-Aktivierung verknüpft sind, aufheben darf. `ssm:DeregisterManagedInstance` AWS In der Vorlage ist der Standardwert CloudFormation . `TAG_KEY` `EKSClusterARN`
   c. `TAG_VALUE`Ersetzen Sie es durch den Wert des AWS SSM-Ressourcen-Tags, den Sie bei der Erstellung Ihrer AWS SSM-Hybrid-Aktivierung verwendet haben. Die Kombination aus Tag-Schlüssel und Tag-Wert wird in der Bedingung verwendet, dass nur die IAM-Rolle Hybrid Nodes die Registrierung der mit AWS SSM verwalteten Instanzen, die mit Ihrer SSM-Hybrid-Aktivierung verknüpft sind, aufheben darf. `ssm:DeregisterManagedInstance` AWS Wenn Sie den Standardwert `TAG_KEY` von `EKSClusterARN` verwenden, übergeben Sie Ihren EKS-Cluster-ARN als `TAG_VALUE`. EKS-Cluster haben das Format. ARNs `arn:aws:eks:AWS_REGION:AWS_ACCOUNT_ID:cluster/CLUSTER_NAME`

```
{
  "Parameters": {
    "RoleName": "ROLE_NAME",
    "SSMDeregisterConditionTagKey": "TAG_KEY",
    "SSMDeregisterConditionTagValue": "TAG_VALUE"
```

```
  }
}
```

3. Stellen Sie den CloudFormation Stack bereit. `STACK_NAME`Ersetzen Sie ihn durch Ihren Namen für den CloudFormation Stack.

```
aws cloudformation deploy \
    --stack-name STACK_NAME \
    --template-file hybrid-ssm-cfn.yaml \
    --parameter-overrides file://cfn-ssm-parameters.json \
    --capabilities CAPABILITY_NAMED_IAM
```

Schritte für AWS IAM Roles Anywhere

Der CloudFormation Stack erstellt den AWS IAM Roles Anywhere-Vertrauensanker mit der von Ihnen konfigurierten Zertifizierungsstelle (CA), erstellt das AWS IAM Roles Anywhere-Profil und erstellt die IAM-Rolle Hybrid Nodes mit den zuvor beschriebenen Berechtigungen.

1. So richten Sie eine Zertifizierungsstelle (CA) ein:
   a. Um eine AWS private CA-Ressource zu verwenden, öffnen Sie die AWS Private Certificate Authority-Konsole. Befolgen Sie die Anweisungen im Benutzerhandbuch der privaten AWS - Zertifizierungsstelle.
   b. Um eine externe Zertifizierungsstelle zu verwenden, befolgen Sie die Anweisungen der Zertifizierungsstelle. Die Zertifizierungsstelle geben Sie in einem späteren Schritt an.
   c. Öffentlich ausgestellte Zertifikate CAs können nicht als Vertrauensanker verwendet werden.
2. Laden Sie die AWS IAM Roles CloudFormation Anywhere-Vorlage für Hybridknoten herunter

```
curl -OL 'https://raw.githubusercontent.com/aws/eks-hybrid/refs/heads/main/example/
hybrid-ira-cfn.yaml'
```

3. Erstellen Sie einen `cfn-iamra-parameters.json` mit den folgenden Optionen:
   a. Ersetzen Sie `ROLE_NAME` durch den Namen Ihrer IAM-Rolle für Hybridknoten. Standardmäßig verwendet `AmazonEKSHybridNodesRole` die CloudFormation Vorlage den Namen der Rolle, die sie erstellt, wenn Sie keinen Namen angeben.
   b. Ersetzen Sie `CERT_ATTRIBUTE` durch das rechnerspezifische Zertifikatsattribut, das Ihren Host eindeutig identifiziert. Das von Ihnen verwendete Zertifikatsattribut muss mit dem nodeName übereinstimmen, den Sie für die `nodeadm`-Konfiguration verwenden, wenn Sie

Ihrem Cluster Hybridknoten zuordnen. Weitere Informationen hierzu finden Sie unter the section called “nodeadm für Hybridknoten”. Standardmäßig verwendet die CloudFormation Vorlage `${aws:PrincipalTag/x509Subject/CN}` as`CERT_ATTRIBUTE`, was dem CN-Feld Ihrer Computerzertifikate entspricht. Alternativ können Sie `$(aws:PrincipalTag/x509SAN/Name/CN}` als `CERT_ATTRIBUTE` übergeben.

c. Ersetzen Sie `CA_CERT_BODY` durch den Zertifikatstext Ihrer Zertifizierungsstelle ohne Zeilenumbrüche. Das `CA_CERT_BODY` müssen im Privacy Enhanced Mail (PEM)-Format vorliegen. Wenn Sie ein CA-Zertifikat im PEM-Format haben, entfernen Sie die Zeilenumbrüche und die Zeilen BEGIN CERTIFICATE und END CERTIFICATE, bevor Sie den CA-Zertifikatstext in Ihre `cfn-iamra-parameters.json`-Datei einfügen.

```
{
  "Parameters": {
    "RoleName": "ROLE_NAME",
    "CertAttributeTrustPolicy": "CERT_ATTRIBUTE",
    "CABundleCert": "CA_CERT_BODY"
  }
}
```

4. Stellen Sie die CloudFormation Vorlage bereit. `STACK_NAME`Ersetzen Sie es durch Ihren Namen für den CloudFormation Stack.

```
aws cloudformation deploy \
    --stack-name STACK_NAME \
    --template-file hybrid-ira-cfn.yaml \
    --parameter-overrides file://cfn-iamra-parameters.json
    --capabilities CAPABILITY_NAMED_IAM
```

AWS CLI

Installieren und konfigurieren Sie die AWS CLI, falls Sie dies noch nicht getan haben. Siehe Installation oder Aktualisierung auf die letzte Version der AWS CLI.

Erstellung einer EKS-Describe-Cluster-Richtlinie

1. Erstellen Sie eine Datei `eks-describe-cluster-policy.json` mit dem folgenden Inhalt:

```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks:DescribeCluster"
            ],
            "Resource": "*"
        }
    ]
}
```

2. Richtlinie mit dem folgenden Befehl erstellen:

```
aws iam create-policy \
    --policy-name EKSDescribeClusterPolicy \
    --policy-document file://eks-describe-cluster-policy.json
```

Schritte für AWS SSM-Hybrid-Aktivierungen

1. Erstellen Sie eine Datei mit dem Namen `eks-hybrid-ssm-policy.json` und dem folgenden Inhalt. Die Richtlinie gewährt die Berechtigung für zwei Aktionen `ssm:DescribeInstanceInformation` und `ssm:DeregisterManagedInstance`. Die Richtlinie beschränkt die `ssm:DeregisterManagedInstance` Zugriffsrechte auf AWS SSM-verwaltete Instanzen, die mit Ihrer AWS SSM-Hybrid-Aktivierung verknüpft sind, auf der Grundlage des Ressourcen-Tags, das Sie in Ihrer Vertrauensrichtlinie angeben.
   a. Ersetzen Sie es `AWS_REGION` durch die AWS Region für Ihre AWS SSM-Hybrid-Aktivierung.
   b. Ersetzen Sie `AWS_ACCOUNT_ID` durch Ihre AWS Konto-ID.
   c. `TAG_KEY`Ersetzen Sie es durch den AWS SSM-Ressourcen-Tag-Schlüssel, den Sie bei der Erstellung Ihrer AWS SSM-Hybrid-Aktivierung verwendet haben. Die Kombination aus Tag-Schlüssel und Tag-Wert wird in der Bedingung verwendet, dass nur die IAM-Rolle Hybrid Nodes die Registrierung der mit AWS SSM verwalteten Instanzen, die mit Ihrer SSM-Hybrid-Aktivierung verknüpft sind, aufheben darf. `ssm:DeregisterManagedInstance` AWS In der Vorlage ist der Standardwert CloudFormation . `TAG_KEY` `EKSClusterARN`
   d. `TAG_VALUE`Ersetzen Sie es durch den Wert des AWS SSM-Ressourcen-Tags, den Sie bei der Erstellung Ihrer AWS SSM-Hybrid-Aktivierung verwendet haben. Die Kombination aus Tag-Schlüssel und Tag-Wert wird in der Bedingung verwendet, dass nur die IAM-Rolle Hybrid Nodes die Registrierung der mit AWS SSM verwalteten Instanzen, die mit Ihrer SSM-Hybrid-Aktivierung verknüpft sind, aufheben darf. `ssm:DeregisterManagedInstance` AWS

Wenn Sie den Standardwert TAG_KEY von EKSClusterARN verwenden, übergeben Sie Ihren EKS-Cluster-ARN als TAG_VALUE. EKS-Cluster haben das Format. ARNs `arn:aws:eks:AWS_REGION:AWS_ACCOUNT_ID:cluster/CLUSTER_NAME`

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ssm:DescribeInstanceInformation",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ssm:DeregisterManagedInstance",
            "Resource": "arn:aws:ssm:us-east-1:123456789012:managed-instance/*",
            "Condition": {
                "StringEquals": {
                    "ssm:resourceTag/TAG_KEY": "TAG_VALUE"
                }
            }
        }
    ]
}
```

2. Richtlinie mit dem folgenden Befehl erstellen

```
aws iam create-policy \
    --policy-name EKSHybridSSMPolicy \
    --policy-document file://eks-hybrid-ssm-policy.json
```

3. Erstellen Sie eine Datei namens `eks-hybrid-ssm-trust.json`. AWS_REGIONErsetzen Sie es durch die AWS Region Ihrer AWS SSM-Hybrid-Aktivierung und AWS_ACCOUNT_ID durch Ihre AWS Konto-ID.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"",
         "Effect":"Allow",
         "Principal":{
```

```
            "Service":"ssm.amazonaws.com"
         },
         "Action":"sts:AssumeRole",
         "Condition":{
            "StringEquals":{
               "aws:SourceAccount":"123456789012"
            },
            "ArnEquals":{
               "aws:SourceArn":"arn:aws:ssm:us-east-1:123456789012:*"
            }
         }
      }
   ]
}
```

4. Erstellen Sie die Rolle mit dem folgenden Befehl.

```
aws iam create-role \
    --role-name AmazonEKSHybridNodesRole \
    --assume-role-policy-document file://eks-hybrid-ssm-trust.json
```

5. Fügen Sie `EKSDescribeClusterPolicy` und `EKSHybridSSMPolicy` an, die Sie in den vorherigen Schritten erstellt haben. Ersetzen Sie es `AWS_ACCOUNT_ID` durch Ihre AWS Konto-ID.

```
aws iam attach-role-policy \
    --role-name AmazonEKSHybridNodesRole \
    --policy-arn arn:aws: iam::AWS_ACCOUNT_ID:policy/EKSDescribeClusterPolicy
```

```
aws iam attach-role-policy \
    --role-name AmazonEKSHybridNodesRole \
    --policy-arn arn:aws: iam::AWS_ACCOUNT_ID:policy/EKSHybridSSMPolicy
```

6. Hängen Sie die `AmazonEC2ContainerRegistryPullOnly` und die `AmazonSSMManagedInstanceCore` AWS verwalteten Richtlinien an.

```
aws iam attach-role-policy \
    --role-name AmazonEKSHybridNodesRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEC2ContainerRegistryPullOnly
```

```
aws iam attach-role-policy \
    --role-name AmazonEKSHybridNodesRole \
```

```
    --policy-arn arn:aws: iam::aws:policy/AmazonSSMManagedInstanceCore
```

Schritte für AWS IAM Roles Anywhere

Um AWS IAM Roles Anywhere verwenden zu können, müssen Sie Ihren AWS IAM Roles Anywhere-Vertrauensanker einrichten, bevor Sie die IAM-Rolle Hybrid Nodes erstellen. Detaillierte Anweisungen finden Sie unter the section called “Richten Sie AWS IAM-Rollen überall ein”.

1. Erstellen Sie eine Datei namens `eks-hybrid-iamra-trust.json`. Ersetzen Sie `TRUST_ANCHOR ARN` durch die ARN des Trust Anchors, den Sie in den the section called “Richten Sie AWS IAM-Rollen überall ein”-Schritten erstellt haben. Die Bedingung in dieser Vertrauensrichtlinie schränkt die Fähigkeit von AWS IAM Roles Anywhere ein, die IAM-Rolle Hybrid Nodes zum Austausch temporärer IAM-Anmeldeinformationen nur dann anzunehmen, wenn der Name der Rollensitzung mit der CN im x509-Zertifikat übereinstimmt, das auf Ihren Hybridknoten installiert ist. Alternativ können Sie auch andere Zertifikatsattribute verwenden, um Ihren Knoten eindeutig zu identifizieren. Das Zertifikatattribut, das Sie in der Vertrauensrichtlinie verwenden, muss dem `nodeName` entsprechen, das Sie in Ihrer `nodeadm`-Konfiguration festgelegt haben. Weitere Informationen hierzu finden Sie unter the section called “nodeadm für Hybridknoten”.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "rolesanywhere.amazonaws.com"
            },
            "Action": [
                "sts:TagSession",
                "sts:SetSourceIdentity"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalTag/x509Subject/CN": "${aws:PrincipalTag/
x509Subject/CN}"
                },
                "ArnEquals": {
                    "aws:SourceArn": "arn:aws:rolesanywhere:us-
east-1:123456789012:trust-anchor/TA_ID"
```

```
                }
            }
        },
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "rolesanywhere.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "sts:RoleSessionName": "${aws:PrincipalTag/x509Subject/CN}",
                    "aws:PrincipalTag/x509Subject/CN": "${aws:PrincipalTag/
x509Subject/CN}"
                },
                "ArnEquals": {
                    "aws:SourceArn": "arn:aws:rolesanywhere:us-
east-1:123456789012:trust-anchor/TA_ID"
                }
            }
        }
    ]
}
```

2. Erstellen Sie die Rolle mit dem folgenden Befehl.

```
aws iam create-role \
    --role-name AmazonEKSHybridNodesRole \
    --assume-role-policy-document file://eks-hybrid-iamra-trust.json
```

3. Fügen Sie die `EKSDescribeClusterPolicy` an, die Sie in den vorherigen Schritten erstellt haben. Ersetzen Sie `AWS_ACCOUNT_ID` es durch Ihre Konto-ID. AWS

```
aws iam attach-role-policy \
    --role-name AmazonEKSHybridNodesRole \
    --policy-arn arn:aws: iam::AWS_ACCOUNT_ID:policy/EKSDescribeClusterPolicy
```

4. Hängen Sie die `AmazonEC2ContainerRegistryPullOnly` AWS verwaltete Richtlinie an

```
aws iam attach-role-policy \
    --role-name AmazonEKSHybridNodesRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEC2ContainerRegistryPullOnly
```

AWS-Managementkonsole

Erstellung einer EKS-Describe-Cluster-Richtlinie

1. Amazon-IAM-Konsole öffnen
2. Wählen Sie im linken Navigationsbereich die Option Policies aus.
3. Wählen Sie auf der Seite Richtlinien die Option Richtlinie erstellen.
4. Wählen Sie auf der Seite „Berechtigungen festlegen“ im Bereich „Service auswählen“ die Option „EKS“ aus.
   a. Filtern Sie Aktionen nach DescribeClusterund wählen Sie die Aktion DescribeClusterLesen aus.
   b. Wählen Sie Weiter aus.
5. Auf der Seite Überprüfen und erstellen
   a. Geben Sie einen Richtlinien-Namen ein, beispielsweise `EKSDescribeClusterPolicy`.
   b. Wählen Sie Richtlinie erstellen aus.

Schritte für AWS SSM-Hybrid-Aktivierungen

1. Amazon-IAM-Konsole öffnen
2. Wählen Sie im linken Navigationsbereich die Option Policies aus.
3. Wählen Sie auf der Seite Richtlinien die Option Richtlinie erstellen.
4. Wählen Sie auf der Seite Berechtigungen angeben in der Navigationsleiste oben rechts im Richtlinien-Editor die Option JSON aus. Fügen Sie den folgenden Ausschnitt ein. `AWS_REGION`Ersetzen Sie es durch die AWS Region Ihrer AWS SSM-Hybrid-Aktivierung und `AWS_ACCOUNT_ID` ersetzen Sie es durch Ihre AWS Konto-ID. Ersetzen Sie `TAG_KEY` und `TAG_VALUE` durch den AWS SSM-Ressourcen-Tag-Schlüssel, den Sie bei der Erstellung Ihrer AWS SSM-Hybrid-Aktivierung verwendet haben.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ssm:DescribeInstanceInformation",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
```

```
            "Action": "ssm:DeregisterManagedInstance",
            "Resource": "arn:aws:ssm:us-east-1:123456789012:managed-instance/*",
            "Condition": {
                "StringEquals": {
                    "ssm:resourceTag/TAG_KEY": "TAG_VALUE"
                }
            }
        }
    ]
}
```

a. Wählen Sie Weiter aus.

5. Auf der Seite Überprüfen und erstellen.

   a. Geben Sie einen Richtlinien-Namen für Ihre Richtlinie ein, beispielsweise `EKSHybridSSMPolicy`.

   b. Wählen Sie Richtlinie erstellen aus.

6. Wählen Sie im linken Navigationsbereich Roles aus.

7. Klicken Sie auf der Seite Roles (Rollen) auf Create role (Rolle erstellen).

8. Gehen Sie auf der Seite Select trusted entity (Vertrauenswürdige Entität auswählen) wie folgt vor:

   a. Wählen Sie unter Abschnittstyp Vertrauenswürdiger Entität die Option Benutzerdefinierte Vertrauensrichtlinie aus. Fügen Sie Folgendes in den Editor für benutzerdefinierte Vertrauensrichtlinien ein. `AWS_REGION`Ersetzen Sie es durch die AWS Region Ihrer AWS SSM-Hybrid-Aktivierung und durch Ihre `AWS_ACCOUNT_ID` AWS Konto-ID.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"",
         "Effect":"Allow",
         "Principal":{
            "Service":"ssm.amazonaws.com"
         },
         "Action":"sts:AssumeRole",
         "Condition":{
            "StringEquals":{
               "aws:SourceAccount":"123456789012"
            },
            "ArnEquals":{
```

```
                "aws:SourceArn":"arn:aws:ssm:us-east-1:123456789012:*"
            }
        }
    }
  ]
}
```

b. Wählen Sie Weiter aus.

9. Führen Sie auf der Seite Add permissions (Berechtigungen hinzufügen) die folgenden Schritte aus:

   a. Geben Sie im Feld Filterrichtlinien `EKSDescribeClusterPolicy` oder den Namen der oben erstellten Richtlinie ein. Aktivieren Sie in den Suchergebnissen das Kontrollkästchen links neben dem Namen Ihrer Richtlinie.

   b. Geben Sie im Feld Filterrichtlinien `EKSHybridSSMPolicy` oder den Namen der oben erstellten Richtlinie ein. Aktivieren Sie in den Suchergebnissen das Kontrollkästchen links neben dem Namen Ihrer Richtlinie.

   c. Geben Sie im Feld Filter policies (Filterrichtlinien) `AmazonEC2ContainerRegistryPullOnly` ein. Aktivieren Sie in den Suchergebnissen das Kontrollkästchen links neben `AmazonEC2ContainerRegistryPullOnly`.

   d. Geben Sie im Feld Filter policies (Filterrichtlinien) `AmazonSSMManagedInstanceCore` ein. Aktivieren Sie in den Suchergebnissen das Kontrollkästchen links neben `AmazonSSMManagedInstanceCore`.

   e. Wählen Sie Weiter aus.

10. Gehen Sie auf der Seite Name, review, and create (Benennen, überprüfen und erstellen) wie folgt vor:

    a. Geben Sie unter Role name (Rollenname) einen eindeutigen Namen für die Rolle ein, z. B. `AmazonEKSHybridNodesRole`.

    b. Ersetzen Sie unter Description (Beschreibung) den aktuellen Text durch beschreibenden Text wie beispielsweise `Amazon EKS - Hybrid Nodes role`.

    c. Wählen Sie Rolle erstellen aus.

Schritte für AWS IAM Roles Anywhere

Um AWS IAM Roles Anywhere verwenden zu können, müssen Sie Ihren AWS IAM Roles Anywhere-Vertrauensanker einrichten, bevor Sie die IAM-Rolle Hybrid Nodes erstellen. Detaillierte Anweisungen finden Sie unter the section called “Richten Sie AWS IAM-Rollen überall ein”.

1. Amazon-IAM-Konsole öffnen
2. Wählen Sie im linken Navigationsbereich Roles aus.
3. Klicken Sie auf der Seite Roles (Rollen) auf Create role (Rolle erstellen).
4. Gehen Sie auf der Seite Select trusted entity (Vertrauenswürdige Entität auswählen) wie folgt vor:
   a. Wählen Sie unter Abschnittstyp Vertrauenswürdiger Entität die Option Benutzerdefinierte Vertrauensrichtlinie aus. Fügen Sie Folgendes in den Editor für benutzerdefinierte Vertrauensrichtlinien ein. Ersetzen Sie `TRUST_ANCHOR ARN` durch die ARN des Trust Anchors, den Sie in den the section called “Richten Sie AWS IAM-Rollen überall ein”-Schritten erstellt haben. Die Bedingung in dieser Vertrauensrichtlinie schränkt die Fähigkeit von AWS IAM Roles Anywhere ein, die IAM-Rolle Hybrid Nodes zum Austausch temporärer IAM-Anmeldeinformationen nur dann anzunehmen, wenn der Name der Rollensitzung mit der CN im x509-Zertifikat übereinstimmt, das auf Ihren Hybridknoten installiert ist. Alternativ können Sie auch andere Zertifikatsattribute verwenden, um Ihren Knoten eindeutig zu identifizieren. Das Zertifikatattribut, das Sie in der Vertrauensrichtlinie verwenden, muss dem Knotennamen entsprechen, den Sie in Ihrer nodeadm-Konfiguration festgelegt haben. Weitere Informationen hierzu finden Sie unter the section called “nodeadm für Hybridknoten”.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "rolesanywhere.amazonaws.com"
            },
            "Action": [
                "sts:TagSession",
                "sts:SetSourceIdentity"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalTag/x509Subject/CN": "${aws:PrincipalTag/
x509Subject/CN}"
                },
                "ArnEquals": {
                    "aws:SourceArn": "arn:aws:rolesanywhere:us-
east-1:123456789012:trust-anchor/TA_ID"
                }
            }
```

```
        },
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "rolesanywhere.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "sts:RoleSessionName": "${aws:PrincipalTag/x509Subject/CN}",
                    "aws:PrincipalTag/x509Subject/CN": "${aws:PrincipalTag/
x509Subject/CN}"
                },
                "ArnEquals": {
                    "aws:SourceArn": "arn:aws:rolesanywhere:us-
east-1:123456789012:trust-anchor/TA_ID"
                }
            }
        }
    ]
}
```

b. Wählen Sie Weiter aus.

5. Führen Sie auf der Seite Add permissions (Berechtigungen hinzufügen) die folgenden Schritte aus:

   a. Geben Sie im Feld Filterrichtlinien `EKSDescribeClusterPolicy` oder den Namen der oben erstellten Richtlinie ein. Aktivieren Sie in den Suchergebnissen das Kontrollkästchen links neben dem Namen Ihrer Richtlinie.

   b. Geben Sie im Feld Filter policies (Filterrichtlinien) `AmazonEC2ContainerRegistryPullOnly` ein. Aktivieren Sie in den Suchergebnissen das Kontrollkästchen links neben `AmazonEC2ContainerRegistryPullOnly`.

   c. Wählen Sie Weiter aus.

6. Gehen Sie auf der Seite Name, review, and create (Benennen, überprüfen und erstellen) wie folgt vor:

   a. Geben Sie unter Role name (Rollenname) einen eindeutigen Namen für die Rolle ein, z. B. `AmazonEKSHybridNodesRole`.

   b. Ersetzen Sie unter Description (Beschreibung) den aktuellen Text durch beschreibenden Text wie beispielsweise `Amazon EKS - Hybrid Nodes role`.

   c. Wählen Sie Rolle erstellen aus.

Einen Amazon-EKS-Cluster mit Hybridknoten erstellen

Dieses Thema bietet einen Überblick über die verfügbaren Optionen und beschreibt, was bei der Erstellung eines Hybridknoten-fähigen Amazon-EKS-Clusters zu beachten ist. EKS-Hybridknoten bieten denselben Support für Kubernetes-Versionen wie Amazon-EKS-Cluster mit Cloud-Knoten, einschließlich Standard- und erweiterten Support

Wenn Sie nicht vorhaben, EKS-Hybridknoten zu verwenden, lesen Sie die primäre Amazon-EKS-Dokumentation zum Erstellen von Clustern unter the section called “Erstellen eines Clusters”.

Voraussetzungen

- the section called “Voraussetzungen” wurde abgeschlossen. Bevor Sie Ihren Cluster mit aktivierten Hybridknoten erstellen, müssen Sie Ihren lokalen Knoten und optional Ihren Pod CIDRs identifizieren, Ihre VPC und Subnetze gemäß den EKS-Anforderungen und den Anforderungen der Hybridknoten erstellt haben und Ihre Sicherheitsgruppe mit eingehenden Regeln für Ihren lokalen und optionalen Pod. CIDRs Weitere Informationen zu diesen Voraussetzungen finden Sie unter the section called “Netzwerk vorbereiten”.
- Die neueste Version der AWS Befehlszeilenschnittstelle (AWS CLI) ist auf Ihrem Gerät installiert und konfiguriert. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version`. Paketmanager wie yum, apt-get oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der CLI. AWS Informationen zur Installation der neuesten Version finden Sie unter Installation oder Aktualisierung auf die letzte Version der AWS CLI und Konfigurieren von Einstellungen für die AWS CLI im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle.
- Ein IAM-Prinzipal mit Berechtigungen zum Erstellen von IAM-Rollen und Anfügen von Richtlinien sowie zum Erstellen und Beschreiben von EKS-Clustern

Überlegungen

- Ihr Cluster muss entweder `API` oder `API_AND_CONFIG_MAP` für den Cluster-Authentifizierungsmodus verwenden.
- Ihr Cluster muss die IPv4 Adressfamilie verwenden.
- Ihr Cluster muss entweder öffentliche oder private Cluster-Endpunkt-Konnektivität verwenden. Ihr Cluster kann die Cluster-Endpunktkonnektivität „öffentlich und privat“ nicht verwenden, da der Amazon EKS Kubernetes API-Serverendpunkt IPs für Hybridknoten, die außerhalb Ihrer VPC laufen, öffentlich auflöst.
- Die OIDC-Authentifizierung wird für EKS-Cluster mit Hybridknoten unterstützt.

- Sie können die Hybridknoten-Konfiguration eines vorhandenen Clusters hinzufügen, ändern oder entfernen. Weitere Informationen finden Sie unter the section called “Vorhandener Cluster”.

Schritt 1: Cluster-IAM-Rolle erstellen

Wenn Sie bereits eine Cluster-IAM-Rolle haben oder Ihren Cluster mit `eksctl` oder erstellen möchten AWS CloudFormation, können Sie diesen Schritt überspringen. Standardmäßig erstellt die AWS CloudFormation Vorlage die Cluster-IAM-Rolle für Sie. `eksctl`

1. Führen Sie den folgenden Befehl aus, um eine JSON-Datei für eine IAM-Vertrauensrichtlinie zu erstellen.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Erstellen Sie die Amazon-EKS-Cluster-IAM-Rolle. Geben Sie bei Bedarf als Präfix eks-cluster-role-trust -policy.json den Pfad auf Ihrem Computer ein, in den Sie die Datei im vorherigen Schritt geschrieben haben. Der Befehl verknüpft die im vorherigen Schritt erstellte Vertrauensrichtlinie mit der Rolle. Um eine IAM-Rolle zu erstellen, muss dem IAM-Prinzipal, der die Rolle erstellt, die `iam:CreateRole`-Aktion (Berechtigung) zugewiesen werden.

```
aws iam create-role \
    --role-name myAmazonEKSClusterRole \
    --assume-role-policy-document file://"eks-cluster-role-trust-policy.json"
```

3. Sie können entweder die von Amazon EKS verwaltete Richtlinie zuweisen oder Ihre eigene benutzerdefinierte Richtlinie erstellen. Informationen zu den Mindestberechtigungen, die Sie in Ihrer benutzerdefinierten Richtlinie verwenden müssen, finden Sie unter the section called “Knoten-IAM-Rolle”. Hängen Sie die von Amazon EKS verwaltete Richtlinie `AmazonEKSClusterPolicy` an die Rolle an. Um eine IAM-Richtlinie an einen IAM-

Prinzipal anzuhängen, muss der Prinzipal, der die Richtlinie anhängt, eine der folgenden IAM-Aktionen (Berechtigungen) zugewiesen werden: `iam:AttachUserPolicy` oder `iam:AttachRolePolicy`.

```
aws iam attach-role-policy \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSClusterPolicy \
    --role-name myAmazonEKSClusterRole
```

Schritt 2: Cluster mit Hybridknoten erstellen

Sie können einen Cluster erstellen, indem Sie Folgendes verwenden:

- eksctl
- AWS CloudFormation
- AWS CLI
- AWS-Managementkonsole

Cluster mit Hybridknoten erstellen – eksctl

Sie müssen die neueste Version des `eksctl`-Befehlszeilentools installieren. Informationen zum Installieren und Aktualisieren von `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.

1. Erstellen`cluster-config.yaml`, um einen Amazon IPv4 EKS-Cluster mit Hybridknoten zu definieren. Nehmen Sie die folgenden Ersetzungen in Ihrem `cluster-config.yaml` vor. Eine vollständige Liste der Einstellungen finden Sie in der eksctl-Dokumentation.
   a. Ersetzen Sie `CLUSTER_NAME` durch Ihren Cluster-Namen. Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein.
   b. `AWS_REGION`Ersetzen Sie es durch die AWS Region, in der Sie Ihren Cluster erstellen möchten.
   c. Ersetzen Sie `K8S_VERSION` durch eine von Amazon EKS unterstützte Version.
   d. Ersetzen Sie `CREDS_PROVIDER` durch `ssm` oder `ira`, je nachdem, welchen Anmeldeinformationsanbieter Sie in den Schritten für the section called “Anmeldeinformationen vorbereiten” konfiguriert haben.

e. Ersetzen SieCA_BUNDLE_CERT, wenn Ihr Anmeldeinformationsanbieter auf eingestellt istira, der AWS IAM Roles Anywhere als Anmeldeinformationsanbieter verwendet. CA_BUNDLE_CERT ist die Zertifizierungsstelle (CA) und hängt von Ihrer Wahl der CA ab. Das Zertifikat muss im PEM-Format (Privacy Enhanced Mail) vorliegen.

f. Ersetzen Sie GATEWAY_ID durch die ID Ihres Virtual Private Gateways oder Transit-Gateways, das an Ihre VPC angefügt werden soll.

g. Ersetzen Sie REMOTE_NODE_CIDRS durch den On-Premises-Knoten-CIDR für Ihre Hybridknoten.

h. Ersetzen Sie REMOTE_POD_CIDRS durch das On-Premises-Pod-CIDR für Workloads, die in Hybridknoten ausgeführt werden, oder entfernen Sie die Zeile aus Ihrer Konfiguration, wenn Sie keine Webhooks in Hybridknoten ausführen. Sie müssen Ihre REMOTE_POD_CIDRS konfigurieren, wenn Ihr CNI keine Network Address Translation (NAT) oder Maskierung für Pod-IP-Adressen verwendet, wenn Pod-Datenverkehr Ihre On-Premises-Hosts verlässt. Sie müssen REMOTE_POD_CIDRS konfigurieren, wenn Sie Webhooks auf Hybridknoten ausführen. Weitere Informationen finden Sie unter the section called “Webhooks konfigurieren”.

i. Ihre On-Premises-Knoten- und Pod-CIDR-Blöcke müssen die folgenden Anforderungen erfüllen:

   i. Innerhalb eines der IPv4 RFC-1918-Bereiche liegen:,, oder. 10.0.0.0/8 172.16.0.0/12 192.168.0.0/16

   ii. Überschneiden Sie sich nicht miteinander, das CIDR VPC CIDR für Ihren Cluster oder Ihren Kubernetes-Service IPv4

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: CLUSTER_NAME
  region: AWS_REGION
  version: "K8S_VERSION"

remoteNetworkConfig:
  iam:
    provider: CREDS_PROVIDER # default SSM, can also be set to IRA
    # caBundleCert: CA_BUNDLE_CERT
  vpcGatewayID: GATEWAY_ID
  remoteNodeNetworks:
  - cidrs: ["REMOTE_NODE_CIDRS"]
  remotePodNetworks:
```

```
  - cidrs: ["REMOTE_POD_CIDRS"]
```

2. Führen Sie den folgenden Befehl aus:

```
eksctl create cluster -f cluster-config.yaml
```

Die Clusterbereitstellung dauert mehrere Minuten. Während der Cluster erstellt wird, werden mehrere Ausgabezeilen angezeigt. Die letzte Ausgabezeile ähnelt der folgenden Beispielzeile.

```
[#]  EKS cluster "CLUSTER_NAME" in "REGION" region is ready
```

3. Fahren Sie fort mit the section called “Schritt 3: kubeconfig aktualisieren”.

Erstellen Sie einen Cluster mit aktivierten Hybridknoten - AWS CloudFormation

Der CloudFormation Stack erstellt die IAM-Rolle des EKS-Clusters und einen EKS-Cluster mit dem von Ihnen angegebenen `RemoteNodeNetwork` und`RemotePodNetwork`. Ändern Sie die CloudFormation Vorlage, wenn Sie Einstellungen für Ihren EKS-Cluster anpassen müssen, die nicht in der CloudFormation Vorlage verfügbar gemacht werden.

1. Laden Sie die CloudFormation Vorlage herunter.

```
curl -OL 'https://raw.githubusercontent.com/aws/eks-hybrid/refs/heads/main/example/
hybrid-eks-cfn.yaml'
```

2. Erstellen Sie einen `cfn-eks-parameters.json` und geben Sie für jeden Wert Ihre Konfiguration an.
   a. `CLUSTER_NAME`: Name des zu erstellenden EKS-Clusters
   b. `CLUSTER_ROLE_NAME`: Name der zu erstellenden IAM-Rolle des EKS-Clusters. Die Standardeinstellung in der Vorlage ist „EKSClusterRolle“.
   c. `SUBNET1_ID`: die ID des ersten Subnetzes, das Sie in den erforderlichen Schritten erstellt haben
   d. `SUBNET2_ID`: die ID des zweiten Subnetzes, das Sie in den erforderlichen Schritten erstellt haben
   e. `SG_ID`: die Sicherheitsgruppen-ID, die Sie in den erforderlichen Schritten erstellt haben.
   f. `REMOTE_NODE_CIDRS`: das On-Premises-Knoten-CIDR für Ihre Hybridknoten

g. `REMOTE_POD_CIDRS`: das On-Premises-Pod-CIDR für Workloads, die in Hybridknoten ausgeführt werden. Sie müssen Ihre `REMOTE_POD_CIDRS` konfigurieren, wenn Ihr CNI keine Network Address Translation (NAT) oder Maskierung für Pod-IP-Adressen verwendet, wenn Pod-Datenverkehr Ihre On-Premises-Hosts verlässt. Sie müssen `REMOTE_POD_CIDRS` konfigurieren, wenn Sie Webhooks auf Hybridknoten ausführen. Weitere Informationen finden Sie unter the section called “Webhooks konfigurieren”.

h. Ihre On-Premises-Knoten- und Pod-CIDR-Blöcke müssen die folgenden Anforderungen erfüllen:

   i. Innerhalb eines der IPv4 RFC-1918-Bereiche liegen:`10.0.0.0/8`,`172.16.0.0/12`, oder. `192.168.0.0/16`

   ii. Überschneiden Sie sich nicht miteinander, weder mit dem CIDR `VPC  CIDR` für Ihren Cluster noch mit Ihrem Kubernetes-Service. IPv4

i. `CLUSTER_AUTH`: der Cluster-Authentifizierungsmodus für Ihren Cluster. Gültige Werte sind `API` und `API_AND_CONFIG_MAP`. Der Standardwert in der Vorlage ist `API_AND_CONFIG_MAP`.

j. `CLUSTER_ENDPOINT`: die Cluster-Endpunkt-Konnektivität für Ihren Cluster. Gültige Werte sind „Öffentlich“ und „Privat“. Die Standardeinstellung in der Vorlage ist „Privat“, was bedeutet, dass Sie nur von Ihrem VPC aus eine Verbindung zum Kubernetes-API-Endpunkt herstellen können.

k. `K8S_VERSION`: die für Ihren Cluster zu verwendende Kubernetes-Version. Siehe die Von Amazon EKS unterstützten Versionen.

```
{
  "Parameters": {
    "ClusterName": "CLUSTER_NAME",
    "ClusterRoleName": "CLUSTER_ROLE_NAME",
    "SubnetId1": "SUBNET1_ID",
    "SubnetId2": "SUBNET2_ID",
    "SecurityGroupId" "SG_ID",
    "RemoteNodeCIDR": "REMOTE_NODE_CIDRS",
    "RemotePodCIDR": "REMOTE_POD_CIDRS",
    "ClusterAuthMode": "CLUSTER_AUTH",
    "ClusterEndpointConnectivity": "CLUSTER_ENDPOINT",
    "K8sVersion": "K8S_VERSION"
  }
 }
```

3. Stellen Sie den Stack bereit. CloudFormation `STACK_NAME`Ersetzen Sie ihn durch Ihren Namen für den CloudFormation Stack und `AWS_REGION` durch Ihre AWS Region, in der der Cluster erstellt werden soll.

```
aws cloudformation deploy \
    --stack-name STACK_NAME \
    --region AWS_REGION \
    --template-file hybrid-eks-cfn.yaml \
    --parameter-overrides file://cfn-eks-parameters.json \
    --capabilities CAPABILITY_NAMED_IAM
```

Die Clusterbereitstellung dauert mehrere Minuten. Sie können den Status Ihres Stacks mit dem folgenden Befehl überprüfen. `STACK_NAME`Ersetzen Sie es durch Ihren Namen für den CloudFormation Stack und `AWS_REGION` durch Ihre AWS Region, in der der Cluster erstellt werden soll.

```
aws cloudformation describe-stacks \
    --stack-name STACK_NAME \
    --region AWS_REGION \
    --query 'Stacks[].StackStatus'
```

4. Fahren Sie fort mit the section called “Schritt 3: kubeconfig aktualisieren”.

Cluster mit Hybridknoten erstellen — CLI AWS

1. Führen Sie den folgenden Befehl aus, um einen EKS-Cluster mit aktivierten Hybridknoten zu erstellen. Ersetzen Sie vor dem Ausführen des Befehls Folgendes durch Ihre Einstellungen. Eine vollständige Liste der Einstellungen finden Sie in der the section called “Erstellen eines Clusters”-Dokumentation.
   a. `CLUSTER_NAME`: Name des zu erstellenden EKS-Clusters
   b. `AWS_REGION`: AWS Region, in der der Cluster erstellt wird.
   c. `K8S_VERSION`: die für Ihren Cluster zu verwendende Kubernetes-Version. Siehe die von Amazon EKS unterstützten Versionen.
   d. `ROLE_ARN`: die Amazon-EKS-Cluster-Rolle, die Sie für Ihren Cluster konfiguriert haben. Weitere Informationen finden Sie unter IAM-Rolle des Amazon-EKS-Clusters.
   e. `SUBNET1_ID`: die ID des ersten Subnetzes, das Sie in den erforderlichen Schritten erstellt haben
   f. `SUBNET2_ID`: die ID des zweiten Subnetzes, das Sie in den erforderlichen Schritten erstellt haben
   g. `SG_ID`: die Sicherheitsgruppen-ID, die Sie in den erforderlichen Schritten erstellt haben.

h. Sie können `API` und `API_AND_CONFIG_MAP` für Ihren Cluster-Zugriffs-Authentifizierungsmodus verwenden. Im folgenden Befehl ist der Cluster-Zugriffs-Authentifizierungsmodus auf `API_AND_CONFIG_MAP` eingestellt.

i. Sie können die `endpointPublicAccess`- und `endpointPrivateAccess`-Parameter verwenden, um den öffentlichen und privaten Zugriff auf den Kubernetes-API-Server-Endpunkt Ihres Clusters zu aktivieren oder zu deaktivieren. Im folgenden Befehl wird `endpointPublicAccess` auf „false“ und `endpointPrivateAccess` auf „true“ gesetzt.

j. `REMOTE_NODE_CIDRS`: das On-Premises-Knoten-CIDR für Ihre Hybridknoten.

k. `REMOTE_POD_CIDRS` (optional): das On-Premises-Pod-CIDR für Workloads, die in Hybridknoten ausgeführt werden.

l. Ihre On-Premises-Knoten- und Pod-CIDR-Blöcke müssen die folgenden Anforderungen erfüllen:

   i. Sie müssen sich innerhalb eines der IPv4 RFC-1918-Bereiche befinden:`10.0.0.0/8`,`172.16.0.0/12`, oder. `192.168.0.0/16`

   ii. Keine Überschneidungen miteinander, weder `VPC CIDR` für Ihren Amazon EKS-Cluster noch für Ihren Kubernetes-Service IPv4 CIDR.

```
aws eks create-cluster \
    --name CLUSTER_NAME \
    --region AWS_REGION \
    --kubernetes-version K8S_VERSION \
    --role-arn ROLE_ARN \
    --resources-vpc-config
 subnetIds=SUBNET1_ID,SUBNET2_ID,securityGroupIds=SG_ID,endpointPrivateAccess=true,endpo
 \
    --access-config authenticationMode=API_AND_CONFIG_MAP \
    --remote-network-config '{"remoteNodeNetworks":[{"cidrs":
["REMOTE_NODE_CIDRS"]}],"remotePodNetworks":[{"cidrs":["REMOTE_POD_CIDRS"]}]}'
```

2. Die Bereitstellung des Clusters dauert mehrere Minuten. Sie können den Status Ihres Clusters mit dem folgenden Befehl überprüfen. `CLUSTER_NAME`Ersetzen Sie ihn durch den Namen des Clusters, den Sie erstellen, und `AWS_REGION` durch die AWS Region, in der der Cluster erstellt wird. Fahren Sie nicht mit dem nächsten Schritt fort, bis die zurückgegebene Ausgabe `ACTIVE` ist.

```
aws eks describe-cluster \
    --name CLUSTER_NAME \
    --region AWS_REGION \
    --query "cluster.status"
```

3. Fahren Sie fort mit the section called “Schritt 3: kubeconfig aktualisieren”.

Erstellen Sie einen Cluster mit aktivierten Hybridknoten - AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole unter Amazon-EKS-Konsole.
2. Wählen Sie Cluster hinzufügen und dann Erstellen aus.
3. Füllen Sie auf der Seite Configure cluster (Cluster konfigurieren) die folgenden Felder aus:
   a. Name – Ein Name für Ihren Cluster. Der Name darf nur alphanumerische Zeichen (Groß-/Kleinschreibung beachten), Bindestriche und Unterstriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein.
   b. Cluster-IAM-Rolle — Wählen Sie die Amazon EKS-Cluster-IAM-Rolle aus, die Sie erstellt haben, damit die Kubernetes-Steuerebene AWS Ressourcen in Ihrem Namen verwalten kann.
   c. Kubernetes-Version – Die Kubernetes-Version, die für den Cluster verwendet wird. Wir empfehlen, die frühere Version auszuwählen, es sei denn, Sie benötigen eine ältere Version.
   d. Upgrade-Richtlinie – Wählen Sie entweder „Erweitert“ oder „Standard“.
      i. Erweitert: Diese Option unterstützt die Kubernetes-Version 26 Monate nach dem Veröffentlichungsdatum. Für den erweiterten Support-Zeitraum fallen zusätzliche Stundenkosten an, die nach Ablauf des Standard-Support-Zeitraums beginnen. Nach Ablauf des erweiterten Supports wird Ihr Cluster automatisch auf die nächste Version aktualisiert.
      ii. Standard: Diese Option unterstützt die Kubernetes-Version 14 Monate nach dem Veröffentlichungsdatum. Es entstehen keine zusätzlichen Kosten. Wenn der Standard-Support endet, wird Ihr Cluster automatisch auf die nächste Version aktualisiert.
   e. Cluster-Zugriff – Wählen Sie, ob Sie den Zugriff für Cluster-Administratoren zulassen oder verweigern möchten, und wählen Sie einen Authentifizierungsmodus aus Die folgenden Authentifizierungs-Modi werden für Cluster mit Hybridknoten unterstützt.
      i. EKS-API: Der Cluster bezieht authentifizierte IAM-Prinzipale nur aus dem EKS-Zugriffseintrag. APIs
      ii. EKS-API und ConfigMap: Der Cluster bezieht authentifizierte IAM-Prinzipale sowohl aus dem EKS-Zugriffseintrag als auch aus dem. APIs `aws-auth` ConfigMap
   f. Secrets-Verschlüsselung – (Optional) Aktivieren Sie die Secrets-Verschlüsselung von Kubernetes-Secrets mithilfe eines KMS-Schlüssels. Sie können diese auch aktivieren, nachdem

Sie Ihren Cluster erstellt haben. Stellen Sie vor Aktivierung dieser Funktion sicher, dass Sie mit den Informationen unter the section called “Geheime Verschlüsselung aktivieren” vertraut sind.

g. ARC-Zonenverschiebung – Wenn aktiviert, registriert EKS Ihren Cluster mit ARC-Zonenverschiebung, damit Sie mithilfe der Zonenverschiebung den Anwendungsverkehr von einer AZ weg verschieben können.

h. Tags – (Optional) Fügen Sie Ihrem Cluster beliebige Tags hinzu. Weitere Informationen finden Sie unter the section called “Markieren Ihrer -Ressourcen”.

i. Wenn Sie mit dieser Seite fertig sind, wählen Sie Weiter aus.

4. Wählen Sie auf der Seite Specify networking (Netzwerk angeben) die Werte für die folgenden Felder aus:

a. VPC – Wählen Sie eine vorhandene VPC, welche die Anforderungen von the section called “VPC- und Subnetz-Anforderungen” und Amazon EKS Hybrid Nodes erfüllt. Bevor Sie sich für eine VPC entscheiden, empfehlen wir Ihnen, sich mit allen Anforderungen und Überlegungen in „Amazon-EKS-Netzwerkanforderungen für VPC, Subnetze und Hybridknoten anzeigen“ vertraut zu machen. Sie können nach der Cluster-Erstellung nicht mehr ändern, welche VPC Sie verwenden möchten. Wenn keine aufgeführt VPCs sind, müssen Sie zuerst eine erstellen. Weitere Informationen finden Sie unter the section called “Erstellen einer VPC” und den Netzwerkanforderungen für Amazon EKS Hybrid Nodes.

b. Subnetze – Standardmäßig sind alle im vorherigen Feld angegebenen Subnetze in der VPC vorausgewählt. Sie müssen mindestens zwei auswählen.

c. Security groups (Sicherheitsgruppen) – (Optional) Geben Sie eine oder mehrere Sicherheitsgruppen an, die Amazon EKS den erstellten Netzwerkschnittstellen zuordnen soll. Für mindestens eine der von Ihnen angegebenen Sicherheitsgruppen müssen eingehende Regeln für Ihren lokalen Knoten und optional für den Pod gelten. CIDRs Weitere Informationen finden Sie in den Netzwerkanforderungen für Amazon EKS Hybrid Nodes. Unabhängig davon, ob Sie Sicherheitsgruppen wählen oder nicht, erstellt Amazon EKS eine Sicherheitsgruppe, die die Kommunikation zwischen Ihrem Cluster und Ihrer VPC ermöglicht. Amazon EKS verknüpft diese Sicherheitsgruppe und alle, die Sie wählen, mit den erstellten Netzwerkschnittstellen. Weitere Informationen zur Cluster-Sicherheitsgruppe, die von Amazon EKS erstellt wird, finden Sie unterthe section called “Anforderungen an Sicherheitsgruppen”. Sie können die Regeln in der von Amazon EKS erstellten Cluster-Sicherheitsgruppe ändern.

d. Cluster-IP-Adressfamilie wählen — Sie müssen sich IPv4 für Cluster mit aktivierten Hybridknoten entscheiden.

e. (Optional) Wählen Sie „Kubernetes Service IP-Adressbereich konfigurieren“ und geben Sie einen Dienstbereich an. IPv4

f. Wählen Sie Remotenetzwerke konfigurieren, um Hybridknoten zu aktivieren, und geben Sie Ihren lokalen Knoten und Pod CIDRs für Hybridknoten an.

g. Sie müssen die CIDR Ihres Pods aus der Ferne konfigurieren, wenn Ihr CNI keine Network Address Translation (NAT) oder Maskierung für Pod-IP-Adressen verwendet, wenn der Pod-Datenverkehr Ihre On-Premises-Hosts verlässt. Sie müssen das Pod-CIDR aus der Ferne konfigurieren, wenn Sie Webhooks in Hybridknoten ausführen.

h. Ihre On-Premises-Knoten- und Pod-CIDR-Blöcke müssen die folgenden Anforderungen erfüllen:

   i. Sie müssen sich innerhalb eines der IPv4 RFC-1918-Bereiche befinden:`10.0.0.0/8`,, `172.16.0.0/12` oder. `192.168.0.0/16`

   ii. Überschneiden Sie sich nicht miteinander, das CIDR `VPC CIDR` für Ihren Cluster oder Ihren Kubernetes-Service IPv4

i. Wählen Sie eine Option für den Cluster-Endpunktzugriff aus. Nachdem Ihr Cluster erstellt wurde, können Sie diese Option ändern. Für Cluster mit aktivierten Hybridknoten müssen Sie entweder „Öffentlich“ oder „Privat“ auswählen. Bevor Sie eine nicht standardmäßige Option auswählen, sollten Sie sich mit den Optionen und deren Auswirkungen vertraut machen. Weitere Informationen finden Sie unter the section called “Cluster-Endpunktzugriff”.

j. Wenn Sie mit dieser Seite fertig sind, wählen Sie Weiter aus.

5. (Optional) Auf der Seite Beobachtbarkeit konfigurieren können Sie Metriken und Optionen zur Steuerebenen-Protokollierung aktivieren. Standardmäßig sind alle Protokollierungstypen deaktiviert.

   a. Weitere Informationen zur Prometheus-Metrik-Option finden Sie unter the section called “Prometheus-Metriken”.

   b. Weitere Informationen zu den Protokollierungsoptionen der EKS-Steuerung finden Sie unter the section called “Protokolle der Steuerebene”.

   c. Wenn Sie mit dieser Seite fertig sind, wählen Sie Weiter aus.

6. Wählen Sie auf der Seite Select add-ons (Add-Ons auswählen) die Add-Ons aus, die Sie Ihrem Cluster hinzufügen möchten.

   a. Sie können so viele Amazon EKS-Add-Ons und AWS Marketplace-Add-Ons auswählen, wie Sie benötigen. Amazon-EKS-Add-Ons, die nicht mit Hybridknoten kompatibel sind, sind mit „Nicht kompatibel mit Hybridknoten“ gekennzeichnet und verfügen über eine Anti-Affinitätsregel, um zu verhindern, dass sie auf Hybridknoten ausgeführt werden. Weitere Informationen finden Sie unter „Konfiguration von Add-Ons für Hybridknoten“. Wenn die AWS Marketplace-Add-Ons, die Sie installieren möchten, nicht aufgeführt sind, können Sie nach verfügbaren AWS Marketplace-Add-Ons suchen, indem Sie Text in das Suchfeld eingeben. Sie können auch nach

category (Kategorie), vendor (Anbieter) oder pricing model (Preismodell) suchen und dann die Add-Ons aus den Suchergebnissen auswählen.

b. Einige Add-Ons, wie CoreDNS und Kube-Proxy, sind standardmäßig installiert. Wenn Sie eines der Standard-Add-Ons deaktivieren, kann dies Ihre Fähigkeit beeinträchtigen, Kubernetes-Anwendungen auszuführen.

c. Wenn Sie mit dieser Seite fertig sind, wählen Sie `Next`.

7. Wählen Sie auf der Seite Einstellungen für ausgewählte Add-ons konfigurieren die Version aus, die Sie installieren möchten.

   a. Sie können nach der Clustererstellung jederzeit auf eine neuere Version aktualisieren. Sie können die Konfiguration jedes Add-Ons nach der Cluster-Erstellung aktualisieren. Weitere Informationen zum Konfigurieren eines Add-Ons finden Sie unter the section called “Add-On aktualisieren”. Die mit Hybridknoten kompatiblen Add-Ons-Versionen finden Sie unter the section called “Add-Ons konfigurieren”.

   b. Wenn Sie mit dieser Seite fertig sind, wählen Sie Weiter aus.

8. Überprüfen Sie auf der Seite Review and create (Überprüfen und erstellen) die Informationen, die Sie auf den vorherigen Seiten eingegeben oder ausgewählt haben. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Wenn Sie zufrieden sind, klicken Sie auf Erstellen. Das Feld Status zeigt CREATING (WIRD ERSTELLT) an, während der Cluster bereitgestellt wird. Die Clusterbereitstellung dauert mehrere Minuten.

9. Fahren Sie fort mit the section called “Schritt 3: kubeconfig aktualisieren”.

Schritt 3: kubeconfig aktualisieren

Wenn Sie Ihren Cluster mit `eksctl` erstellt haben, können Sie diesen Schritt überspringen. Das liegt daran, dass `eksctl` diesen Schritt bereits für Sie durchgeführt hat. Aktivieren Sie `kubectl`, um mit Ihrem Cluster zu kommunizieren, indem Sie einen neuen Kontext zur `kubectl`-config-Datei hinzufügen. Weitere Informationen zum Erstellen und Aktualisieren der Datei finden Sie unter the section called “Zugriff auf den Cluster mit kubectl”.

```
aws eks update-kubeconfig --name CLUSTER_NAME --region AWS_REGION
```

Eine Beispielausgabe sieht wie folgt aus.

```
Added new context arn:aws: eks:AWS_REGION:111122223333:cluster/CLUSTER_NAME to /home/
username/.kube/config
```

Bestätigen Sie die Kommunikation mit Ihrem Cluster, indem Sie den folgenden Befehl ausführen.

```
kubectl get svc
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   28h
```

Schritt 4: Cluster-Einrichtung

Sehen Sie sich als nächsten Schritt the section called “Cluster-Zugriff vorbereiten” an, um den Zugriff für Ihre Hybridknoten zu aktivieren, damit diese Ihrem Cluster beitreten können.

Hybridknoten in einem vorhandenen Amazon-EKS-Cluster aktivieren oder die Konfiguration ändern

Dieses Thema bietet einen Überblick über die verfügbaren Optionen und beschreibt, was zu beachten ist, wenn Sie die Hybridknoten-Konfiguration für einen Amazon-EKS-Cluster hinzufügen, ändern oder entfernen.

Um einem Amazon-EKS-Cluster die Verwendung von Hybridknoten zu ermöglichen, fügen Sie der `RemoteNetworkConfig`-Konfiguration die CIDR-Bereiche der IP-Adressen Ihres On-Premises-Knotens und optional des Pod-Netzwerks hinzu. EKS verwendet diese CIDR-Liste, um die Konnektivität zwischen dem Cluster und Ihren On-Premises-Netzwerken zu ermöglichen. Eine vollständige Liste der Optionen zum Aktualisieren Ihrer Cluster-Konfiguration finden Sie unter UpdateClusterConfig in der Amazon-EKS-API-Referenz.

Sie können für die Netzwerkkonfiguration der EKS-Hybridknoten in einem Cluster die folgenden Aktionen ausführen:

- Fügen Sie eine Fern-Netzwerkkonfiguration hinzu, um EKS-Hybridknoten in einem vorhandenen Cluster zu aktivieren.
- Fügen Sie die Fern-Knotennetzwerke oder die Pod-Netzwerke aus der Ferne in einem vorhandenen Cluster hinzu, ändern oder entfernen Sie sie.
- Entfernen Sie alle CIDR-Bereiche des Knotennetzwerks aus der Ferne, um EKS-Hybridknoten in einem vorhandenen Cluster zu deaktivieren.

Voraussetzungen

- Bevor Sie Ihren Amazon-EKS-Cluster für Hybridknoten aktivieren, stellen Sie sicher, dass Ihre Umgebung die unter the section called “Voraussetzungen” beschriebenen und unter the section called “Netzwerk vorbereiten”, the section called “Betriebssystem vorbereiten” und the section called “Anmeldeinformationen vorbereiten” ausführlich beschriebenen Anforderungen erfüllt.
- Ihr Cluster muss die IPv4-Adressfamilie verwenden.
- Ihr Cluster muss entweder `API` oder `API_AND_CONFIG_MAP` für den Cluster-Authentifizierungsmodus verwenden. Der Vorgang zum Ändern des Cluster-Authentifizierungsmodus wird unter the section called “Authentifizierungsmodus” beschrieben.
- Wir empfehlen, entweder den öffentlichen oder den privaten Endpunktzugriff für den API-Server-Endpunkt für Amazon EKS Kubernetes zu verwenden, jedoch nicht beide gleichzeitig. Wenn Sie „Öffentlich und Privat“ auswählen, wird der API-Serverendpunkt für Amazon EKS Kubernetes immer auf die öffentlichen IP-Adressen für Hybridknoten aufgelöst, die außerhalb Ihrer VPC ausgeführt werden. Dies kann dazu führen, dass Ihre Hybridknoten nicht dem Cluster beitreten können. Der Vorgang zum Ändern des Netzwerkzugriffs in Ihren Cluster wird unter the section called “Cluster-Endpunktzugriff” beschrieben.
- Die neueste Version der AWS-Befehlszeilenschnittstelle (AWS-CLI) muss auf Ihrem Gerät installiert und konfiguriert sein. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version`. Paket-Manager wie yum, apt-get oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der AWS-CLI zurück. Um die neueste Version zu installieren, lesen Sie Installation oder Aktualisierung auf die letzte Version der AWS-CLI und Konfiguration der Einstellungen für die AWS-CLI im Benutzerhandbuch zur AWS-Befehlszeilenschnittstelle.
- Ein IAM-Prinzipal mit der Berechtigung, UpdateClusterConfig in Ihrem Amazon-EKS-Cluster aufzurufen.
- Aktualisieren Sie Add-Ons in Versionen, die mit Hybridknoten kompatibel sind. Die mit Hybridknoten kompatiblen Add-On-Versionen finden Sie unter the section called “Add-Ons konfigurieren”.
- Wenn Sie Add-Ons ausführen, die nicht mit Hybridknoten kompatibel sind, stellen Sie sicher, dass das Add-On DaemonSet oder Bereitstellung über die folgende Affinitätsregel verfügt, um die Bereitstellung auf Hybridknoten zu verhindern. Fügen Sie die folgende Affinitätsregel hinzu, falls sie noch nicht vorhanden ist.

```
affinity:
  nodeAffinity:
```

```
    requiredDuringSchedulingIgnoredDuringExecution:
      nodeSelectorTerms:
      - matchExpressions:
        - key: eks.amazonaws.com/compute-type
          operator: NotIn
          values:
          - hybrid
```

Überlegungen

Das `remoteNetworkConfig`-JSON-Objekt weist während einer Aktualisierung folgendes Verhalten auf:

- Alle vorhandenen Teile der Konfiguration, die Sie nicht angeben, bleiben unverändert. Wenn Sie weder `remoteNodeNetworks` noch `remotePodNetworks` angeben, bleibt dieser Teil unverändert.
- Wenn Sie die CIDR-Liste `remoteNodeNetworks` oder `remotePodNetworks` ändern, müssen Sie die vollständige Liste der CIDRs angeben, die Sie in Ihrer endgültigen Konfiguration haben möchten. Wenn Sie eine Änderung an der CIDR-Liste `remoteNodeNetworks` oder `remotePodNetworks` angeben, ersetzt EKS während der Aktualisierung die ursprüngliche Liste.
- Ihre On-Premises-Knoten- und Pod-CIDR-Blöcke müssen die folgenden Anforderungen erfüllen:
  1. Sie müssen sich innerhalb eines der IPv4-RFC-1918-Bereiche befinden: 10.0.0.0/8, 172.16.0.0/12 oder 192.168.0.0/16.
  2. Alle CIDRs der VPC für Ihren Amazon-EKS-Cluster oder Ihr Kubernetes-Service-IPv4-CIDR dürfen sich nicht überschneiden.

Aktivierung von Hybridknoten in einem vorhandenen Cluster

Sie können EKS-Hybridknoten in einem vorhandenen Cluster aktivieren, indem Sie Folgendes verwenden:

- AWS CloudFormation
- AWS CLI
- AWS-Managementkonsole

Hybridknoten in einem vorhandenen Cluster aktivieren – AWS CloudFormation

1. Um EKS-Hybridknoten in Ihrem Cluster zu aktivieren, fügen Sie `RemoteNodeNetwork` und `RemotePodNetwork` (optional) zu Ihrer CloudFormation-Vorlage hinzu und aktualisieren Sie den Stack. Beachten Sie, dass `RemoteNodeNetwork` eine Liste mit maximal einem `Cidrs`-Element ist und `Cidrs` eine Liste mehrerer IP-CIDR-Bereiche ist.

```
RemoteNetworkConfig:
  RemoteNodeNetworks:
    - Cidrs: [RemoteNodeCIDR]
  RemotePodNetworks:
    - Cidrs: [RemotePodCIDR]
```

2. Fahren Sie fort mit the section called “Cluster-Zugriff vorbereiten”.

Hybridknoten in einem vorhandenen Cluster aktivieren – AWS-CLI

1. Führen Sie den folgenden Befehl aus, um `RemoteNetworkConfig` für EKS-Hybridknoten für Ihren EKS-Cluster zu aktivieren. Ersetzen Sie vor dem Ausführen des Befehls Folgendes durch Ihre Einstellungen. Eine vollständige Liste der Einstellungen finden Sie unter UpdateClusterConfig in der Amazon-EKS-API-Referenz.
   a. `CLUSTER_NAME`: Name des zu aktualisierenden EKS-Clusters.
   b. `AWS_REGION`: AWS-Region, in welcher der EKS-Cluster ausgeführt wird.
   c. `REMOTE_NODE_CIDRS`: das On-Premises-Knoten-CIDR für Ihre Hybridknoten.
   d. `REMOTE_POD_CIDRS` (optional): das On-Premises-Pod-CIDR für Workloads, die in Hybridknoten ausgeführt werden.

```
aws eks update-cluster-config \
    --name CLUSTER_NAME \
    --region AWS_REGION \
    --remote-network-config '{"remoteNodeNetworks":[{"cidrs":
["REMOTE_NODE_CIDRS"]}],"remotePodNetworks":[{"cidrs":["REMOTE_POD_CIDRS"]}]}'
```

2. Die Aktualisierung des Clusters dauert mehrere Minuten. Sie können den Status Ihres Clusters mit dem folgenden Befehl überprüfen. Ersetzen Sie `CLUSTER_NAME` durch den Namen des Clusters, den Sie ändern, und `AWS_REGION` durch die AWS-Region, in welcher der Cluster ausgeführt wird. Fahren Sie nicht mit dem nächsten Schritt fort, bis die zurückgegebene Ausgabe `ACTIVE` ist.

```
aws eks describe-cluster \
    --name CLUSTER_NAME \
    --region AWS_REGION \
    --query "cluster.status"
```

3. Fahren Sie fort mit the section called “Cluster-Zugriff vorbereiten”.

Hybridknoten in einem vorhandenen Cluster aktivieren – AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole unter Amazon-EKS-Konsole.
2. Wählen Sie den Namen des Clusters aus, um Ihre Cluster-Informationen anzuzeigen.
3. Wählen Sie die Registerkarte Netzwerk und dann Verwalten aus.
4. Wählen Sie in der Dropdownliste Fern-Netzwerke aus.
5. Wählen Sie „Fern-Netzwerke konfigurieren“, um Hybridknoten zu aktivieren, und geben Sie Ihre On-Premises-Knoten- und Pod-CIDRs für Hybridknoten an.
6. Wählen Sie Änderungen speichern, um den Vorgang abzuschließen. Warten Sie, bis der Cluster-Status wieder auf Aktiv zurückgesetzt ist.
7. Fahren Sie fort mit the section called “Cluster-Zugriff vorbereiten”.

Konfiguration der Hybridknoten in einem vorhandenen Cluster aktualisieren

Sie können `remoteNetworkConfig` in einem vorhandenen Hybrid-Cluster mithilfe einer der folgenden Optionen ändern:

- AWS CloudFormation
- AWS CLI
- AWS-Managementkonsole

Hybrid-Konfiguration in einem vorhandenen Cluster aktualisieren – AWS CloudFormation

1. Aktualisieren Sie Ihre CloudFormation-Vorlage mit den neuen Netzwerk-CIDR-Werten.

```
RemoteNetworkConfig:
  RemoteNodeNetworks:
    - Cidrs: [NEW_REMOTE_NODE_CIDRS]
```

```
  RemotePodNetworks:
    - Cidrs: [NEW_REMOTE_POD_CIDRS]
```

> **Note**
>
> Berücksichtigen Sie beim Aktualisieren der CIDR-Listen `RemoteNodeNetworks` oder `RemotePodNetworks` alle CIDRs (neue und vorhandene). EKS ersetzt bei Aktualisierungen die gesamte Liste. Durch das Weglassen dieser Felder aus der Aktualisierungsanforderung werden ihre vorhandenen Konfigurationen beibehalten.

2. Aktualisieren Sie Ihren CloudFormation-Stack mit der geänderten Vorlage und warten Sie, bis die Stack-Aktualisierung abgeschlossen ist.

Hybrid-Konfiguration in einem vorhandenen Cluster aktualisieren – AWS CLI

1. Um die CIDRs des Fern-Netzwerks zu ändern, führen Sie den folgenden Befehl aus. Ersetzen Sie die Werte durch Ihre Einstellungen:

```
aws eks update-cluster-config
--name CLUSTER_NAME
--region AWS_REGION
--remote-network-config '{"remoteNodeNetworks":[{"cidrs":
["NEW_REMOTE_NODE_CIDRS"]}],"remotePodNetworks":[{"cidrs":
["NEW_REMOTE_POD_CIDRS"]}]}'
```

> **Note**
>
> Berücksichtigen Sie beim Aktualisieren der CIDR-Listen `remoteNodeNetworks` oder `remotePodNetworks` alle CIDRs (neue und vorhandene). EKS ersetzt bei Aktualisierungen die gesamte Liste. Durch das Weglassen dieser Felder aus der Aktualisierungsanforderung werden ihre vorhandenen Konfigurationen beibehalten.

2. Warten Sie, bis der Cluster-Status wieder auf „AKTIV“ zurückkehrt, bevor Sie fortfahren.

Hybrid-Konfiguration in einem vorhandenen Cluster aktualisieren – AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole unter Amazon-EKS-Konsole.
2. Wählen Sie den Namen des Clusters aus, um Ihre Cluster-Informationen anzuzeigen.

3. Wählen Sie die Registerkarte Netzwerk und dann Verwalten aus.
4. Wählen Sie in der Dropdownliste Fern-Netzwerke aus.
5. Aktualisieren Sie die CIDRs unter `Remote node networks` und `Remote pod networks - Optional` nach Bedarf.
6. Wählen Sie Änderungen speichern und warten Sie, bis der Cluster-Status wieder auf Aktiv wechselt.

Hybridknoten in einem vorhandenen Cluster deaktivieren

Sie können EKS-Hybridknoten in einem vorhandenen Cluster deaktivieren, indem Sie Folgendes verwenden:

- AWS CloudFormation
- AWS CLI
- AWS-Managementkonsole

EKS-Hybridknoten in einem vorhandenen Cluster deaktivieren – AWS CloudFormation

1. Um EKS-Hybridknoten in Ihrem Cluster zu deaktivieren, setzen Sie `RemoteNodeNetworks` und `RemotePodNetworks` in Ihrer CloudFormation-Vorlage auf leere Arrays und aktualisieren Sie den Stack.

```
RemoteNetworkConfig:
  RemoteNodeNetworks: []
  RemotePodNetworks: []
```

EKS-Hybridknoten in einem vorhandenen Cluster deaktivieren – AWS-CLI

1. Führen Sie den folgenden Befehl aus, um `RemoteNetworkConfig` aus Ihrem EKS-Cluster zu entfernen. Ersetzen Sie vor dem Ausführen des Befehls Folgendes durch Ihre Einstellungen. Eine vollständige Liste der Einstellungen finden Sie unter UpdateClusterConfig in der Amazon-EKS-API-Referenz.
   a. `CLUSTER_NAME`: Name des zu aktualisierenden EKS-Clusters.
   b. `AWS_REGION`: AWS-Region, in welcher der EKS-Cluster ausgeführt wird.

```
aws eks update-cluster-config \
```

```
    --name CLUSTER_NAME \
    --region AWS_REGION \
    --remote-network-config '{"remoteNodeNetworks":[],"remotePodNetworks":[]}'
```

2. Die Aktualisierung des Clusters dauert mehrere Minuten. Sie können den Status Ihres Clusters mit dem folgenden Befehl überprüfen. Ersetzen Sie `CLUSTER_NAME` durch den Namen des Clusters, den Sie ändern, und `AWS_REGION` durch die AWS-Region, in welcher der Cluster ausgeführt wird. Fahren Sie nicht mit dem nächsten Schritt fort, bis die zurückgegebene Ausgabe `ACTIVE` ist.

```
aws eks describe-cluster \
    --name CLUSTER_NAME \
    --region AWS_REGION \
    --query "cluster.status"
```

EKS-Hybridknoten in einem vorhandenen Cluster deaktivieren – AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole unter Amazon-EKS-Konsole.
2. Wählen Sie den Namen des Clusters aus, um Ihre Cluster-Informationen anzuzeigen.
3. Wählen Sie die Registerkarte Netzwerk und dann Verwalten aus.
4. Wählen Sie in der Dropdownliste Fern-Netzwerke aus.
5. Wählen Sie Fern-Netzwerke zur Aktivierung von Hybridknoten konfigurieren und alle CIDRs unter `Remote node networks` und `Remote pod networks - Optional` zu entfernen.
6. Wählen Sie Änderungen speichern, um den Vorgang abzuschließen. Warten Sie, bis der Cluster-Status wieder auf Aktiv zurückgesetzt ist.

Vorbereitung des Cluster-Zugriffs für Hybridknoten

Bevor Sie Hybridknoten mit Ihrem Amazon-EKS-Cluster verbinden, müssen Sie Ihre Hybridknoten-IAM-Rolle mit Kubernetes-Berechtigungen aktivieren, um dem Cluster beitreten zu können. Informationen zum Erstellen der IAM-Rolle Hybridknoten finden Sie unter the section called “Anmeldeinformationen vorbereiten”. Amazon EKS unterstützt zwei Möglichkeiten, IAM-Prinzipale mit Kubernetes Role-Based Access Control (RBAC), Amazon EKS-Zugriffseinträgen und dem zu verknüpfen. `aws-auth` ConfigMap Weitere Informationen zur Amazon-EKS-Zugriffsverwaltung finden Sie unter the section called “Kubernetes-API-Zugriff”.

Verwenden Sie die folgenden Verfahren, um Ihre IAM-Rolle für Hybridknoten den Kubernetes-Berechtigungen zuzuordnen. Um Amazon-EKS-Zugriffseinträge verwenden zu können, muss Ihr

Cluster mit den Authentifizierungsmodi `API` oder `API_AND_CONFIG_MAP` erstellt worden sein. Um den verwenden zu können `aws-auth` ConfigMap, muss Ihr Cluster im Authentifizierungsmodus erstellt worden sein. `API_AND_CONFIG_MAP` Der `CONFIG_MAP`-eigene-Authentifizierungsmodus wird für Amazon-EKS-Cluster mit aktivierten Hybridknoten nicht unterstützt.

Verwendung von Amazon-EKS-Zugriffseinträgen für die IAM-Rolle für Hybridknoten

Es gibt einen Amazon-EKS-Zugriffseintragstyp für Hybridknoten mit dem Namen HYBRID_LINUX, der mit einer IAM-Rolle verwendet werden kann. Bei diesem Zugriffseintragstyp wird der Benutzername automatisch auf system:node: {{SessionName}} gesetzt. Weitere Informationen zum Erstellen von Zugriffseinträgen finden Sie unter the section called “Zugriffseinträge erstellen”.

AWS CLI

1. Sie müssen die neueste Version der AWS CLI auf Ihrem Gerät installiert und konfiguriert haben. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version`. Paketmanager wie yum, apt-get oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der CLI. AWS Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle.
2. Erstellen Sie Ihren Zugriffseintrag mit dem folgenden Befehl. Ersetzen Sie CLUSTER_NAME durch den Namen Ihres Clusters und HYBRID_NODES_ROLE_ARN durch die ARN der Rolle, die Sie in den Schritten für the section called “Anmeldeinformationen vorbereiten” erstellt haben.

```
aws eks create-access-entry --cluster-name CLUSTER_NAME \
    --principal-arn HYBRID_NODES_ROLE_ARN \
    --type HYBRID_LINUX
```

AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole unter Amazon-EKS-Konsole.
2. Wählen Sie den Namen Ihres Clusters mit aktivierten Hybridknoten.
3. Wählen Sie die Registerkarte Zugriff aus.
4. Wählen Sie Zugriffseintrag erstellen aus.
5. Wählen Sie für IAM-Prinzipal die IAM-Rolle für Hybridknoten aus, die Sie in den Schritten für the section called “Anmeldeinformationen vorbereiten” erstellt haben.
6. Wählen Sie für Typ die Option Hybrid Linux aus.

7. (Optional) Weisen Sie dem Zugriffseintrag unter Tags Beschriftungen zu. So können Sie beispielsweise ganz einfach nach allen Ressourcen mit dem gleichen Tag suchen.
8. Wählen Sie „Überspringen“, um die Überprüfung und Erstellung fortzusetzen. Sie können dem Hybrid-Linux-Zugriffseintrag keine Richtlinien hinzufügen oder seinen Zugriffsbereich ändern.
9. Überprüfen Sie die Konfiguration für Ihren Zugriffseintrag. Sollten Sie einen Fehler entdecken, wählen Sie Zurück aus, um schrittweise zurück zu navigieren und den Fehler zu korrigieren. Ist die Konfiguration korrekt, wählen Sie Erstellen aus.

Verwendung von aws-auth ConfigMap für die IAM-Rolle Hybrid Nodes

In den folgenden Schritten erstellen oder aktualisieren Sie die IAM-Rolle `aws-auth` ConfigMap mit dem ARN der Hybrid Nodes IAM-Rolle, die Sie in den Schritten für the section called “Anmeldeinformationen vorbereiten” erstellt haben.

1. Prüfen Sie, ob `aws-auth` ConfigMap für Ihren Cluster bereits eine vorhanden ist. Beachten Sie, dass Sie bei Verwendung einer bestimmten `kubeconfig`-Datei das `--kubeconfig`-Flag verwenden sollten.

```
kubectl describe configmap -n kube-system aws-auth
```

2. Wenn Ihnen eine angezeigt wird `aws-auth` ConfigMap, aktualisieren Sie sie nach Bedarf.
   a. Öffnen Sie das ConfigMap zur Bearbeitung.

```
kubectl edit -n kube-system configmap/aws-auth
```

   b. Fügen Sie nach Bedarf einen neuen `mapRoles`-Eintrag hinzu. Ersetzen Sie `HYBRID_NODES_ROLE_ARN` durch die ARN Ihrer IAM-Rolle für Hybridknoten. Hinweis, `{{SessionName}}` ist das richtige Vorlagenformat zum Speichern in ConfigMap. Ersetzen Sie es nicht durch andere Werte.

```
data:
  mapRoles: |
  - groups:
    - system:bootstrappers
    - system:nodes
    rolearn: HYBRID_NODES_ROLE_ARN
    username: system:node:{{SessionName}}
```

   c. Speichern Sie die Datei und beenden Sie den Text-Editor.

3. Wenn `aws-auth` ConfigMap für Ihren Cluster noch kein vorhandenes vorhanden ist, erstellen Sie es mit dem folgenden Befehl. Ersetzen Sie `HYBRID_NODES_ROLE_ARN` durch die ARN Ihrer IAM-Rolle für Hybridknoten. Beachten Sie, `{{SessionName}}` dass dies das richtige Vorlagenformat zum Speichern in ist ConfigMap. Ersetzen Sie es nicht durch andere Werte.

```
kubectl apply -f=/dev/stdin <<-EOF
apiVersion: v1
kind: ConfigMap
metadata:
  name: aws-auth
  namespace: kube-system
data:
  mapRoles: |
  - groups:
    - system:bootstrappers
    - system:nodes
    rolearn: HYBRID_NODES_ROLE_ARN
    username: system:node:{{SessionName}}
EOF
```

Ausführung von On-Premises-Workloads in Hybridknoten

In einem EKS-Cluster mit aktivierten EKS-Hybridknoten können Sie On-Premises- und Edge-Anwendungen auf Ihrer eigenen Infrastruktur mit denselben Amazon-EKS-Clustern, Features und Tools ausführen, die Sie in der AWS Cloud verwenden.

Die folgenden Abschnitte enthalten schrittweise Anleitungen zur Verwendung von Hybridknoten.

Themen

- Hybridknoten verbinden
- Hybridknoten mit Bottlerocket verbinden
- Aktualisierung von Hybridknoten für Ihren Cluster
- Patch-Sicherheitsaktualisierungen für Hybridknoten
- Hybridknoten entfernen

Hybridknoten verbinden

> **Note**
>
> Die folgenden Schritte gelten für Hybridknoten mit kompatiblen Betriebssystemen außer Bottlerocket. Anweisungen zum Verbinden eines Hybridknotens mit Bottlerocket finden Sie unter the section called “Hybridknoten mit Bottlerocket verbinden”.

In diesem Thema wird beschrieben, wie Hybridknoten mit einem Amazon-EKS-Cluster verbunden werden. Nachdem Ihre Hybridknoten dem Cluster beigetreten sind, werden sie in der Amazon-EKS-Konsole und in Kubernetes-kompatiblen Tools wie kubectl mit dem Status „Nicht bereit“ angezeigt. Nachdem Sie die Schritte auf dieser Seite abgeschlossen haben, fahren Sie mit the section called “CNI konfigurieren” fort, um Ihre Hybridknoten für die Ausführung von Anwendungen vorzubereiten.

Voraussetzungen

Bevor Sie Hybridknoten mit Ihrem Amazon-EKS-Cluster verbinden, stellen Sie sicher, dass Sie die erforderlichen Schritte abgeschlossen haben.

- Sie verfügen über eine Netzwerkverbindung von Ihrer On-Premises-Umgebung zu der AWS-Region, in der Ihr Amazon-EKS-Cluster gehostet wird. Weitere Informationen finden Sie unter the section called “Netzwerk vorbereiten”.
- Sie verfügen über ein kompatibles Betriebssystem für Hybridknoten, das auf Ihren On-Premises-Hosts installiert ist. Weitere Informationen finden Sie unter the section called “Betriebssystem vorbereiten”.
- Sie haben Ihre IAM-Rolle für Hybridknoten erstellt und Ihren On-Premises-Anmeldeinformationsanbieter (AWS-Systems-Manager-Hybridaktivierungen oder AWS IAM Roles Anywhere) eingerichtet. Weitere Informationen finden Sie unter the section called “Anmeldeinformationen vorbereiten”.
- Sie haben Ihren für Hybridknoten aktivierten Amazon-EKS-Cluster erstellt. Weitere Informationen finden Sie unter the section called “Cluster erstellen”.
- Sie haben Ihre Hybridknoten-IAM-Rolle den Berechtigungen der rollenbasierten Zugriffssteuerung (RBAC) von Kubernetes zugeordnet. Weitere Informationen finden Sie unter the section called “Cluster-Zugriff vorbereiten”.

Schritt 1: Hybridknoten-CLI (**`nodeadm`**) auf jedem On-Premises-Host installieren

Wenn Sie die Amazon-EKS-Hybrid-Nodes-CLI (`nodeadm`) in Ihre vorgefertigten Betriebssystem-Images integrieren, können Sie diesen Schritt überspringen. Weitere Informationen zur Hybridknotenversion von `nodeadm` finden Sie unter the section called “nodeadm für Hybridknoten”.

Die Hybridknoten-Version von `nodeadm` wird in Amazon S3 gehostet und von Amazon CloudFront bereitgestellt. Um die Installation auf jedem On-Premises-Host `nodeadm` durchzuführen, können Sie den folgenden Befehl von Ihren On-Premises-Hosts aus ausführen.

Für x86_64 hosts:

```
curl -OL 'https://hybrid-assets.eks.amazonaws.com/releases/latest/bin/linux/amd64/
nodeadm'
```

Für ARM-Hosts

```
curl -OL 'https://hybrid-assets.eks.amazonaws.com/releases/latest/bin/linux/arm64/
nodeadm'
```

Fügen Sie der heruntergeladenen Binärdatei auf jedem Host die Berechtigung zum Ausführen von Dateien hinzu.

```
chmod +x nodeadm
```

Schritt 2: Die Abhängigkeiten der Hybridknoten mit **`nodeadm`** installieren

Wenn Sie die Abhängigkeiten der Hybridknoten in vorgefertigten Betriebssystem-Images installieren, können Sie diesen Schritt überspringen. Mit dem `nodeadm install`-Befehl können Sie alle für Hybridknoten erforderlichen Abhängigkeiten installieren. Die Abhängigkeiten der Hybridknoten beinhalten Containerd, kubelet, kubectl und AWS-SSM- oder AWS-IAM-Roles-Anywhere-Komponenten. Weitere Informationen zu den von the section called “nodeadm für Hybridknoten” installierten Komponenten und Dateispeicherorten finden Sie unter `nodeadm install`. Weitere Informationen zu den Domains, die in Ihrer On-Premises-Firewall für den `nodeadm install`-Prozess zugelassen werden müssen, finden Sie unter the section called “Netzwerk vorbereiten” für Hybridknoten.

Führen Sie den folgenden Befehl aus, um die Abhängigkeiten der Hybridknoten auf Ihrem On-Premises-Host zu installieren. Der nachfolgende Befehl muss von einem Benutzer ausgeführt werden, der über sudo-/root-Zugriff auf Ihren Host verfügt.

> **Important**
>
> Die Hybridknoten-CLI (`nodeadm`) muss mit einem Benutzer ausgeführt werden, der über sudo/root-Zugriff auf Ihren Host verfügt.

- Ersetzen Sie `K8S_VERSION` durch die Kubernetes-Nebenversion Ihres Amazon-EKS-Clusters, z. B. `1.31`. Eine Liste der unterstützten Kubernetes-Versionen finden Sie unter Unterstützte Versionen von Amazon EKS.
- Ersetzen Sie `CREDS_PROVIDER` durch den von Ihnen verwendeten On-Premises-Anmeldeinformationsanbieter. Gültige Werte sind `ssm` für AWS SSM und `iam-ra` für AWS IAM Roles Anywhere.

```
nodeadm install K8S_VERSION --credential-provider CREDS_PROVIDER
```

Schritt 3: Hybridknoten mit Ihrem Cluster verbinden

Bevor Sie Ihre Hybridknoten mit Ihrem Cluster verbinden, stellen Sie sicher, dass Sie in Ihrer On-Premises-Firewall und in der Sicherheitsgruppe für Ihren Cluster den erforderlichen Zugriff für die Kommunikation zwischen der Amazon-EKS-Steuerebene und dem Hybridknoten zugelassen haben. Die meisten Probleme in diesem Schritt stehen im Zusammenhang mit der Firewall-Konfiguration, der Konfiguration der Sicherheitsgruppe oder der Konfiguration der IAM-Rolle für Hybridknoten.

> **Important**
>
> Die Hybridknoten-CLI (`nodeadm`) muss mit einem Benutzer ausgeführt werden, der über sudo/root-Zugriff auf Ihren Host verfügt.

1. Erstellen Sie auf jedem Host eine `nodeConfig.yaml`-Datei mit den Werten für Ihre Bereitstellung. Eine vollständige Beschreibung der verfügbaren Konfigurationseinstellungen finden Sie unter the section called “nodeadm für Hybridknoten”. Wenn Ihre IAM-Rolle für Hybridknoten über keine Berechtigung für die `eks:DescribeCluster`-Aktion verfügt, müssen Sie Ihren Kubernetes-API-Endpunkt, Ihr Cluster-CA-Bundle und Ihre Kubernetes-Service-IPv4-CIDR im Cluster-Abschnitt Ihres `nodeConfig.yaml` übergeben.

a. Verwenden Sie das nachfolgende `nodeConfig.yaml`-Beispiel, wenn Sie AWS-SSM-Hybridaktivierungen für Ihren On-Premises-Anmeldeinformationsanbieter verwenden.

   i. Ersetzen Sie `CLUSTER_NAME` mit dem Namen Ihres Clusters.

   ii. Ersetzen Sie `AWS_REGION` mit der AWS-Region, in der Ihr Cluster gehostet wird. Beispiel, `us-west-2`.

   iii. Ersetzen Sie `ACTIVATION_CODE` durch den Aktivierungscode, den Sie beim Erstellen Ihrer AWS-SSM-Hybridaktivierung erhalten haben. Weitere Informationen finden Sie unter the section called “Anmeldeinformationen vorbereiten”.

   iv. Ersetzen Sie `ACTIVATION_ID` durch die Aktivierungs-ID, die Sie beim Erstellen Ihrer AWS-SSM-Hybridaktivierung erhalten haben. Sie können diese Informationen von der AWS-Systems-Manager-Konsole oder über den `aws ssm describe-activations`-Befehl der AWS-CLI abrufen.

```
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name: CLUSTER_NAME
    region: AWS_REGION
  hybrid:
    ssm:
      activationCode: ACTIVATION_CODE
      activationId: ACTIVATION_ID
```

b. Verwenden Sie das nachfolgende `nodeConfig.yaml`-Beispiel, wenn Sie AWS IAM Roles Anywhere für Ihren On-Premises-Anmeldeinformationsanbieter verwenden.

   i. Ersetzen Sie `CLUSTER_NAME` mit dem Namen Ihres Clusters.

   ii. Ersetzen Sie `AWS_REGION` mit der AWS-Region, in der Ihr Cluster gehostet wird. Beispiel, `us-west-2`.

   iii. Ersetzen Sie `NODE_NAME` durch den Namen Ihres Knotens. Der Knotenname muss mit dem CN des Zertifikats auf dem Host übereinstimmen, wenn Sie die Vertrauensrichtlinie Ihrer IAM-Rolle für Hybridknoten mit der `"sts:RoleSessionName": "${aws:PrincipalTag/x509Subject/CN}"`-Ressourcenbedingung konfiguriert haben. Der von Ihnen verwendete `nodeName` darf nicht länger als 64 Zeichen sein.

   iv. Ersetzen Sie `TRUST_ANCHOR_ARN` durch die ARN des Trust Anchors, den Sie in den Schritten zum Vorbereiten der Anmeldeinformationen für Hybridknoten konfiguriert haben.

v. Ersetzen Sie `PROFILE_ARN` durch die ARN des Trust Anchors, den Sie in den Schritten für the section called “Anmeldeinformationen vorbereiten” konfiguriert haben.

vi. Ersetzen Sie `ROLE_ARN` durch die ARN Ihrer IAM-Rolle für Hybridknoten.

vii. Ersetzen Sie `CERTIFICATE_PATH` durch den Pfad auf Ihrer Festplatte zu Ihrem Knotenzertifikat. Wenn Sie keinen Pfad angeben, wird standardmäßig `/etc/iam/pki/server.pem` verwendet.

viii. Ersetzen Sie `KEY_PATH` durch den Pfad auf Ihrer Festplatte zu Ihrem privaten Zertifikatsschlüssel. Wenn Sie keinen Pfad angeben, wird standardmäßig `/etc/iam/pki/server.key` verwendet.

```
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name: CLUSTER_NAME
    region: AWS_REGION
  hybrid:
    iamRolesAnywhere:
      nodeName: NODE_NAME
      trustAnchorArn: TRUST_ANCHOR_ARN
      profileArn: PROFILE_ARN
      roleArn: ROLE_ARN
      certificatePath: CERTIFICATE_PATH
      privateKeyPath: KEY_PATH
```

2. Führen Sie den `nodeadm init`-Befehl mit Ihrem `nodeConfig.yaml` aus, um Ihre Hybridknoten mit Ihrem Amazon-EKS-Cluster zu verbinden.

```
nodeadm init -c file://nodeConfig.yaml
```

Wenn der obige Befehl erfolgreich abgeschlossen wurde, ist Ihr Hybridknoten Ihrem Amazon-EKS-Cluster beigetreten. Sie können dies in der Amazon-EKS-Konsole überprüfen, indem Sie zur Registerkarte „Datenverarbeitung“ für Ihren Cluster navigieren (stellen Sie sicher, dass der IAM-Prinzipal über Anzeigeberechtigungen verfügt) oder mit `kubectl get nodes`.

> **Important**
>
> Ihre Knoten haben den Status `Not Ready`, was zu erwarten ist und darauf zurückzuführen ist, dass auf Ihren Hybridknoten kein CNI ausgeführt wird. Wenn Ihre Knoten nicht dem Cluster beigetreten sind, sehen Sie unter the section called “Fehlerbehebung” nach.

Schritt 4: CNI für Hybridknoten konfigurieren

Um Ihre Hybridknoten für die Ausführung von Anwendungen vorzubereiten, führen Sie die Schritte in the section called “CNI konfigurieren” aus.

Hybridknoten mit Bottlerocket verbinden

In diesem Thema wird beschrieben, wie Sie Hybridknoten, auf denen Bottlerocket ausgeführt wird, mit einem Amazon-EKS-Cluster verbinden. Bottlerocket ist eine Open-Source-Linux-Distribution, die von gesponsert und unterstützt wird. AWS Bottlerocket wurde speziell für das Hosting von Container-Workloads entwickelt. Mit Bottlerocket können Sie die Verfügbarkeit von containerisierten Bereitstellungen verbessern und die Betriebskosten senken, indem Sie Aktualisierungen Ihrer Container-Infrastruktur automatisieren. Bottlerocket enthält ausschließlich die für den Betrieb von Containern erforderliche Software. Dies führt zu einer verbesserten Ressourcennutzung, einer Reduzierung von Sicherheitsrisiken und einem geringeren Verwaltungsaufwand.

Nur VMware Varianten von Bottlerocket Version v1.37.0 und höher werden von EKS Hybrid Nodes unterstützt. VMware Varianten von Bottlerocket sind für die Kubernetes-Versionen v1.28 und höher verfügbar. Die Betriebssystem-Images für diese Varianten beinhalten Kubelet, Containerd aws-iam-authenticator und andere Softwarevoraussetzungen für EKS-Hybridknoten. Sie können diese Komponenten mithilfe einer Bottlerocket-Einstellungsdatei konfigurieren, die Base64-codierte Benutzerdaten für die Bottlerocket-Bootstrap- und Admin-Container enthält. Durch die Konfiguration dieser Einstellungen kann Bottlerocket Ihren Anmeldeinformationsanbieter für Hybridknoten verwenden, um Hybridknoten gegenüber Ihrem Cluster zu authentifizieren. Nachdem Ihre Hybridknoten dem Cluster beigetreten sind, werden sie mit Status `Not Ready` in der Amazon-EKS-Konsole und in Kubernetes-kompatiblen Tools wie `kubectl` angezeigt. Nachdem Sie die Schritte auf dieser Seite abgeschlossen haben, fahren Sie mit the section called “CNI konfigurieren” fort, um Ihre Hybridknoten für die Ausführung von Anwendungen vorzubereiten.

Voraussetzungen

Bevor Sie Hybridknoten mit Ihrem Amazon-EKS-Cluster verbinden, stellen Sie sicher, dass Sie die erforderlichen Schritte abgeschlossen haben.

- Sie haben Netzwerkkonnektivität von Ihrer lokalen Umgebung zur AWS Region, in der Ihr Amazon EKS-Cluster gehostet wird. Weitere Informationen finden Sie unter the section called “Netzwerk vorbereiten”.
- Sie haben Ihre IAM-Rolle Hybrid Nodes erstellt und Ihren lokalen Anbieter für Anmeldeinformationen eingerichtet (AWS Systems Manager Manager-Hybridaktivierungen oder AWS IAM Roles Anywhere). Weitere Informationen finden Sie unter the section called “Anmeldeinformationen vorbereiten”.
- Sie haben Ihren für Hybridknoten aktivierten Amazon-EKS-Cluster erstellt. Weitere Informationen finden Sie unter the section called “Cluster erstellen”.
- Sie haben Ihre Hybridknoten-IAM-Rolle den Berechtigungen der rollenbasierten Zugriffssteuerung (RBAC) von Kubernetes zugeordnet. Weitere Informationen finden Sie unter the section called “Cluster-Zugriff vorbereiten”.

Schritt 1: TOML-Datei für die Bottlerocket-Einstellungen erstellen

Um Bottlerocket für Hybridknoten zu konfigurieren, ist es erforderlich, eine `settings.toml`-Datei mit der notwendigen Konfiguration zu erstellen. Der Inhalt der TOML-Datei unterscheidet sich je nach verwendetem Anmeldeinformationsanbieter (SSM oder IAM Roles Anywhere). Diese Datei wird bei der Bereitstellung der Bottlerocket-Instance als Benutzerdaten übermittelt.

> **Note**
>
> Die unten angegebenen TOML-Dateien stellen nur die Mindesteinstellungen dar, die für die Initialisierung einer VMWare Bottlerocket-Maschine als Knoten in einem EKS-Cluster erforderlich sind. Bottlerocket bietet eine Vielzahl von Einstellungen für verschiedene Anwendungsfälle. Weitere Konfigurationsoptionen, die über die Initialisierung des Hybridknotens hinausgehen, finden Sie in der Bottlerocket-Dokumentation eine umfassende Liste aller dokumentierten Einstellungen für die von Ihnen verwendete Bottlerocket-Version (hier sind beispielsweise alle für Bottlerocket 1.51.x verfügbaren Einstellungen).

SSM

Wenn Sie AWS Systems Manager als Ihren Anmeldeinformationsanbieter verwenden, erstellen Sie eine `settings.toml` Datei mit dem folgenden Inhalt:

```
[settings.kubernetes]
cluster-name = "<cluster-name>"
api-server = "<api-server-endpoint>"
cluster-certificate = "<cluster-certificate-authority>"
hostname-override = "<hostname>"
provider-id = "eks-hybrid:///<region>/<cluster-name>/<hostname>"
authentication-mode = "aws"
cloud-provider = ""
server-tls-bootstrap = true

[settings.network]
hostname = "<hostname>"

[settings.aws]
region = "<region>"

[settings.kubernetes.credential-providers.ecr-credential-provider]
enabled = true
cache-duration = "12h"
image-patterns = [
    "*.dkr.ecr.*.amazonaws.com",
    "*.dkr.ecr.*.amazonaws.com.cn",
    "*.dkr.ecr.*.amazonaws.eu",
    "*.dkr.ecr-fips.*.amazonaws.com",
    "*.dkr.ecr-fips.*.amazonaws.eu",
    "public.ecr.aws"
]

[settings.kubernetes.node-labels]
"eks.amazonaws.com/compute-type" = "hybrid"
"eks.amazonaws.com/hybrid-credential-provider" = "ssm"

[settings.host-containers.admin]
enabled = true
user-data = "<base64-encoded-admin-container-userdata>"

[settings.bootstrap-containers.eks-hybrid-setup]
mode = "always"
user-data = "<base64-encoded-bootstrap-container-userdata>"
```

```
[settings.host-containers.control]
enabled = true
```

Ersetzen Sie die Platzhalter durch die folgenden Werte:

- `<cluster-name>`: Der Name Ihres Amazon-EKS-Clusters.
- `<api-server-endpoint>`: Der API-Server-Endpunkt Ihres Clusters.
- `<cluster-certificate-authority>`: Das base64-codierte CA-Paket Ihres Clusters.
- `<region>`: Die AWS Region, in der Ihr Cluster gehostet wird, zum Beispiel „us-east-1".
- `<hostname>`: Der Host-Name der Bottlerocket-Instance, der auch als Knotenname konfiguriert wird. Dies kann ein beliebiger eindeutiger Wert Ihrer Wahl sein, muss jedoch den Benennungskonventionen für Kubernetes-Objekte entsprechen. Außerdem darf der von Ihnen verwendete Hostname nicht länger als 64 Zeichen sein. HINWEIS: Bei Verwendung des SSM-Anbieters werden dieser Host-Name und der Knotenname nach der Registrierung der Instance bei SSM durch die ID der verwalteten Instance (z. B. `mi-*`-ID) ersetzt.
- `<base64-encoded-admin-container-userdata>`: Der Base64-codierte Inhalt der Bottlerocket-Admin-Container-Konfiguration. Durch Aktivieren des Admin-Containers können Sie zur Systemerkundung und zum Debuggen per SSH eine Verbindung zu Ihrer Bottlerocket-Instance herstellen. Obwohl dies keine erforderliche Einstellung ist, empfehlen wir, sie zur einfacheren Fehlerbehebung zu aktivieren. Weitere Informationen zur Authentifizierung mit dem Admin-Container finden Sie in der Dokumentation zum Bottlerocket-Admin-Container. Der Admin-Container akzeptiert SSH-Benutzer- und Schlüsseleingaben im JSON-Format, zum Beispiel:

```
{
  "user": "<ssh-user>",
  "ssh": {
    "authorized-keys": [
      "<ssh-authorized-key>"
    ]
  }
}
```

- `<base64-encoded-bootstrap-container-userdata>`: Der Base64-codierte Inhalt der Bottlerocket-Bootstrap-Container-Konfiguration. Weitere Informationen zur Konfiguration finden Sie in der Dokumentation zum Bottlerocket-Bootstrap-Container. Der Bootstrap-Container ist dafür verantwortlich, die Instance als AWS SSM-verwaltete Instance zu registrieren und ihr als

Kubernetes-Knoten in Ihrem Amazon EKS-Cluster beizutreten. Die an den Bootstrap-Container übergebenen Benutzerdaten haben die Form eines Befehlsaufrufs, der den zuvor erstellten SSM-Hybrid-Aktivierungscode und die ID als Eingabe akzeptiert:

```
eks-hybrid-ssm-setup --activation-id=<activation-id> --activation-code=<activation-
code> --region=<region>
```

IAM Roles Anywhere

Wenn Sie AWS IAM Roles Anywhere als Ihren Anbieter für Anmeldeinformationen verwenden, erstellen Sie eine `settings.toml` Datei mit dem folgenden Inhalt:

```
[settings.kubernetes]
cluster-name = "<cluster-name>"
api-server = "<api-server-endpoint>"
cluster-certificate = "<cluster-certificate-authority>"
hostname-override = "<hostname>"
provider-id = "eks-hybrid:///<region>/<cluster-name>/<hostname>"
authentication-mode = "aws"
cloud-provider = ""
server-tls-bootstrap = true

[settings.network]
hostname = "<hostname>"

[settings.aws]
region = "<region>"
config = "<base64-encoded-aws-config-file>"

[settings.kubernetes.credential-providers.ecr-credential-provider]
enabled = true
cache-duration = "12h"
image-patterns = [
    "*.dkr.ecr.*.amazonaws.com",
    "*.dkr.ecr.*.amazonaws.com.cn",
    "*.dkr.ecr.*.amazonaws.eu",
    "*.dkr.ecr-fips.*.amazonaws.com",
    "*.dkr.ecr-fips.*.amazonaws.eu",
    "public.ecr.aws"
]

[settings.kubernetes.node-labels]
```

```
"eks.amazonaws.com/compute-type" = "hybrid"
"eks.amazonaws.com/hybrid-credential-provider" = "iam-ra"

[settings.host-containers.admin]
enabled = true
user-data = "<base64-encoded-admin-container-userdata>"

[settings.bootstrap-containers.eks-hybrid-setup]
mode = "always"
user-data = "<base64-encoded-bootstrap-container-userdata>"
```

Ersetzen Sie die Platzhalter durch die folgenden Werte:

- `<cluster-name>`: Der Name Ihres Amazon-EKS-Clusters.
- `<api-server-endpoint>`: Der API-Server-Endpunkt Ihres Clusters.
- `<cluster-certificate-authority>`: Das base64-codierte CA-Paket Ihres Clusters.
- `<region>`: Die AWS Region, in der Ihr Cluster gehostet wird, z. B. „us-east-1"
- `<hostname>`: Der Host-Name der Bottlerocket-Instance, der auch als Knotenname konfiguriert wird. Dies kann ein beliebiger eindeutiger Wert Ihrer Wahl sein, muss jedoch den Benennungskonventionen für Kubernetes-Objekte entsprechen. Außerdem darf der von Ihnen verwendete Hostname nicht länger als 64 Zeichen sein. HINWEIS: Bei Verwendung des IAM-RA-Anbieters muss der Knotenname mit dem CN des Zertifikats auf dem Host übereinstimmen, wenn Sie die Vertrauensrichtlinie Ihrer IAM-Rolle für Hybridknoten mit der `"sts:RoleSessionName": "${aws:PrincipalTag/x509Subject/CN}"`-Ressourcenbedingung konfiguriert haben.
- `<base64-encoded-aws-config-file>`: Der Base64-kodierte Inhalt Ihrer Konfigurationsdatei. AWS Der Inhalt der Datei sollte wie folgt aussehen:

```
[default]
credential_process = aws_signing_helper credential-process --certificate /root/.aws/
node.crt --private-key /root/.aws/node.key --profile-arn <profile-arn> --role-arn
 <role-arn> --trust-anchor-arn <trust-anchor-arn> --role-session-name <role-session-
name>
```

- `<base64-encoded-admin-container-userdata>`: Der Base64-codierte Inhalt der Bottlerocket-Admin-Container-Konfiguration. Durch Aktivieren des Admin-Containers können Sie zur Systemerkundung und zum Debuggen per SSH eine Verbindung zu Ihrer Bottlerocket-Instance herstellen. Obwohl dies keine erforderliche Einstellung ist, empfehlen wir, sie zur einfacheren

Fehlerbehebung zu aktivieren. Weitere Informationen zur Authentifizierung mit dem Admin-Container finden Sie in der Dokumentation zum Bottlerocket-Admin-Container. Der Admin-Container akzeptiert SSH-Benutzer- und Schlüsseleingaben im JSON-Format, zum Beispiel:

```
{
  "user": "<ssh-user>",
  "ssh": {
    "authorized-keys": [
      "<ssh-authorized-key>"
    ]
  }
}
```

- `<base64-encoded-bootstrap-container-userdata>`: Der Base64-codierte Inhalt der Bottlerocket-Bootstrap-Container-Konfiguration. Weitere Informationen zur Konfiguration finden Sie in der Dokumentation zum Bottlerocket-Bootstrap-Container. Der Bootstrap-Container ist für die Erstellung der IAM Roles Anywhere-Host-Zertifikats- und Zertifikat-Privatschlüsseldateien auf der Instance verantwortlich. Diese werden dann von `aws_signing_helper` verwendet, um temporäre Anmeldeinformationen für die Authentifizierung bei Ihrem Amazon-EKS-Cluster zu erhalten. Die an den Bootstrap-Container übergebenen Benutzerdaten haben die Form eines Befehlsaufrufs, der als Eingabe den Inhalt des zuvor erstellten Zertifikats und des privaten Schlüssels akzeptiert:

```
eks-hybrid-iam-ra-setup --certificate=<certificate> --key=<private-key>
```

Schritt 2: Bottlerocket vSphere VM mit Benutzerdaten bereitstellen

Nachdem Sie die TOML-Datei erstellt haben, übergeben Sie sie während der Erstellung der vSphere-VM als Benutzerdaten. Beachten Sie, dass die Benutzerdaten konfiguriert werden müssen, bevor die VM zum ersten Mal gestartet wird. Daher müssen Sie diese bei der Erstellung der Instance angeben. Wenn Sie die VM im Voraus erstellen möchten, muss sie sich im Status „poweredOff“ befinden, bis Sie die Benutzerdaten dafür konfiguriert haben. Beispiel bei Verwendung der `govc`-CLI:

Erstellen einer VM zum ersten Mal

```
govc vm.create \
  -on=true \
  -c=2 \
```

```
  -m=4096 \
  -net.adapter=<network-adapter> \
  -net=<network-name> \
  -e guestinfo.userdata.encoding="base64" \
  -e guestinfo.userdata="$(base64 -w0 settings.toml)" \
  -template=<template-name> \
  <vm-name>
```

Aktualisieren der Benutzerdaten für eine vorhandene VM

```
govc vm.create \
    -on=false \
    -c=2 \
    -m=4096 \
    -net.adapter=<network-adapter> \
    -net=<network-name> \
    -template=<template-name> \
    <vm-name>

govc vm.change
    -vm <vm-name> \
    -e guestinfo.userdata="$(base64 -w0 settings.toml)" \
    -e guestinfo.userdata.encoding="base64"

govc vm.power -on <vm-name>
```

In den oben genannten Abschnitten gibt die `-e guestinfo.userdata.encoding="base64"`-Option an, dass die Benutzerdaten base64-codiert sind. Die `-e guestinfo.userdata`-Option übergibt den base64-codierten Inhalt der `settings.toml`-Datei als Benutzerdaten an die Bottlerocket-Instance. Ersetzen Sie die Platzhalter durch Ihre spezifischen Werte, wie beispielsweise die Bottlerocket-OVA-Vorlage und die Netzwerkdetails.

Schritt 3: Verbindung des Hybridknotens überprüfen

Nach dem Start der Bottlerocket-Instance wird versucht, eine Verbindung zu Ihrem Amazon-EKS-Cluster herzustellen. Sie können die Verbindung in der Amazon-EKS-Konsole überprüfen, indem Sie zur Registerkarte „Rechnen“ für Ihren Cluster navigieren oder den folgenden Befehl ausführen:

```
kubectl get nodes
```

> **Important**
>
> Ihre Knoten haben den Status `Not Ready`, was zu erwarten ist und darauf zurückzuführen ist, dass auf Ihren Hybridknoten kein CNI ausgeführt wird. Wenn Ihre Knoten nicht dem Cluster beigetreten sind, sehen Sie unter the section called “Fehlerbehebung” nach.

Schritt 4: CNI für Hybridknoten konfigurieren

Um Ihre Hybridknoten für die Ausführung von Anwendungen vorzubereiten, führen Sie die Schritte in the section called “CNI konfigurieren” aus.

Aktualisierung von Hybridknoten für Ihren Cluster

Die Anleitung zum Upgrade von Hybridknoten ähnelt der für selbstverwaltete Amazon-EKS-Knoten, die in Amazon EC2 ausgeführt werden. Wir empfehlen Ihnen, neue Hybridknoten auf Ihrer Kubernetes-Zielversion zu erstellen, Ihre vorhandenen Anwendungen ordnungsgemäß auf die Hybridknoten der neuen Kubernetes-Version zu migrieren und die Hybridknoten der alten Kubernetes-Version aus Ihrem Cluster zu entfernen. Lesen Sie unbedingt die Bewährten Methoden für Amazon-EKS-Upgrades, bevor Sie ein Upgrade starten. Amazon EKS Hybrid Nodes verfügen über dieselbe Kubernetes-Versionsunterstützung für Amazon-EKS-Cluster mit Cloud-Knoten, einschließlich Standard- und erweiterten Support.

Amazon EKS Hybrid Nodes befolgen dieselbe Versionsabweichungsrichtlinie für Knoten wie vorgeschaltete Kubernetes. Amazon EKS Hybrid Nodes dürfen keine neuere Version als die Amazon-EKS-Steuerebene aufweisen und Hybridknoten dürfen bis zu drei Kubernetes-Nebenversionen älter sein als die Nebenversion der Amazon-EKS-Steuerebene.

Wenn Sie keine freien Kapazitäten haben, um neue Hybridknoten auf Ihrer Zielversion von Kubernetes für eine Cutover-Migrations-Upgrade-Strategie zu erstellen, können Sie alternativ die CLI (`nodeadm`) für Amazon EKS Hybrid Nodes verwenden, um die Kubernetes-Version Ihrer Hybridknoten vor Ort zu aktualisieren.

> **Important**
>
> Wenn Sie Ihre Hybridknoten vor Ort mit `nodeadm` aktualisieren, kommt es während des Vorgangs zu Ausfallzeiten für den Knoten, bei denen die ältere Version der Kubernetes-Komponenten heruntergefahren und die Komponenten der neuen Kubernetes-Version installiert und gestartet werden.

Voraussetzungen

Stellen Sie vor dem Upgrade sicher, dass Sie die folgenden Voraussetzungen erfüllt haben.

- Die Ziel-Kubernetes-Version für Ihr Hybridknoten-Upgrade muss gleich oder niedriger als die Version der Amazon-EKS-Steuerebene sein.
- Wenn Sie eine Cutover-Migrations-Upgrade-Strategie verfolgen, müssen die neuen Hybridknoten, die Sie auf Ihrer Ziel-Kubernetes-Version installieren, die the section called “Voraussetzungen”-Anforderungen erfüllen. Dies umfasst IP-Adressen innerhalb des Fern-Knoten-Netzwerk-CIDR, das Sie bei der Erstellung des Amazon-EKS-Clusters angegeben haben.
- Sowohl für die Cutover-Migration als auch für direkte Upgrades müssen die Hybridknoten Zugriff auf die erforderlichen Domains haben, um die neuen Versionen der Abhängigkeiten der Hybridknoten abzurufen.
- Sie müssen kubectl auf Ihrem lokalen Rechner oder Ihrer Instance installiert haben, die Sie für die Interaktion mit Ihrem API-Endpunkt für Amazon EKS Kubernetes verwenden.
- Die Version Ihres CNI muss die Kubernetes-Version unterstützen, auf die Sie aktualisieren. Ist dies nicht der Fall, aktualisieren Sie Ihre CNI-Version, bevor Sie Ihre Hybridknoten aktualisieren. Weitere Informationen finden Sie unter the section called “CNI konfigurieren”.

Cutover-Migration (Blau-Grün)-Upgrades

Cutover-Migrations-Upgrades beziehen sich auf den Prozess der Erstellung neuer Hybridknoten auf neuen Hosts mit Ihrer Ziel-Kubernetes-Version, der ordnungsgemäßen Migration Ihrer vorhandenen Anwendungen auf die neuen Hybridknoten auf Ihrer Ziel-Kubernetes-Version und der Entfernung der Hybridknoten auf der alten Kubernetes-Version aus Ihrem Cluster. Diese Strategie wird auch als Blau-Grün-Migration bezeichnet.

1. Verbinden Sie Ihre neuen Hosts als Hybridknoten, indem Sie die Schritte the section called “Hybridknoten verbinden” befolgen. Verwenden Sie beim Ausführen des `nodeadm install`-Befehls Ihre Ziel-Kubernetes-Version.
2. Aktivieren Sie die Kommunikation zwischen den neuen Hybridknoten auf der Kubernetes-Zielversion und Ihren Hybridknoten auf der alten Kubernetes-Version. Diese Konfiguration ermöglicht die Kommunikation zwischen Pods, während Sie Ihre Workload auf die Hybridknoten der Kubernetes-Zielversion migrieren.
3. Bestätigen Sie, dass Ihre Hybridknoten auf Ihrer Ziel-Kubernetes-Version Ihrem Cluster erfolgreich beigetreten sind und den Status „Bereit“ aufweisen.

4. Verwenden Sie den folgenden Befehl, um jeden der Knoten, die Sie entfernen möchten, als nicht planbar zu markieren. Dadurch wird verhindert, dass auf den zu ersetzenden Knoten neue Pods geplant oder neu geplant werden. Weitere Informationen finden Sie unter kubectl cordon in der Kubernetes-Dokumentation. Ersetzen Sie `NODE_NAME` durch den Namen der Hybridknoten in der alten Kubernetes-Version.

```
kubectl cordon NODE_NAME
```

   Mit dem folgenden Code-Ausschnitt können Sie alle Knoten einer bestimmten Kubernetes-Version (in diesem Fall `1.28`) identifizieren und abgrenzen.

```
K8S_VERSION=1.28
for node in $(kubectl get nodes -o json | jq --arg K8S_VERSION
 "$K8S_VERSION" -r '.items[] | select(.status.nodeInfo.kubeletVersion |
 match("\($K8S_VERSION)")).metadata.name')
do
    echo "Cordoning $node"
    kubectl cordon $node
done
```

5. Wenn Ihre aktuelle Bereitstellung weniger als zwei CoreDNS-Replikate in Ihren Hybridknoten ausführt, skalieren die Bereitstellung auf mindestens zwei Replikate. Wir empfehlen, dass Sie aus Gründen der Ausfallsicherheit im Normalbetrieb mindestens zwei CoreDNS-Replikate in Hybridknoten ausführen.

```
kubectl scale deployments/coredns --replicas=2 -n kube-system
```

6. Leeren Sie jeden Hybridknoten auf der alten Kubernetes-Version, den Sie aus Ihrem Cluster entfernen möchten, mit dem folgenden Befehl. Weitere Informationen zum Entleeren von Knoten finden Sie unter Sicheres Entleeren eines Knotens in der Kubernetes-Dokumentation. Ersetzen Sie `NODE_NAME` durch den Namen der Hybridknoten in der alten Kubernetes-Version.

```
kubectl drain NODE_NAME --ignore-daemonsets --delete-emptydir-data
```

   Mit dem folgenden Code-Ausschnitt können Sie alle Knoten einer bestimmten Kubernetes-Version (in diesem Fall `1.28`) identifizieren und entleeren.

```
K8S_VERSION=1.28
```

```
for node in $(kubectl get nodes -o json | jq --arg K8S_VERSION
 "$K8S_VERSION" -r '.items[] | select(.status.nodeInfo.kubeletVersion |
 match("\($K8S_VERSION)")).metadata.name')
do
    echo "Draining $node"
    kubectl drain $node --ignore-daemonsets --delete-emptydir-data
done
```

7. Sie können `nodeadm` verwenden, um die Artefakte der Hybridknoten zu stoppen und vom Host zu entfernen. Sie müssen `nodeadm` mit einem Benutzer ausführen, der über Root-/Sudo-Berechtigungen verfügt. Standardmäßig wird `nodeadm uninstall` nicht fortgesetzt, wenn auf dem Knoten noch Pods vorhanden sind. Weitere Informationen finden Sie unter the section called “nodeadm für Hybridknoten”.

   ```
   nodeadm uninstall
   ```

8. Nachdem die Artefakte der Hybridknoten angehalten und deinstalliert wurden, entfernen Sie die Knoten-Ressource aus Ihrem Cluster.

   ```
   kubectl delete node node-name
   ```

   Mit dem folgenden Code-Ausschnitt können Sie alle Knoten einer bestimmten Kubernetes-Version (in diesem Fall `1.28`) identifizieren und löschen.

   ```
   K8S_VERSION=1.28
   for node in $(kubectl get nodes -o json | jq --arg K8S_VERSION
    "$K8S_VERSION" -r '.items[] | select(.status.nodeInfo.kubeletVersion |
    match("\($K8S_VERSION)")).metadata.name')
   do
       echo "Deleting $node"
       kubectl delete node $node
   done
   ```

9. Abhängig von Ihrer CNI-Wahl können nach der Ausführung der obigen Schritte Artefakte auf Ihren Hybridknoten verbleiben. Weitere Informationen finden Sie unter the section called “CNI konfigurieren”.

Direkte Upgrades

Der direkte Upgrade-Prozess bezieht sich auf die Verwendung von `nodeadm upgrade` zum Upgrade der Kubernetes-Version für Hybridknoten ohne Verwendung neuer physischer oder virtueller Hosts und einer Cutover-Migrationsstrategie. Der `nodeadm upgrade`-Prozess beendet die vorhandenen älteren Kubernetes-Komponenten, die auf dem Hybridknoten ausgeführt werden, deinstalliert die vorhandenen älteren Kubernetes-Komponenten, installiert die neuen Ziel-Kubernetes-Komponenten und startet die neuen Ziel-Kubernetes-Komponenten. Es wird dringend empfohlen, jeweils nur einen Knoten zu aktualisieren, um die Auswirkungen auf Anwendungen, die auf den Hybridknoten ausgeführt werden, zu minimieren. Die Dauer dieses Vorgangs hängt von Ihrer Netzwerkbandbreite und Latenz ab.

1. Verwenden Sie den folgenden Befehl, um den Knoten, den Sie aktualisieren, als nicht planbar zu markieren. Dadurch wird sichergestellt, dass auf dem Knoten, den Sie aktualisieren, keine neuen Pods geplant oder neu geplant werden. Weitere Informationen finden Sie unter kubectl cordon in der Kubernetes-Dokumentation. `NODE_NAME` durch den Namen des Hybridknotens, den Sie aktualisieren, ersetzen

   ```
   kubectl cordon NODE_NAME
   ```

2. Entleeren Sie den Knoten, den Sie aktualisieren, mit dem folgenden Befehl. Weitere Informationen zum Entleeren von Knoten finden Sie unter Sicheres Entleeren eines Knotens in der Kubernetes-Dokumentation. Ersetzen Sie `NODE_NAME` durch den Namen des Hybridknotens, den Sie aktualisieren.

   ```
   kubectl drain NODE_NAME --ignore-daemonsets --delete-emptydir-data
   ```

3. Führen Sie `nodeadm upgrade` auf dem Hybridknoten aus, den Sie aktualisieren. Sie müssen `nodeadm` mit einem Benutzer ausführen, der über Root-/Sudo-Berechtigungen verfügt. Der Name des Knotens bleibt während des Upgrades für die Anmeldeinformationsanbieter AWS SSM und AWS IAM Roles Anywhere erhalten. Sie können die Anmeldeinformationsanbieter während des Upgrades nicht ändern. Konfigurationswerte für `nodeConfig.yaml` finden Sie unter the section called “nodeadm für Hybridknoten”. Ersetzen Sie `K8S_VERSION` durch die Ziel-Version von Kubernetes, auf die Sie aktualisieren möchten.

   ```
   nodeadm upgrade K8S_VERSION -c file://nodeConfig.yaml
   ```

4. Um die Planung von Pods auf dem Knoten nach dem Upgrade zuzulassen, geben Sie Folgendes ein. Ersetzen Sie `NODE_NAME` durch den Namen des Knotens.

```
kubectl uncordon NODE_NAME
```

5. Überwachen Sie den Status Ihrer Hybridknoten und warten Sie, bis Ihre Knoten heruntergefahren sind und mit dem Status „Bereit“ auf der neuen Kubernetes-Version neu gestartet wurden.

```
kubectl get nodes -o wide -w
```

Patch-Sicherheitsaktualisierungen für Hybridknoten

In diesem Thema wird das Verfahren zum direkten Patchen von Sicherheitsupdates für bestimmte Pakete und Abhängigkeiten beschrieben, die auf Ihren Hybridknoten ausgeführt werden. Als bewährte Methode empfehlen wir Ihnen, Ihre Hybridknoten regelmäßig zu aktualisieren, um CVEs und Sicherheits-Patches zu erhalten.

Schritte zum Upgrade der Kubernetes-Version finden Sie unter the section called “Hybridknoten aktualisieren”.

Ein Beispiel für Software, die möglicherweise Sicherheits-Patches benötigt, ist `containerd`.

`Containerd`

`containerd` ist die Standard-Kubernetes-Container-Laufzeitumgebung und Kernabhängigkeit für EKS-Hybridknoten, die zur Verwaltung des Container-Lebenszyklus verwendet wird, einschließlich des Abrufens von Images und der Verwaltung der Container-Ausführung. Auf einem Hybridknoten können Sie `containerd` über die nodeadm CLI oder manuell installieren. Abhängig vom Betriebssystem Ihres Knotens installiert `nodeadm` `containerd` aus dem vom Betriebssystem bereitgestellten Paket oder dem Docker-Paket.

Wenn eine CVE in `containerd` veröffentlicht wurde, stehen Ihnen die folgenden Optionen zur Verfügung, um auf Ihren Hybridknoten ein Upgrade auf die gepatchte Version von `containerd` durchzuführen.

Schritt 1: Überprüfen, ob der Patch in den Paket-Managern veröffentlicht wurde

Sie können überprüfen, ob der `containerd`-CVE-Patch für den jeweiligen Betriebssystem-Paket-Manager veröffentlicht wurde, indem Sie die entsprechenden Sicherheitsberichte lesene:

- Amazon Linux 2023
- RHEL

- Ubuntu 20.04
- Ubuntu 22.04
- Ubuntu 24.04

Wenn Sie das Docker-Repository als Quelle von `containerd` verwenden, können Sie die Docker-Sicherheitsankündigungen überprüfen, um die Verfügbarkeit der gepatchten Version im Docker-Repository zu ermitteln.

Schritt 2: Methode zur Installation des Patches auswählen

Es gibt drei Methoden, um Sicherheitsupgrades direkt auf Knoten zu patchen und zu installieren. Welche Methode Sie verwenden können, hängt davon ab, ob der Patch vom Betriebssystem im Paket-Manager verfügbar ist oder nicht:

1. Installieren Sie Patches mit `nodeadm upgrade`, die in Paket-Managern veröffentlicht wurden, siehe Schritt 2 a.
2. Installieren Sie Patches direkt mit den Paket-Managern, siehe Schritt 2 b.
3. Installieren Sie benutzerdefinierte Patches, die nicht in Paket-Managern veröffentlicht wurden. Beachten Sie, dass für benutzerdefinierte Patches für `containerd`, Schritt 2 c, besondere Überlegungen angestellt werden müssen.

Schritt 2 a: Patchen mit **`nodeadm upgrade`**

Nachdem Sie bestätigt haben, dass der `containerd`-CVE-Patch im Betriebssystem oder in den Docker-Repos (entweder Apt oder RPM) veröffentlicht wurde, können Sie mit dem `nodeadm upgrade`-Befehl auf die neueste Version von `containerd` aktualisieren. Da es sich hierbei nicht um ein Upgrade der Kubernetes-Version handelt, müssen Sie Ihre aktuelle Kubernetes-Version an den `nodeadm`-Upgrade-Befehl übergeben.

```
nodeadm upgrade K8S_VERSION --config-source file:///root/nodeConfig.yaml
```

Schritt 2 b: Patchen mit Paket-Managern des Betriebssystems

Alternativ können Sie auch über den jeweiligen Paket-Manager aktualisieren und damit das `containerd`-Paket wie folgt aktualisieren.

Amazon Linux 2023

```
sudo yum update -y
sudo yum install -y containerd
```

RHEL

```
sudo yum install -y yum-utils
sudo yum-config-manager --add-repo https://download.docker.com/linux/rhel/docker-
ce.repo
sudo yum update -y
sudo yum install -y containerd
```

Ubuntu

```
sudo mkdir -p /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/
docker.asc
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc]
 https://download.docker.com/linux/ubuntu \
  $(. /etc/os-release && echo "${UBUNTU_CODENAME:-$VERSION_CODENAME}") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt update -y
sudo apt install -y --only-upgrade containerd.io
```

Schritt 2 c: **`Containerd`**-CVE-Patch nicht in Paket-Managern veröffentlicht

Wenn die gepatchte `containerd`-Version nur über andere Kanäle als den Paket-Manager verfügbar ist, beispielsweise in GitHub-Veröffentlichungen, können Sie `containerd` von der offiziellen GitHub-Website installieren.

1. Wenn der Rechner bereits als Hybridknoten dem Cluster beigetreten ist, müssen Sie den `nodeadm uninstall`-Befehl ausführen.
2. Installieren Sie die offiziellen `containerd`-Binärdateien. Befolgen Sie die offiziellen Installationsschritte auf GitHub.
3. Führen Sie den `nodeadm install`-Befehl mit dem `--containerd-source`-Argument `none` aus, wodurch die `containerd`-Installation über `nodeadm` übersprungen wird. Sie können den Wert `none` in der `containerd`-Quelle für jedes Betriebssystem verwenden, auf dem der Knoten ausgeführt wird.

```
nodeadm install K8S_VERSION --credential-provider CREDS_PROVIDER --containerd-source
 none
```

Hybridknoten entfernen

In diesem Thema wird beschrieben, wie Sie Hybridknoten aus Ihrem Amazon-EKS-Cluster löschen. Sie müssen Ihre Hybridknoten mit einem Kubernetes-kompatiblen Tool Ihrer Wahl, wie beispielsweise kubectl, löschen. Die Gebühren für Hybridknoten werden eingestellt, sobald das Knotenobjekt aus dem Amazon-EKS-Cluster entfernt wird. Weitere Informationen zu den Preisen für Hybridknoten finden Sie unter Amazon EKS – Preise.

> **Important**
>
> Das Entfernen von Knoten führt zu einer Unterbrechung der auf dem Knoten ausgeführten Workloads. Bevor Sie Hybridknoten löschen, empfehlen wir Ihnen, zunächst den Knoten zu entleeren, um Pods auf einen anderen aktiven Knoten zu verschieben. Weitere Informationen zum Entleeren von Knoten finden Sie unter Sicheres Entleeren eines Knotens in der Kubernetes-Dokumentation.

Führen Sie die folgenden kubectl-Schritte von Ihrem lokalen Computer oder Ihrer Instance aus, die Sie für die Interaktion mit dem Kubernetes-API-Endpunkt des Amazon-EKS-Clusters verwenden. Wenn Sie eine bestimmte `kubeconfig`-Datei verwenden, verwenden Sie das `--kubeconfig`-Flag.

Schritt 1: Knoten auflisten

```
kubectl get nodes
```

Schritt 2: Knoten entleeren

Weitere Informationen zu diesem `kubectl drain`-Befehl finden Sie unter kubectl drain in der Kubernetes-Dokumentation.

```
kubectl drain --ignore-daemonsets <node-name>
```

Schritt 3: Hybridknoten-Artefakte beenden und deinstallieren

Sie können die CLI von Amazon EKS Hybrid Nodes (`nodeadm`) verwenden, um die Hybridknoten-Artefakte auf dem Host zu beenden und zu entfernen. Sie müssen `nodeadm` mit einem Benutzer ausführen, der über Root-/Sudo-Berechtigungen verfügt. Standardmäßig wird `nodeadm uninstall` nicht fortgesetzt, wenn auf dem Knoten noch Pods vorhanden sind. Wenn Sie AWS Systems Manager (SSM) als Anmeldeinformationsanbieter verwenden, wird der Host mit diesem `nodeadm uninstall`-Befehl als von AWS SSM verwaltete Instance deregistriert. Weitere Informationen finden Sie unter the section called “nodeadm für Hybridknoten”.

```
nodeadm uninstall
```

Schritt 4: Knoten aus dem Cluster entfernen

Nachdem die Artefakte der Hybridknoten angehalten und deinstalliert wurden, entfernen Sie die Knoten-Ressource aus Ihrem Cluster.

```
kubectl delete node <node-name>
```

Schritt 5: Auf verbleibende Artefakte überprüfen

Abhängig von Ihrer CNI-Wahl können nach der Ausführung der obigen Schritte Artefakte auf Ihren Hybridknoten verbleiben. Weitere Informationen finden Sie unter the section called “CNI konfigurieren”.

Konfiguration von Anwendungsnetzwerken, Add-Ons und Webhooks für Hybridknoten

Nachdem Sie einen EKS-Cluster für Hybridknoten erstellt haben, konfigurieren Sie zusätzliche Funktionen für Anwendungsnetzwerke (CNI, BGP, Ingress, Load Balancing, Netzwerkrichtlinien), Add-Ons, Webhooks und Proxy-Einstellungen. Eine vollständige Liste der EKS- und Community-Add-Ons, die mit Hybridknoten kompatibel sind, finden Sie unte the section called “Add-Ons konfigurieren”.

EKS-Cluster-Erkenntnisse EKS umfasst Überprüfungen auf Fehlkonfigurationen in Ihrer Hybridknoten-Einrichtung, welche die Funktionalität Ihres Clusters oder Ihrer Workloads beeinträchtigen könnten. Weitere Informationen zu Cluster- Erkenntnissen finden Sie unter the section called “Cluster-Einblicke”.

Nachfolgend sind die allgemeinen Funktionen und Add-Ons aufgeführt, die Sie mit Hybridknoten verwenden können:

- Container Networking Interface (CNI): AWS unterstützt Cilium als CNI für Hybridknoten. Weitere Informationen finden Sie unter the section called “CNI konfigurieren”. Beachten Sie, dass das AWS VPC CNI nicht mit Hybridknoten verwendet werden kann.
- CoreDNS und **`kube-proxy`** : CoreDNS und `kube-proxy` werden automatisch installiert, wenn Hybridknoten dem EKS-Cluster beitreten. Diese Add-Ons können nach der Cluster-Erstellung als EKS-Add-Ons verwaltet werden.
- Ingress und Lastenausgleich: Sie können den AWS Load Balancer Controller und den Application Load Balancer (ALB) oder Network Load Balancer (NLB) mit dem Zieltyp `ip` für Workloads verwenden, die auf Hybridknoten ausgeführt werden. AWS unterstützt die in Cilium integrierten Feature für Ingress, Gateway und Kubernetes-Service-Lastenausgleich für Workloads, die auf Hybridknoten ausgeführt werden. Weitere Informationen erhalten Sie unter the section called “Ingress konfigurieren” und the section called “LoadBalancer-Services konfigurieren”.
- Metriken: Sie können Amazon Managed Service für Prometheus (AMP) agentenlose Scraper, AWS Distro for Open Telemetry (ADOT) und den Amazon-CloudWatch-Beobachtbarkeits-Agent mit hybriden Knoten verwenden. Um agentenlose AMP-Scraper für Pod-Metriken in Hybridknoten zu verwenden, müssen Ihre Pods über die VPC, die Sie für den EKS-Cluster verwenden, zugänglich sein.
- Protokolle: Sie können die Protokollierung der EKS-Steuerebene für Cluster mit aktivierten Hybridknoten aktivieren. Sie können das ADOT-EKS-Add-On und das EKS-Add-On für den Amazon-CloudWatch-Beobachtbarkeits-Agenten für die Protokollierung von Hybridknoten und Pods verwenden.
- Pod Identities und IRSA: Sie können EKS Pod Identities und IAM-Rollen für Servicekonten (IRSA) mit Anwendungen verwenden, die in Hybridknoten ausgeführt werden, um einen differenzierten Zugriff für Ihre Pods zu ermöglichen, die auf Hybridknoten mit anderen AWS-Services ausgeführt werden.
- Webhooks: Wenn Sie Webhooks ausführen, finden Sie unter the section called “Webhooks konfigurieren” Überlegungen und Schritte zum optionalen Ausführen von Webhooks auf Cloud-Knoten, falls Sie Ihre On-Premises-Pod-Netzwerke nicht routingfähig machen können.
- Proxy: Wenn Sie in Ihrer On-Premises-Umgebung einen Proxy-Server für den Datenverkehr verwenden, der Ihr Rechenzentrum oder Ihre Edge-Umgebung verlässt, können Sie Ihre Hybridknoten und Ihren Cluster so konfigurieren, dass sie Ihren Proxy-Server verwenden. Weitere Informationen finden Sie unter the section called “Proxy konfigurieren”.

Themen

- CNI für Hybridknoten konfigurieren
- Konfiguration von Add-Ons für Hybridknoten
- Konfiguration von Webhooks für Hybridknoten
- Konfiguration des Proxys für Hybridknoten
- Konfiguration von Cilium BGP für Hybridknoten
- Konfiguration von Kubernetes Ingress für Hybridknoten
- Konfiguration von Services vom Typ LoadBalancer für Hybridknoten
- Konfiguration von Kubernetes-Netzwerkrichtlinien für Hybridknoten

CNI für Hybridknoten konfigurieren

Cilium ist das AWS unterstützte Container Networking Interface (CNI) für Amazon EKS-Hybridknoten. Sie müssen ein CNI für Hybridknoten installieren, damit diese für die Verarbeitung von Workloads bereit sind. Hybridknoten werden mit dem Status `Not Ready` angezeigt, bis ein CNI ausgeführt wird. Sie können das CNI mit Tools Ihrer Wahl, wie beispielsweise Helm, verwalten. Die Anweisungen auf dieser Seite behandeln das Lebenszyklusmanagement von Cilium (Installation, Upgrade, Löschung). Informationen zur Konfiguration von Cilium für Ingress, Load Balancing und Netzwerkrichtlinien finden Sie unter the section called “Übersicht über Cilium Ingress und Cilium Gateway”, the section called “Servicetyp LoadBalancer” und the section called “Netzwerkrichtlinien konfigurieren”.

Cilium wird nicht unterstützt, AWS wenn es auf Knoten in der Cloud ausgeführt wird. AWS Das Amazon VPC CNI ist nicht mit Hybridknoten kompatibel und das VPC CNI ist mit Anti-Affinität für das Label `eks.amazonaws.com/compute-type: hybrid` konfiguriert.

Die zuvor auf dieser Seite befindliche Calico-Dokumentation wurde in das EKS-Repository für Hybrid-Beispiele verschoben.

Versionskompatibilität

Cilium-Versionen `v1.17.x` und `v1.18.x` werden für EKS-Hybridknoten für jede in Amazon EKS unterstützte Kubernetes-Version unterstützt.

> **Note**
>
> Kernelanforderung für Cilium v1.18.3: Aufgrund der Kernelanforderungen (Linux-Kernel >= 5.10) wird Cilium v1.18.3 nicht unterstützt auf:

- Ubuntu 20.04
- Red Hat Enterprise Linux (RHEL) 8

Die Systemanforderungen finden Sie unter Systemanforderungen für Cilium.

Informationen zu den von Amazon EKS unterstützten Kubernetes-Versionen finden Sie unter Kubernetes-Versionsunterstützung. EKS-Hybridknoten verfügen über dieselbe Kubernetes-Versionsunterstützung wie Amazon-EKS-Cluster mit Cloud-Knoten.

Unterstützte Funktionen

AWS verwaltet Builds von Cilium für EKS-Hybridknoten, die auf dem Open-Source-Projekt Cilium basieren. Um Support von AWS Cilium zu erhalten, müssen Sie die von Cilium AWS gepflegten Cilium-Builds und die unterstützten Cilium-Versionen verwenden.

AWS bietet technischen Support für die Standardkonfigurationen der folgenden Funktionen von Cilium zur Verwendung mit EKS-Hybridknoten. Wenn Sie beabsichtigen, Funktionen außerhalb des AWS Supports zu nutzen, wird empfohlen, alternativen kommerziellen Support für Cilium in Anspruch zu nehmen oder über das interne Fachwissen zu verfügen, um Fehler zu beheben und Problemlösungen für das Cilium-Projekt bereitzustellen.

Cilium-Feature	Unterstützt von AWS
Kubernetes-Netzwerkkonformität	Ja
Konnektivität des zentralen Clusters	Ja
IP-Familie	IPv4
Verwaltung des Lebenszyklus	Helm
Netzwerkmodus	VXLAN-Kapselung

Cilium-Feature	Unterstützt von AWS
IP-Adressverwaltung (IPAM)	Cilium-IPAM-Cluster-Umfang
Netzwerkrichtlinie	Kubernetes-Netzwerkrichtlinie
Border Gateway Protocol (BGP)	Cilium-BGP-Steuerebene
Kubernetes Ingress	Cilium Ingress, Cilium-Gateway
Zuweisung von LoadBalancer Dienst-IP-Adressen	Cilium Load Balancer IPAM
Werbung für die LoadBalancer IP-Adresse des Dienstes	Cilium-BGP-Steuerebene
Ersatz für kube-proxy	Ja

Überlegungen zu Cilium

- Helm-Repository — AWS hostet das Cilium Helm-Diagramm im Amazon Elastic Container Registry Public (Amazon ECR Public) bei Amazon EKS Cilium/Cilium. Zu den verfügbaren Versionen gehören:
  - Cilium v1.17.9: `oci://public.ecr.aws/eks/cilium/cilium:1.17.9-0`
  - Zilium v1.18.3: `oci://public.ecr.aws/eks/cilium/cilium:1.18.3-0`

    Die Befehle in diesem Thema verwenden dieses Repository. Beachten Sie, dass bestimmte `helm repo`-Befehle für Helm-Repositorys in Amazon ECR Public nicht gültig sind. Sie können daher nicht von einem lokalen Helm-Repository-Namen auf dieses Repository verweisen. Verwenden Sie stattdessen in den meisten Befehlen die vollständige URI.
- Standardmäßig ist Cilium so konfiguriert, dass es im Überlagerungs-/Tunnelmodus mit VXLAN als Kapselungsmethode ausgeführt wird. Dieser Modus stellt die geringsten Anforderungen an das zugrunde liegende physische Netzwerk.
- Standardmäßig maskiert Cilium die Quell-IP-Adresse des gesamten Pod-Datenverkehrs, der den Cluster verlässt, mit der IP-Adresse des Knotens. Wenn Sie Masquerading deaktivieren, CIDRs muss Ihr Pod in Ihrem lokalen Netzwerk routingfähig sein.

- Wenn Sie Webhooks auf Hybridknoten ausführen, CIDRs muss Ihr Pod in Ihrem lokalen Netzwerk routingfähig sein. Wenn CIDRs Ihr Pod in Ihrem lokalen Netzwerk nicht routbar ist, wird empfohlen, Webhooks auf Cloud-Knoten im selben Cluster auszuführen. Weitere Informationen finden Sie unter the section called “Webhooks konfigurieren” und the section called “Netzwerk vorbereiten”.
- AWS empfiehlt, die integrierte BGP-Funktionalität von Cilium zu verwenden, um Ihren Pod in Ihrem lokalen Netzwerk CIDRs routingfähig zu machen. Weitere Informationen zur Konfiguration von Cilium BGP mit Hybridknoten finden Sie unter the section called “BGP konfigurieren”.
- Die standardmäßige IP-Adressverwaltung (IPAM) in Cilium heißt Cluster Scope. Dabei weist der Cilium-Operator jedem Knoten auf der Grundlage des vom Benutzer konfigurierten Pods IP-Adressen zu. CIDRs

Cilium in Hybridknoten installieren

Verfahren

1. Erstellen Sie eine YAML-Datei mit dem Namen `cilium-values.yaml`. Im folgenden Beispiel wird Cilium so konfiguriert, dass es nur in Hybridknoten ausgeführt wird, indem die Affinität für das `eks.amazonaws.com/compute-type: hybrid`-Label für den Cilium-Agenten und -Operator festgelegt wird.
   - Verwenden Sie für die Konfiguration *`clusterPoolIpv4PodCIDRList`* denselben Pod, den CIDRs Sie für die Remote-Pod-Netzwerke Ihres EKS-Clusters konfiguriert haben. Beispiel, `10.100.0.0/24`. Der Cilium-Operator weist IP-Adressbereiche aus der konfigurierten `clusterPoolIpv4PodCIDRList`-IP-Umgebung zu. Ihr Pod-CIDR darf sich nicht mit Ihrem On-Premises-Knoten-CIDR, Ihrem VPC-CIDR oder Ihrem Kubernetes-Service-CIDR überschneiden.
   - Konfigurieren Sie `clusterPoolIpv4MaskSize` basierend auf den von Ihnen benötigten Pods pro Knoten. Beispiel: `25` für eine /25-Segmentgröße von 128 Pods pro Knoten.
   - Ändern Sie `clusterPoolIpv4PodCIDRList` oder `clusterPoolIpv4MaskSize` nicht, nachdem Sie Cilium in Ihrem Cluster bereitgestellt haben. Weitere Informationen finden Sie unter Erweiterung des Cluster-Pools.
   - Wenn Sie Cilium im Kube-Proxy-Ersatzmodus ausführen, setzen Sie `kubeProxyReplacement: "true"` in Ihren Helm-Werten und stellen Sie sicher, dass Sie keine vorhandene Kube-Proxy-Bereitstellung auf denselben Knoten wie Cilium ausführen.
   - Das folgende Beispiel deaktiviert den Envoy Layer 7 (L7)-Proxy, den Cilium für L7-Netzwerkrichtlinien und Ingress verwendet. Weitere Informationen erhalten Sie unter the section

called “Netzwerkrichtlinien konfigurieren” und the section called “Übersicht über Cilium Ingress und Cilium Gateway”.

- Das nachstehende Beispiel konfiguriert `loadBalancer.serviceTopology: true` für die Verteilung des Service-Datenverkehrs, damit dieser korrekt funktioniert, wenn Sie ihn für Ihre Services konfigurieren. Weitere Informationen finden Sie unter the section called “Verteilung des Service-Datenverkehrs konfigurieren”.
- Eine vollständige Liste der Helm-Werte für Cilium finden Sie in der Helm-Referenz in der Cilium-Dokumentation.

```
affinity:
  nodeAffinity:
    requiredDuringSchedulingIgnoredDuringExecution:
      nodeSelectorTerms:
      - matchExpressions:
        - key: eks.amazonaws.com/compute-type
          operator: In
          values:
          - hybrid
ipam:
  mode: cluster-pool
  operator:
    clusterPoolIPv4MaskSize: 25
    clusterPoolIPv4PodCIDRList:
    - POD_CIDR
loadBalancer:
  serviceTopology: true
operator:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: eks.amazonaws.com/compute-type
            operator: In
            values:
              - hybrid
  unmanagedPodWatcher:
    restart: false
loadBalancer:
  serviceTopology: true
envoy:
  enabled: false
```

```
kubeProxyReplacement: "false"
```

2. Installieren Sie Cilium auf Ihrem Cluster.
   - `CILIUM_VERSION`Ersetzen Sie es durch eine Cilium-Version (zum Beispiel `1.17.9-0` oder`1.18.3-0`). Es wird empfohlen, die neueste Patch-Version für die Cilium-Nebenversion zu verwenden.
   - Stellen Sie sicher, dass Ihre Knoten die Kernel-Anforderungen für die von Ihnen gewählte Version erfüllen. Cilium v1.18.3 erfordert einen Linux-Kernel >= 5.10.
   - Wenn Sie eine bestimmte Kubeconfig-Datei verwenden, verwenden Sie das Flag `--kubeconfig` mit dem Helm-Installationsbefehl.

```
helm install cilium oci://public.ecr.aws/eks/cilium/cilium \
    --version CILIUM_VERSION \
    --namespace kube-system \
    --values cilium-values.yaml
```

3. Bestätigen Sie mit den folgenden Befehlen, dass Ihre Cilium-Installation erfolgreich war Sie sollten die `cilium-operator`-Bereitstellung und die `cilium-agent`-Ausführung auf jedem Ihrer Hybridknoten anzeigen können. Darüber hinaus sollten Ihre Hybridknoten nun den Status `Ready` aufweisen. Informationen darüber, wie Sie Cilium BGP so konfigurieren, dass Ihr Pod in Ihrem lokalen Netzwerk angekündigt wird, finden CIDRs Sie unter. the section called “BGP konfigurieren”

```
kubectl get pods -n kube-system
```

```
NAME                              READY   STATUS    RESTARTS   AGE
cilium-jjjn8                      1/1     Running   0          11m
cilium-operator-d4f4d7fcb-sc5xn   1/1     Running   0          11m
```

```
kubectl get nodes
```

```
NAME                   STATUS   ROLES    AGE   VERSION
mi-04a2cf999b7112233   Ready    <none>   19m   v1.31.0-eks-a737599
```

Cilium in Hybridknoten aktualisieren

Lesen Sie vor dem Upgrade Ihrer Cilium-Bereitstellung die Cilium-Upgrade-Dokumentation und die Upgrade-Hinweise sorgfältig durch, um die Änderungen in der Zielversion von Cilium zu verstehen.

1. Stellen Sie sicher, dass Sie die `helm`-CLI in Ihrer Befehlszeilenumgebung installiert haben. Installationsanweisungen finden Sie in der Helm-Dokumentation.
2. Führen Sie die Vorabprüfung für das Cilium-Upgrade durch. Ersetzen Sie `CILIUM_VERSION` durch Ihre gewünschte Cilium-Version. Wir empfehlen, die neueste Patch-Version für Ihre Cilium-Nebenversion zu verwenden. Die neueste Patch-Version für eine bestimmte Cilium-Nebenversion finden Sie im Abschnitt Stabile Versionen der Cilium-Dokumentation.

```
helm install cilium-preflight oci://public.ecr.aws/eks/cilium/cilium --version
 CILIUM_VERSION \
  --namespace=kube-system \
  --set preflight.enabled=true \
  --set agent=false \
  --set operator.enabled=false
```

3. Stellen Sie nach dem Anwenden von `cilium-preflight.yaml` sicher, dass die Anzahl der `READY`-Pods mit der Anzahl der ausgeführten Cilium-Pods übereinstimmt.

```
kubectl get ds -n kube-system | sed -n '1p;/cilium/p'
```

```
NAME                      DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE
 SELECTOR   AGE
cilium                    2         2         2       2            2           <none>
          1h20m
cilium-pre-flight-check   2         2         2       2            2           <none>
          7m15s
```

4. Sobald die Anzahl der READY-Pods gleich ist, stellen Sie sicher, dass die Cilium-Bereitstellung vor dem Flug auch als READY 1/1 markiert ist. Wenn „READY 0/1“ angezeigt wird, lesen Sie den Abschnitt zur CNP-Validierung und beheben Sie Probleme mit der Bereitstellung, bevor Sie mit dem Upgrade fortfahren.

```
kubectl get deployment -n kube-system cilium-pre-flight-check -w
```

```
NAME                      READY   UP-TO-DATE   AVAILABLE   AGE
cilium-pre-flight-check   1/1     1            0           12s
```

5. Vorabprüfung löschen

```
helm uninstall cilium-preflight --namespace kube-system
```

6. Bewahren Sie vor dem Ausführen des `helm upgrade`-Befehls die Werte für Ihre Bereitstellung in einem `existing-cilium-values.yaml` auf oder verwenden Sie `--set`-Befehlszeilenoptionen für Ihre Einstellungen, wenn Sie den Upgrade-Befehl ausführen. Der Upgrade-Vorgang überschreibt das Cilium ConfigMap, daher ist es wichtig, dass Ihre Konfigurationswerte beim Upgrade übergeben werden.

```
helm get values cilium --namespace kube-system -o yaml > existing-cilium-values.yaml
```

7. Im normalen Cluster-Betrieb sollten alle Cilium-Komponenten in derselben Version ausgeführt werden. Die folgenden Schritte beschreiben, wie Sie alle Komponenten von einer stabilen Version auf eine spätere stabile Version aktualisieren. Beim Upgrade von einer Nebenversion auf eine andere Nebenversion wird empfohlen, zuerst auf die neueste Patch-Version für die vorhandene Cilium-Nebenversion zu aktualisieren. Um Störungen zu minimieren, setzen Sie die Option `upgradeCompatibility` auf die ursprüngliche Cilium-Version, die Sie in diesem Cluster installiert haben.

```
helm upgrade cilium oci://public.ecr.aws/eks/cilium/cilium --version CILIUM_VERSION \
  --namespace kube-system \
  --set upgradeCompatibility=1.X \
  -f existing-cilium-values.yaml
```

8. (Optional) Wenn Sie Ihr Upgrade aufgrund von Problemen zurücksetzen müssen, führen Sie die folgenden Befehle aus.

```
helm history cilium --namespace kube-system
helm rollback cilium [REVISION] --namespace kube-system
```

Cilium aus Hybridknoten löschen

1. Führen Sie den folgenden Befehl aus, um alle Cilium-Komponenten aus Ihrem Cluster zu deinstallieren. Beachten Sie, dass die Deinstallation des CNI die Integrität von Knoten und Pods beeinträchtigen kann und nicht in Produktions-Clustern durchgeführt werden sollte.

```
helm uninstall cilium --namespace kube-system
```

   Die von Cilium konfigurierten Schnittstellen und Routen werden standardmäßig nicht entfernt, wenn das CNI aus dem Cluster entfernt wird. Weitere Informationen finden Sie unter dem GitHub Problem.

2. Um die Konfigurationsdateien und Ressourcen auf der Festplatte zu bereinigen, können Sie, wenn Sie die Standardkonfigurationsverzeichnisse verwenden, die Dateien wie im `cni-uninstall.sh`Skript im Cilium-Repository unter gezeigt entfernen. GitHub
3. Um die benutzerdefinierten Cilium-Ressourcendefinitionen (CRDs) aus Ihrem Cluster zu entfernen, können Sie die folgenden Befehle ausführen.

```
kubectl get crds -oname | grep "cilium" | xargs kubectl delete
```

Konfiguration von Add-Ons für Hybridknoten

Auf dieser Seite werden Überlegungen zum Ausführen von AWS Add-Ons und Community-Add-Ons auf Amazon EKS-Hybridknoten beschrieben. Weitere Informationen zu Amazon-EKS-Add-Ons und den Prozessen zum Erstellen, Aktualisieren und Entfernen von Add-Ons aus Ihrem Cluster finden Sie unter the section called “Amazon-EKS-Add-ons”. Sofern auf dieser Seite nicht anders angegeben, sind die Verfahren zum Erstellen, Aktualisieren und Entfernen von Amazon EKS-Add-Ons für Amazon EKS-Cluster mit Hybridknoten identisch wie für Amazon EKS-Cluster mit Knoten, die in der AWS Cloud laufen. Nur die auf dieser Seite aufgeführten Add-Ons wurden auf Kompatibilität mit Amazon EKS Hybrid Nodes geprüft.

Die folgenden AWS Add-Ons sind mit Amazon EKS Hybrid Nodes kompatibel.

AWS Add-on	Kompatible Add-On-Versionen
kube-proxy	v1.25.14-eksbuild.2 und höher
CoreDNS	v1.9.3-eksbuild.7 und höher
AWS Distribution für OpenTelemetry (ADOT)	v0.102.1-eksbuild.2 und höher
CloudWatch Agent für Beobachtbarkeit	v2.2.1-eksbuild.1 und höher
EKS Pod Identity Agent	• v1.3.3-eksbuild.1 und höher, mit Ausnahme von Bottlerocket • v1.3.7-eksbuild.2 und höher für Bottlerocket
Knotenüberwachungsagent	v1.2.0-eksbuild.1 und höher
CSI-Snapshot-Controller	v8.1.0-eksbuild.1 und höher

AWS Add-on	Kompatible Add-On-Versionen
AWS Privater CA-Konnektor für Kubernetes	v1.6.0-eksbuild.1 und höher

Die folgenden Community-Add-Ons sind mit Amazon EKS Hybrid Nodes kompatibel. Weitere Informationen zu Community-Add-Ons finden Sie unter the section called “Community-Erweiterungen”.

Community-Add-On	Kompatible Add-On-Versionen
Kubernetes-Metriken-Server	v0.7.2-eksbuild.1 und höher
cert-manager	v1.17.2-eksbuild.1 und höher
Prometheus-Knoten-Exporter	v1.9.1-eksbuild.2 und höher
kube-state-metrics	v2.15.0-eksbuild.4 und höher
Externes DNS	v0.19.0-eksbuild.1 und höher

Zusätzlich zu den Amazon-EKS-Add-Ons in den obigen Tabellen sind der Amazon Managed Service für Prometheus Collector und der AWS Load Balancer Controller für Anwendungseingang (HTTP) und Load Balancing (TCP/UDP) mit Hybridknoten kompatibel.

Es gibt AWS Add-Ons und Community-Add-Ons, die nicht mit Amazon EKS Hybrid Nodes kompatibel sind. Die neuesten Versionen dieser Add-Ons verfügen über eine Anti-Affinitätsregel für die standardmäßige `eks.amazonaws.com/compute-type: hybrid`-Kennzeichnung, das auf Hybridknoten angewendet wird. Dadurch wird verhindert, dass sie auf Hybridknoten ausgeführt werden, wenn sie in Ihren Clustern bereitgestellt werden. Wenn Sie Cluster mit sowohl Hybridknoten als auch Knoten haben, die in der AWS Cloud ausgeführt werden, können Sie diese Add-Ons in Ihrem Cluster auf Knoten bereitstellen, die in der AWS Cloud ausgeführt werden. Das Amazon VPC CNI ist nicht mit Hybridknoten kompatibel, und Cilium und Calico werden als Container Networking Interfaces (CNIs) für Amazon EKS-Hybridknoten unterstützt. Weitere Informationen finden Sie unter the section called “CNI konfigurieren”.

AWS Add-Ons

In den folgenden Abschnitten werden die Unterschiede zwischen der Ausführung kompatibler AWS Add-Ons auf Hybridknoten im Vergleich zu anderen Amazon EKS-Compute-Typen beschrieben.

kube-proxy und CoreDNS

EKS installiert Kube-Proxy und CoreDNS standardmäßig als selbstverwaltete Add-Ons, wenn Sie einen EKS-Cluster mit der AWS API und AWS SDKs, auch über die CLI, erstellen. AWS Sie können diese Add-Ons nach der Cluster-Erstellung mit Amazon-EKS-Add-Ons überschreiben. Weitere Informationen zu the section called “kube-proxy” und the section called “CoreDNS” finden Sie in der EKS-Dokumentation. Wenn Sie einen Cluster im gemischten Modus mit Hybridknoten und Knoten in der AWS Cloud ausführen, AWS empfiehlt es sich, mindestens ein CoreDNS-Replikat auf Hybridknoten und mindestens ein CoreDNS-Replikat auf Ihren Knoten in der Cloud zu haben. AWS Konfigurationsschritte finden Sie unter the section called “CoreDNS-Replikate konfigurieren”.

CloudWatch Beobachtbarkeitsagent

Der CloudWatch Observability-Agent-Operator verwendet Webhooks. Wenn Sie den Operator auf Hybridknoten ausführen, muss Ihr On-Premises-Pod-CIDR in Ihrem On-Premises-Netzwerk routingfähig sein, und Sie müssen Ihren EKS-Cluster mit Ihrem Pod-Netzwerk aus der Ferne konfigurieren. Weitere Informationen finden Sie unter Konfiguration von Webhooks für Hybridknoten.

Metriken auf Knotenebene sind für Hybridknoten nicht verfügbar, da CloudWatch Container Insights von der Verfügbarkeit des Instance Metadata Service (IMDS) für Metriken auf Knotenebene abhängt. Für Hybridknoten sind Metriken auf Cluster-, Workload-, Pod- und Container-Ebene verfügbar.

Nachdem Sie das Add-on mithilfe der unter Installieren des CloudWatch Agenten mit Amazon CloudWatch Observability beschriebenen Schritte installiert haben, muss das Zusatz-Manifest aktualisiert werden, bevor der Agent erfolgreich auf Hybridknoten ausgeführt werden kann. Bearbeiten Sie die `amazoncloudwatchagents`-Ressource im Cluster, um die `RUN_WITH_IRSA`-Umgebungsvariable wie unten angezeigt hinzuzufügen.

```
kubectl edit amazoncloudwatchagents -n amazon-cloudwatch cloudwatch-agent
```

```
apiVersion: v1
items:
- apiVersion: cloudwatch.aws.amazon.com/v1alpha1
  kind: AmazonCloudWatchAgent
```

```
  metadata:
    ...
    name: cloudwatch-agent
    namespace: amazon-cloudwatch
    ...
  spec:
    ...
    env:
    - name: RUN_WITH_IRSA # <-- Add this
      value: "True" # <-- Add this
    - name: K8S_NODE_NAME
      valueFrom:
        fieldRef:
          fieldPath: spec.nodeName
          ...
```

Amazon Managed Prometheus Managed Collector für Hybridknoten

Ein verwalteter Kollector von Amazon Managed Service für Prometheus (AMP) besteht aus einem Scraper, der Metriken aus den Ressourcen in einem Amazon-EKS-Cluster erkennt und erfasst. AMP verwaltet den Scraper für Sie, sodass Sie sich nicht selbst um Instances, Agenten oder Scraper kümmern müssen.

Sie können die von AMP verwalteten Kollektoren ohne zusätzliche Konfiguration für Hybridknoten verwenden. Die metrischen Endpunkte für Ihre Anwendungen auf den Hybridknoten müssen jedoch von der VPC aus erreichbar sein, einschließlich der Routen von der VPC zum Remote-Pod-Netzwerk CIDRs und der offenen Ports in Ihrer lokalen Firewall. Darüber hinaus muss Ihr Cluster über einen privaten Cluster-Endpunktzugriff verfügen.

Folgen Sie den Schritten unter Verwenden eines AWS verwalteten Collectors im Amazon Managed Service for Prometheus Benutzerhandbuch.

AWS Distribution für OpenTelemetry (ADOT)

Sie können das Add-on AWS Distro for OpenTelemetry (ADOT) verwenden, um Metriken, Protokolle und Ablaufverfolgungsdaten von Ihren Anwendungen zu sammeln, die auf Hybridknoten ausgeführt werden. ADOT verwendet Zulassungs-Webhooks, um die Anfragen für benutzerdefinierte Kollektor-Ressourcen zu verändern und zu validieren. Wenn Sie den ADOT-Operator in Hybridknoten ausführen, muss Ihr On-Premises-Pod-CIDR in Ihrem On-Premises-Netzwerk routingfähig sein und Sie müssen Ihren EKS-Cluster mit Ihrem dezentralen Pod-Netzwerk konfigurieren. Weitere Informationen finden Sie unter Konfiguration von Webhooks für Hybridknoten.

Folgen Sie den Schritten unter Erste Schritte mit AWS Distro zur OpenTelemetry Verwendung von EKS-Add-Ons in der AWS Dokumentation von Distro. OpenTelemetry

AWS Load Balancer Balancer-Controller

Sie können den Load AWS Balancer Controller und den Application Load Balancer (ALB) oder den Network Load Balancer (NLB) mit dem Zieltyp `ip` für Workloads auf Hybridknoten verwenden. Die mit dem ALB oder NLB verwendeten IP-Ziele müssen routbar sein. AWS Der AWS Load Balancer Balancer-Controller verwendet auch Webhooks. Wenn Sie den Load AWS Balancer Controller-Operator auf Hybridknoten ausführen, muss Ihr lokales Pod-CIDR in Ihrem lokalen Netzwerk routingfähig sein und Sie müssen Ihren EKS-Cluster mit Ihrem Remote-Pod-Netzwerk konfigurieren. Weitere Informationen finden Sie unter Konfiguration von Webhooks für Hybridknoten.

Um den Load AWS Balancer Controller zu installieren, folgen Sie den Schritten unter the section called “AWS Application Load Balancer” oderthe section called “AWS Network Load Balancer”.

Für den Eingang mit ALB müssen Sie die folgenden Annotationen angeben. Weitere Informationen finden Sie unter the section called “Application Load Balancing”.

```
alb.ingress.kubernetes.io/target-type: ip
```

Für den Lastenausgleich mit NLB müssen Sie die folgenden Annotationen angeben. Weitere Informationen finden Sie unter the section called “Netzwerk-Load-Balancer”.

```
service.beta.kubernetes.io/aws-load-balancer-type: "external"
service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: "ip"
```

EKS Pod Identity Agent

> **Note**
>
> Um das Add-On für EKS Pod Identity Agent erfolgreich in Hybridknoten mit Bottlerocket bereitzustellen, stellen Sie sicher, dass Ihre Bottlerocket-Version mindestens v1.39.0 ist. Der Pod-Identity-Agent wird in früheren Bottlerocket-Versionen in Hybridknoten-Umgebungen nicht unterstützt.

Der ursprüngliche Amazon EKS Pod Identity Agent DaemonSet ist auf die Verfügbarkeit von EC2 IMDS auf dem Knoten angewiesen, um die erforderlichen AWS Anmeldeinformationen zu erhalten. Da IMDS ab Version 1.3.3-eksbuild.1 auf Hybridknoten nicht verfügbar ist, stellt das Pod Identity

Agent-Add-on optional einen bereit, der die erforderlichen Anmeldeinformationen bereitstellt. DaemonSet Hybridknoten, auf denen Bottlerocket ausgeführt wird, benötigen eine andere Methode zum Mounten der Anmeldeinformationen. Ab Version 1.3.7-eksbuild.2 stellt das Pod Identity Agent-Add-on optional eine bereit, die speziell auf Bottlerocket-Hybridknoten abzielt. DaemonSet In den folgenden Abschnitten wird DaemonSets der Prozess zur Aktivierung der optionalen Option beschrieben.

Ubuntu/RHEL/AL2023

1. Um den Pod Identity-Agenten auf Ubuntu/RHEL/Al 2023 Hybridknoten zu verwenden, legen Sie `enableCredentialsFile: true` im Hybrid-Abschnitt der `nodeadm` Konfiguration wie unten gezeigt fest:

   ```
   apiVersion: node.eks.aws/v1alpha1
   kind: NodeConfig
   spec:
       hybrid:
           enableCredentialsFile: true # <-- Add this
   ```

   Dadurch wird `nodeadm` so konfiguriert, dass eine Anmeldeinformationsdatei erstellt wird, die auf dem Knoten unter `/eks-hybrid/.aws/credentials` konfiguriert wird und von `eks-pod-identity-agent`-Pods verwendet wird. Diese Anmeldeinformationsdatei enthält temporäre AWS Anmeldeinformationen, die regelmäßig aktualisiert werden.

2. Nachdem Sie die `nodeadm`-Konfiguration auf jedem Knoten aktualisiert haben, führen Sie den folgenden `nodeadm init`-Befehl mit Ihrem `nodeConfig.yaml` aus, um Ihre Hybridknoten mit Ihrem Amazon-EKS-Cluster zu verbinden. Wenn Ihre Knoten dem Cluster bereits zuvor beigetreten sind, führen Sie den `nodeadm init`-Befehl dennoch erneut aus.

   ```
   nodeadm init -c file://nodeConfig.yaml
   ```

3. Installieren Sie `eks-pod-identity-agent` mit aktivierter Unterstützung für Hybridknoten, entweder über die AWS CLI oder AWS-Managementkonsole.

   a. AWS CLI: Führen Sie auf dem Computer, den Sie zur Verwaltung des Clusters verwenden, den folgenden Befehl aus, um die Installation `eks-pod-identity-agent` mit aktivierter Unterstützung für Hybridknoten zu installieren. Ersetzen Sie `my-cluster` mit dem Namen Ihres Clusters.

      ```
      aws eks create-addon \
      ```

```
    --cluster-name my-cluster \
    --addon-name eks-pod-identity-agent \
    --configuration-values '{"daemonsets":{"hybrid":{"create": true}}}'
```

b. AWS-Managementkonsole: Wenn Sie das Pod Identity Agent-Add-on über die AWS Konsole installieren, fügen Sie der optionalen Konfiguration Folgendes hinzu, um DaemonSet das Add-on für Hybridknoten bereitzustellen.

```
{"daemonsets":{"hybrid":{"create": true}}}
```

Bottlerocket

1. Um den Pod-Identity-Agenten auf Bottlerocket-Hybridknoten zu verwenden, fügen Sie dem Befehl, der für die Benutzerdaten des Bottlerocket-Bootstrap-Containers verwendet wird, das `--enable-credentials-file=true`-Flag hinzu, wie in the section called “Hybridknoten mit Bottlerocket verbinden” beschrieben.

   a. Wenn Sie den SSM-Anmeldeinformations-Anbieter verwenden, sollte Ihr Befehl wie folgt aussehen:

   ```
   eks-hybrid-ssm-setup --activation-id=<activation-id> --activation-
   code=<activation-code> --region=<region> --enable-credentials-file=true
   ```

   b. Wenn Sie den Anmeldeinformationsanbieter IAM Roles Anywhere verwenden, sollte Ihr Befehl folgendermaßen aussehen:

   ```
   eks-hybrid-iam-ra-setup --certificate=<certificate> --key=<private-key> --enable-
   credentials-file=true
   ```

   Dadurch wird das Bootstrap-Skript so konfiguriert, dass auf dem Knoten unter `/var/eks-hybrid/.aws/credentials` eine Anmeldeinformationsdatei erstellt wird, die von `eks-pod-identity-agent`-Pods verwendet wird. Diese Anmeldeinformationsdatei enthält temporäre AWS Anmeldeinformationen, die regelmäßig aktualisiert werden.

2. Installieren Sie `eks-pod-identity-agent` mit aktivierter Unterstützung für Bottlerocket-Hybridknoten, entweder über die AWS CLI oder. AWS-Managementkonsole

   a. AWS CLI: Führen Sie auf dem Computer, den Sie zur Verwaltung des Clusters verwenden, den folgenden Befehl aus, um die Installation `eks-pod-identity-agent` mit aktivierter

Unterstützung für Bottlerocket-Hybridknoten zu installieren. Ersetzen Sie `my-cluster` mit dem Namen Ihres Clusters.

```
aws eks create-addon \
    --cluster-name my-cluster \
    --addon-name eks-pod-identity-agent \
    --configuration-values '{"daemonsets":{"hybrid-bottlerocket":{"create":
 true}}}'
```

b. AWS-Managementkonsole: Wenn Sie das Pod Identity Agent-Add-On über die AWS Konsole installieren, fügen Sie der optionalen Konfiguration Folgendes hinzu, um die auf Bottlerocket-Hybridknoten DaemonSet abzielenden Pakete bereitzustellen.

```
{"daemonsets":{"hybrid-bottlerocket":{"create": true}}}
```

CSI-Snapshot-Controller

Ab Version `v8.1.0-eksbuild.2` wendet das CSI-Snapshot-Controller-Add-on eine Soft-Anti-Affinitätsregel für Hybridknoten an und bevorzugt es, `deployment` dass der Controller EC2 in derselben AWS Region wie die Amazon EKS-Steuerungsebene ausgeführt wird. Die gemeinsame Platzierung `deployment` in derselben AWS Region wie die Amazon EKS-Steuerebene verbessert die Latenz.

Community-Erweiterungen

In den folgenden Abschnitten werden die Unterschiede zwischen der Ausführung kompatibler Community-Add-Ons in Hybridknoten und anderen Amazon-EKS-Rechentypen beschrieben.

Kubernetes-Metriken-Server

Die Steuerebene muss die Pod-IP des Metriken-Servers erreichen (oder die Knoten-IP, wenn hostNetwork aktiviert ist). Wenn Sie den Metriken-Server nicht im hostNetwork-Modus ausführen, müssen Sie daher beim Erstellen Ihres Amazon-EKS-Clusters ein Pod-Netzwerk aus der Ferne konfigurieren und Ihre Pod-IP-Adressen routingfähig machen. Die Implementierung des Border Gateway Protocol (BGP) mit dem CNI ist eine gängige Methode, um die IP-Adressen Ihrer Pods routingfähig zu machen.

cert-manager

`cert-manager` verwendet Webhooks. Wenn Sie `cert-manager` in Hybridknoten ausführen, muss Ihr On-Premises-Pod-CIDR in Ihrem On-Premises-Netzwerk routingfähig sein und Sie müssen Ihren EKS-Cluster mit Ihrem Pod-Netzwerk aus der Ferne konfigurieren. Weitere Informationen finden Sie unter Konfiguration von Webhooks für Hybridknoten.

Konfiguration von Webhooks für Hybridknoten

Auf dieser Seite finden Sie detaillierte Überlegungen zum Ausführen von Webhooks mit Hybridknoten. Webhooks werden in Kubernetes-Anwendungen und Open-Source-Projekten wie dem AWS Load Balancer Controller und dem CloudWatch-Beobachtbarkeits-Agenten verwendet, um Mutations- und Validierungsfunktionen zur Laufzeit auszuführen.

Routingfähige Pod-Netzwerke

Wenn Sie Ihr lokales Pod-CIDR in Ihrem On-Premises-Netzwerk routingfähig machen können, können Sie Webhooks in Hybridknoten ausführen Es gibt mehrere Techniken, mit denen Sie Ihren On-Premises-Pod CIDR in Ihrem On-Premises-Netzwerk routingfähig machen können, darunter Border Gateway Protocol (BGP), statische Routen oder andere benutzerdefinierte Routing-Lösungen. BGP ist die empfohlene Lösung, da es skalierbarer und einfacher zu verwalten ist als alternative Lösungen, die eine benutzerdefinierte oder manuelle Routing-Konfiguration erfordern. AWS unterstützt die BGP-Funktionen von Cilium und Calico für die Bekanntmachung für Pod-CIDRs. Weitere Informationen finden Sie unter the section called “CNI konfigurieren” und the section called “Routingfähige Fern-Pod-CIDRs”.

Nicht routingfähige Pod-Netzwerke

Wenn Sie Ihr On-Premises-Pod-CIDR in Ihrem On-Premises-Netzwerk nicht routingfähig machen können und Webhooks ausführen müssen, empfehlen wir Ihnen, alle Webhooks auf Cloud-Knoten im selben EKS-Cluster wie Ihre Hybridknoten auszuführen.

Überlegungen zu Clustern im kombinierten Modus

Cluster im kombinierten Modus sind EKS-Cluster, die sowohl Hybridknoten als auch Knoten umfassen, die in der AWS Cloud ausgeführt werden. Beachten Sie bei der Ausführung eines Clusters im kombinierten Modus die folgenden Empfehlungen:

- Führen Sie das VPC CNI auf Knoten in der AWS Cloud und entweder Cilium oder Calico auf Hybridknoten aus. Cilium und Calico werden von AWS nicht unterstützt, wenn sie auf Knoten in der AWS Cloud ausgeführt werden.

- Konfigurieren Sie Webhooks für die Ausführung auf Knoten in der AWS Cloud. Informationen zur Konfiguration der Webhooks für AWS und Community-Add-Ons finden Sie unter the section called “Webhooks für Add-Ons konfigurieren”.
- Wenn Ihre Anwendungen erfordern, dass Pods, die auf Knoten in der AWS Cloud ausgeführt werden, direkt mit Pods kommunizieren, die auf Hybridknoten ausgeführt werden („Ost-West-Kommunikation“), und Sie das VPC CNI auf Knoten in der AWS Cloud und Cilium oder Calico auf Hybridknoten verwenden, muss Ihr On-Premises-Pod-CIDR in Ihrem On-Premises-Netzwerk routingfähig sein.
- Führen Sie mindestens eine Replik von CoreDNS in Knoten in der AWS Cloud und mindestens eine Replik von CoreDNS in Hybridknoten aus.
- Konfigurieren Sie die Verteilung des Service-Datenverkehrs, um den Service-Datenverkehr lokal in der Zone zu halten, aus der er stammt. Weitere Informationen zur Verteilung des Service-Datenverkehrs finden Sie unter the section called “Verteilung des Service-Datenverkehrs konfigurieren”.
- Wenn Sie AWS Application Load Balancer (ALB) oder Network Load Balancer (NLB) für den Workload-Datenverkehr auf Hybridknoten verwenden, müssen die mit ALB oder NLB verwendeten IP-Ziele von AWS aus routingfähig sein
- Das Metriken-Server-Add-On erfordert eine Verbindung von der EKS-Steuerebene zur IP-Adresse des Metriken-Server-Pods. Wenn Sie das Metriken-Server-Add-On in Hybridknoten ausführen, muss Ihr lokaler Pod-CIDR in Ihrem On-Premises-Netzwerk routingfähig sein.
- Um Metriken für Hybridknoten mithilfe von Amazon Managed Service für Prometheus (AMP) verwalteten Kollektoren zu erfassen, muss Ihr On-Premises-Pod-CIDR in Ihrem On-Premises-Netzwerk routingfähig sein. Alternativ können Sie den AMP Managed Collector für EKS-Steuerebenen-Metriken und in der AWS Cloud ausgeführte Ressourcen sowie das AWS Distro for OpenTelemetry (ADOT)-Add-On verwenden, um Metriken für Hybridknoten zu erfassen.

Cluster im kombinierten Modus konfigurieren

Um die mutierenden und validierenden Webhooks anzuzeigen, die auf Ihrem Cluster ausgeführt werden, können Sie den Ressourcentyp Erweiterungen im Ressourcenbereich der EKS-Konsole für Ihren Cluster anzeigen oder die folgenden Befehle verwenden. EKS meldet auch Webhook-Metriken im Cluster-Beobachtbarkeits-Dashboard. Weitere Informationen finden Sie unter the section called “Dashboard zur Beobachtbarkeit”.

```
kubectl get mutatingwebhookconfigurations
```

```
kubectl get validatingwebhookconfigurations
```

Verteilung des Service-Datenverkehrs konfigurieren

Bei der Ausführung von Clustern im kombinierten Modus empfehlen wir die Verwendung der Verteilung des Service-Datenverkehrs, um den Service-Datenverkehr lokal in der Zone zu halten, aus der er stammt. Die Verteilung des Service-Datenverkehrs (verfügbar für Kubernetes-Versionen 1.31 und höher in EKS) wird gegenüber dem topologiebewussten Routing als Lösung empfohlen, da sie vorhersehbarer ist. Mit der Verteilung des Service-Datenverkehrs erhalten fehlerfreie Endpunkte in der Zone den gesamten Datenverkehr für diese Zone. Bei Topologiebewusstem Routing muss jeder Service mehrere Bedingungen in dieser Zone erfüllen, um das benutzerdefinierte Routing anzuwenden. Andernfalls wird der Datenverkehr gleichmäßig auf alle Endpunkte verteilt.

Wenn Sie Cilium als CNI verwenden, müssen Sie das CNI mit dem Wert `enable-service-topology` auf `true` ausführen, um die Verteilung des Service-Datenverkehrs zu aktivieren. Sie können diese Konfiguration mit dem Helm-Installations-Flag `--set loadBalancer.serviceTopology=true` übergeben oder eine vorhandene Installation mit dem Cilium-CLI-Befehl `cilium config set enable-service-topology true` aktualisieren. Der auf jedem Knoten ausgeführte Cilium-Agent muss nach der Aktualisierung der Konfiguration für eine vorhandene Installation neu gestartet werden.

Im folgenden Abschnitt wird ein Beispiel für die Konfiguration der Verteilung des Service-Datenverkehrs für den CoreDNS-Service gezeigt. Wir empfehlen, dies für alle Services in Ihrem Cluster zu aktivieren, um unbeabsichtigten Datenverkehr zwischen Umgebungen zu vermeiden.

CoreDNS-Replikate konfigurieren

Wenn Sie einen Cluster im kombinierten Modus mit Hybridknoten und Knoten in der AWS Cloud ausführen, empfehlen wir Ihnen, mindestens eine CoreDNS-Replik auf Hybridknoten und mindestens eine CoreDNS-Replik auf Ihren Knoten in der AWS Cloud zu haben. Um Latenz- und Netzwerkprobleme in einer Cluster-Konfiguration im kombinierten Modus zu vermeiden, können Sie den CoreDNS-Service so konfigurieren, dass er die nächstgelegene CoreDNS-Replik mit der Verteilung des Service-Datenverkehrs bevorzugt.

Die Verteilung des Service-Datenverkehrs (verfügbar für Kubernetes-Versionen 1.31 und höher in EKS) wird gegenüber dem topologiebewussten Routing als Lösung empfohlen, da sie vorhersehbarer ist. Bei der Verteilung des Service-Datenverkehrs erhalten alle fehlerfreien Endpunkte in der Zone den gesamten Datenverkehr für diese Zone. Beim topologiebewussten Routing muss jeder Service mehrere Bedingungen in dieser Zone erfüllen, um das benutzerdefinierte Routing anzuwenden.

Andernfalls wird der Datenverkehr gleichmäßig auf alle Endpunkte verteilt. Die folgenden Schritte konfigurieren die Verteilung des Service-Datenverkehrs.

Wenn Sie Cilium als CNI verwenden, müssen Sie das CNI mit dem Wert `enable-service-topology` auf `true` ausführen, um die Verteilung des Service-Datenverkehrs zu aktivieren. Sie können diese Konfiguration mit dem Helm-Installations-Flag `--set loadBalancer.serviceTopology=true` übergeben oder eine vorhandene Installation mit dem Cilium-CLI-Befehl `cilium config set enable-service-topology true` aktualisieren. Der auf jedem Knoten ausgeführte Cilium-Agent muss nach der Aktualisierung der Konfiguration für eine vorhandene Installation neu gestartet werden.

1. Fügen Sie für jeden Ihrer Hybridknoten eine Topologie-Zonenbezeichnung hinzu, beispielsweise `topology.kubernetes.io/zone: onprem`. Alternativ können Sie die Bezeichnung in der `nodeadm init`-Phase festlegen, indem Sie die Bezeichnung in Ihrer `nodeadm`-Konfiguration angeben, siehe the section called “Knotenkonfiguration zur Anpassung von kubelet (optional)”. Beachten Sie, dass Knoten, die in der AWS Cloud ausgeführt werden, automatisch eine Topologie-Zonenbezeichnung erhalten, die der Availability Zone (AZ) des Knotens entspricht.

   ```
   kubectl label node hybrid-node-name topology.kubernetes.io/zone=zone
   ```

2. Fügen Sie `podAntiAffinity` mit dem Topologie-Zonenschlüssel zur CoreDNS-Bereitstellung hinzu. Alternativ können Sie die CoreDNS-Bereitstellung während der Installation mit EKS-Add-Ons konfigurieren.

   ```
   kubectl edit deployment coredns -n kube-system
   ```

   ```
   spec:
     template:
       spec:
         affinity:
          ...
           podAntiAffinity:
             preferredDuringSchedulingIgnoredDuringExecution:
             - podAffinityTerm:
                 labelSelector:
                   matchExpressions:
                   - key: k8s-app
                     operator: In
                     values:
   ```

```
                - kube-dns
            topologyKey: kubernetes.io/hostname
          weight: 100
        - podAffinityTerm:
            labelSelector:
              matchExpressions:
              - key: k8s-app
                operator: In
                values:
                - kube-dns
            topologyKey: topology.kubernetes.io/zone
          weight: 50
    ...
```

3. Fügen Sie die Einstellung `trafficDistribution: PreferClose` zur Service-Konfiguration `kube-dns` hinzu, um die Verteilung des Service-Datenverkehrs zu aktivieren.

```
kubectl patch svc kube-dns -n kube-system --type=merge -p '{
  "spec": {
    "trafficDistribution": "PreferClose"
  }
}'
```

4. Sie können überprüfen, ob die Verteilung des Service-Datenverkehrs aktiviert ist, indem Sie die Endpunkt-Segmente für den `kube-dns`-Service anzeigen. Ihre Endpunkt-Segmente müssen die `hints` für Ihre Topologie-Zonenbezeichnungen anzeigen, was bestätigt, dass die Verteilung des Service-Datenverkehrs aktiviert ist. Wenn Sie für jede Endpunktadresse kein `hints` sehen, ist die Verteilung des Service-Datenverkehrs nicht aktiviert.

```
kubectl get endpointslice -A | grep "kube-dns"
```

```
kubectl get endpointslice [.replaceable]`kube-dns-<id>`  -n kube-system -o yaml
```

```
addressType: IPv4
apiVersion: discovery.k8s.io/v1
endpoints:
- addresses:
  - <your-hybrid-node-pod-ip>
  hints:
    forZones:
    - name: onprem
```

```
  nodeName: <your-hybrid-node-name>
  zone: onprem
- addresses:
  - <your-cloud-node-pod-ip>
  hints:
    forZones:
    - name: us-west-2a
  nodeName: <your-cloud-node-name>
  zone: us-west-2a
```

Webhooks für Add-Ons konfigurieren

Die folgenden Add-Ons verwenden Webhooks und werden für die Verwendung mit Hybridknoten unterstützt.

- AWS-Lastenverteilungs-Controller
- CloudWatch-Beobachtbarkeits-Agent
- AWS Distro for OpenTelemetry (ADOT)
- `cert-manager`

Informationen zum Konfigurieren der von diesen Add-Ons verwendeten Webhooks für die Ausführung auf Knoten in der AWS Cloud finden Sie in den folgenden Abschnitten.

AWS-Lastenverteilungs-Controller

Um den AWS Load Balancer Controller in einer Cluster-Konfiguration im kombinierten Modus zu verwenden, ist es erforderlich, den Controller auf Knoten in der AWS Cloud auszuführen. Fügen Sie dazu Folgendes zu Ihrer Helm-Wertekonfiguration hinzu oder geben Sie die Werte mithilfe der EKS-Add-On-Konfiguration an.

```
affinity:
  nodeAffinity:
    requiredDuringSchedulingIgnoredDuringExecution:
      nodeSelectorTerms:
      - matchExpressions:
        - key: eks.amazonaws.com/compute-type
          operator: NotIn
          values:
          - hybrid
```

CloudWatch-Beobachtbarkeits-Agent

Das Add-On für den CloudWatch-Beobachtbarkeits-Agent verfügt über einen Kubernetes-Operator, der Webhooks verwendet. Um den Operator auf Knoten in der AWS Cloud in einer Cluster-Konfiguration im kombinierten Modus auszuführen, bearbeiten Sie die Konfiguration des Operators für den CloudWatch-Beobachtbarkeits-Agenten. Sie können die Operator-Affinität während der Installation mit Helm- und EKS-Add-Ons nicht konfigurieren (siehe Container-Roadmap-Problem Nr. 2431).

```
kubectl edit -n amazon-cloudwatch deployment amazon-cloudwatch-observability-
controller-manager
```

```
spec:
  ...
  template:
    ...
    spec:
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
              - key: eks.amazonaws.com/compute-type
                operator: NotIn
                values:
                - hybrid
```

AWS Distro for OpenTelemetry (ADOT)

Das AWS Distro for OpenTelemetry (ADOT)-Add-On verfügt über einen Kubernetes-Operator, der Webhooks verwendet. Um den Operator auf Knoten in der AWS Cloud in einer Cluster-Konfiguration im kombinierten Modus auszuführen, fügen Sie Folgendes zu Ihrer Helm-Wertekonfiguration hinzu oder geben Sie die Werte mithilfe der EKS-Add-On-Konfiguration an.

```
affinity:
  nodeAffinity:
    requiredDuringSchedulingIgnoredDuringExecution:
      nodeSelectorTerms:
      - matchExpressions:
        - key: eks.amazonaws.com/compute-type
          operator: NotIn
```

```
          values:
          - hybrid
```

Wenn Ihr Pod-CIDR in Ihrem On-Premises-Netzwerk nicht routingfähig ist, muss der ADOT-Kollektor in Hybridknoten ausgeführt werden, um die Metriken von Ihren Hybridknoten und den darauf ausgeführten Workloads zu erfassen. Bearbeiten Sie dazu die benutzerdefinierte Ressourcendefinition (CRD).

```
kubectl -n opentelemetry-operator-system edit opentelemetrycollectors.opentelemetry.io
 adot-col-prom-metrics
```

```
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: eks.amazonaws.com/compute-type
            operator: In
            values:
            - hybrid
```

Sie können den ADOT-Kollektor so konfigurieren, dass er nur Metriken von Hybridknoten und den auf Hybridknoten ausgeführten Ressourcen erfasst, indem Sie das folgende `relabel_configs` zu jedem `scrape_configs` in der ADOT-Kollektor-CRD-Konfiguration hinzufügen.

```
relabel_configs:
  - action: keep
    regex: hybrid
    source_labels:
    - __meta_kubernetes_node_label_eks_amazonaws_com_compute_type
```

Das ADOT-Add-on erfordert die Installation von `cert-manager` für die vom ADOT-Operator-Webhook verwendeten TLS-Zertifikate. `cert-manager` führt ebenfalls Webhooks aus und kann mit der folgenden Helm-Wertekonfiguration für die Ausführung auf Knoten in der AWS Cloud konfiguriert werden.

```
affinity:
  nodeAffinity:
    requiredDuringSchedulingIgnoredDuringExecution:
```

```
      nodeSelectorTerms:
      - matchExpressions:
        - key: eks.amazonaws.com/compute-type
          operator: NotIn
          values:
          - hybrid
webhook:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: eks.amazonaws.com/compute-type
            operator: NotIn
            values:
            - hybrid
cainjector:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: eks.amazonaws.com/compute-type
            operator: NotIn
            values:
            - hybrid
startupapicheck:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: eks.amazonaws.com/compute-type
            operator: NotIn
            values:
            - hybrid
```

`cert-manager`

Das `cert-manager`-Add-On führt Webhooks aus und Sie können es mit der folgenden Helm-Wertekonfiguration so konfigurieren, dass es auf Knoten in der AWS Cloud ausgeführt wird.

```
affinity:
```

```
  nodeAffinity:
    requiredDuringSchedulingIgnoredDuringExecution:
      nodeSelectorTerms:
      - matchExpressions:
        - key: eks.amazonaws.com/compute-type
          operator: NotIn
          values:
          - hybrid
webhook:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: eks.amazonaws.com/compute-type
            operator: NotIn
            values:
            - hybrid
cainjector:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: eks.amazonaws.com/compute-type
            operator: NotIn
            values:
            - hybrid
startupapicheck:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: eks.amazonaws.com/compute-type
            operator: NotIn
            values:
            - hybrid
```

Konfiguration des Proxys für Hybridknoten

Wenn Sie in Ihrer On-Premises-Umgebung mithilfe eines Proxy-Servers den Datenverkehr steuern, der Ihr Rechenzentrum oder Ihre Edge-Umgebung verlässt, müssen Sie Ihre Knoten und Ihren Cluster separat für die Verwendung Ihres Proxy-Servers konfigurieren.

Cluster

In Ihrem Cluster müssen Sie `kube-proxy` so konfigurieren, dass Ihr Proxy-Server verwendet wird. Sie müssen `kube-proxy` nach der Erstellung Ihres Amazon-EKS-Clusters konfigurieren.

Knoten

Auf Ihren Knoten müssen Sie das Betriebssystem, `containerd`, `kubelet` und den Amazon SSM-Agenten so konfigurieren, dass sie Ihren Proxy-Server verwenden. Sie können diese Änderungen während des Entwicklungsprozesses für Ihre Betriebssystem-Images oder vor der Ausführung von `nodeadm init` auf jedem Hybridknoten vornehmen.

Konfiguration auf Knotenebene

Sie müssen die folgenden Konfigurationen entweder in Ihren Betriebssystem-Images oder vor der Ausführung von `nodeadm init` auf jedem Hybridknoten anwenden.

`containerd`-Proxy-Konfiguration

`containerd` ist die standardmäßig Container-Management-Laufzeitumgebung für Kubernetes. Wenn Sie einen Proxy für den Internetzugriff verwenden, müssen Sie `containerd` so konfigurieren, dass es die von Kubernetes und Amazon EKS benötigten Container-Images abrufen kann.

Erstellen Sie auf jedem Hybridknoten eine `/etc/systemd/system/containerd.service.d`-Datei mit dem Namen `http-proxy.conf` im Verzeichnis mit dem folgenden Inhalt. Ersetzen Sie `proxy-domain` und `port` durch die Werte für Ihre Umgebung.

```
[Service]
Environment="HTTP_PROXY=http://proxy-domain:port"
Environment="HTTPS_PROXY=http://proxy-domain:port"
Environment="NO_PROXY=localhost"
```

`containerd`-Konfiguration aus Benutzerdaten

Für diese Datei muss das `containerd.service.d`-Verzeichnis erstellt werden. Sie müssen systemd neu laden, um die Konfigurationsdatei ohne Neustart zu übernehmen. In AL2023 wird der

Service wahrscheinlich bereits ausgeführt, wenn Ihr Skript ausgeführt wird, sodass Sie ihn auch neu starten müssen.

```
mkdir -p /etc/systemd/system/containerd.service.d
echo '[Service]' > /etc/systemd/system/containerd.service.d/http-proxy.conf
echo 'Environment="HTTP_PROXY=http://proxy-domain:port"' >> /etc/systemd/system/
containerd.service.d/http-proxy.conf
echo 'Environment="HTTPS_PROXY=http://proxy-domain:port"' >> /etc/systemd/system/
containerd.service.d/http-proxy.conf
echo 'Environment="NO_PROXY=localhost"' >> /etc/systemd/system/containerd.service.d/
http-proxy.conf
systemctl daemon-reload
systemctl restart containerd
```

`kubelet`-Proxy-Konfiguration

`kubelet` ist der Kubernetes-Knotenagent, der auf jedem Kubernetes-Knoten ausgeführt wird und für die Verwaltung des Knotens und der darauf ausgeführten Pods verantwortlich ist. Wenn Sie in Ihrer On-Premises-Umgebung einen Proxy verwenden, müssen Sie `kubelet` so konfigurieren, dass es mit den öffentlichen oder privaten Endpunkten Ihres Amazon-EKS-Clusters kommunizieren kann.

Erstellen Sie auf jedem Hybridknoten eine Datei mit dem Namen `http-proxy.conf` im `/etc/systemd/system/kubelet.service.d/`-Verzeichnis mit dem folgenden Inhalt. Ersetzen Sie `proxy-domain` und `port` durch die Werte für Ihre Umgebung.

```
[Service]
Environment="HTTP_PROXY=http://proxy-domain:port"
Environment="HTTPS_PROXY=http://proxy-domain:port"
Environment="NO_PROXY=localhost"
```

`kubelet`-Konfiguration aus Benutzerdaten

Für diese Datei muss das `kubelet.service.d`-Verzeichnis erstellt werden. Sie müssen systemd neu laden, um die Konfigurationsdatei ohne Neustart zu übernehmen. In AL2023 wird der Service wahrscheinlich bereits ausgeführt, wenn Ihr Skript ausgeführt wird, sodass Sie ihn auch neu starten müssen.

```
mkdir -p /etc/systemd/system/kubelet.service.d
echo '[Service]' > /etc/systemd/system/kubelet.service.d/http-proxy.conf
echo 'Environment="HTTP_PROXY=http://proxy-domain:port"' >> /etc/systemd/system/
kubelet.service.d/http-proxy.conf
```

```
echo 'Environment="HTTPS_PROXY=http://proxy-domain:port"' >> /etc/systemd/system/
kubelet.service.d/http-proxy.conf
echo 'Environment="NO_PROXY=localhost"' >> /etc/systemd/system/kubelet.service.d/http-
proxy.conf
systemctl daemon-reload
systemctl restart kubelet
```

`ssm`-Proxy-Konfiguration

`ssm` ist einer der Anmeldeinformationsanbieter, die zum Initialisieren eines Hybridknotens verwendet werden können. `ssm` ist für die Authentifizierung mit AWS und die Generierung temporärer Anmeldeinformationen verantwortlich, die von `kubelet` verwendet werden. Wenn Sie in Ihrer On-Premises-Umgebung einen Proxy verwenden und `ssm` als Anmeldeinformationsanbieter auf dem Knoten nutzen, müssen Sie `ssm` so konfigurieren, dass es mit Amazon-SSM-Service-Endpunkten kommunizieren kann.

Erstellen Sie auf jedem Hybridknoten eine Datei mit dem Namen `http-proxy.conf` im nachfolgenden Pfad, abhängig vom Betriebssystem

- Ubuntu – `/etc/systemd/system/snap.amazon-ssm-agent.amazon-ssm-agent.service.d/http-proxy.conf`
- Amazon Linux 2023 und Red Hat Enterprise Linux – `/etc/systemd/system/amazon-ssm-agent.service.d/http-proxy.conf`

Füllen Sie die Datei mit dem folgenden Inhalt. Ersetzen Sie `proxy-domain` und `port` durch die Werte für Ihre Umgebung.

```
[Service]
Environment="HTTP_PROXY=http://proxy-domain:port"
Environment="HTTPS_PROXY=http://proxy-domain:port"
Environment="NO_PROXY=localhost"
```

`ssm`-Konfiguration aus Benutzerdaten

Für diese Datei muss das `ssm`-systemd-Service-Dateiverzeichnis erstellt werden. Der Verzeichnispfad hängt vom verwendeten Betriebssystem des Knotens ab.

- Ubuntu – `/etc/systemd/system/snap.amazon-ssm-agent.amazon-ssm-agent.service.d`

- Amazon Linux 2023 und Red Hat Enterprise Linux – `/etc/systemd/system/amazon-ssm-agent.service.d`

systemd-Servicenamen im folgenden Neustartbefehl je nach dem auf dem Knoten verwendeten Betriebssystem ersetzen

- Ubuntu – `snap.amazon-ssm-agent.amazon-ssm-agent`
- Amazon Linux 2023 und Red Hat Enterprise Linux – `amazon-ssm-agent`

```
mkdir -p systemd-service-file-directory
echo '[Service]' > [.replaceable]#systemd-service-file-directory/http-proxy.conf
echo 'Environment="HTTP_PROXY=http://[.replaceable]#proxy-domain:port"' >> systemd-
service-file-directory/http-proxy.conf
echo 'Environment="HTTPS_PROXY=http://[.replaceable]#proxy-domain:port"' >>
 [.replaceable]#systemd-service-file-directory/http-proxy.conf
echo 'Environment="NO_PROXY=localhost"' >> [.replaceable]#systemd-service-file-
directory/http-proxy.conf
systemctl daemon-reload
systemctl restart [.replaceable]#systemd-service-name
```

Proxy-Konfiguration des Betriebssystems

Wenn Sie einen Proxy für den Internetzugang verwenden, müssen Sie Ihr Betriebssystem so konfigurieren, dass es die Abhängigkeiten der Hybridknoten aus dem Paket-Manager Ihres Betriebssystems abrufen kann.

Ubuntu

1. Konfigurieren Sie `snap` mit den folgenden Befehlen für die Verwendung Ihres Proxys:

   ```
   sudo snap set system proxy.https=http://proxy-domain:port
   sudo snap set system proxy.http=http://proxy-domain:port
   ```

2. Um den Proxy für `apt` zu aktivieren, erstellen Sie im `/etc/apt/`-Verzeichnis eine Datei mit dem Namen `apt.conf`. Ersetzen Sie Proxy-Domain und Port durch die entsprechenden Werte für Ihre Umgebung.

   ```
   Acquire::http::Proxy "http://proxy-domain:port";
   ```

```
Acquire::https::Proxy "http://proxy-domain:port";
```

Amazon Linux 2023

1. Konfigurieren Sie `dnf` für die Verwendung Ihres Proxys. Erstellen Sie eine Datei `/etc/dnf/dnf.conf` mit den Werten für die Proxy-Domain und den Port für Ihre Umgebung.

```
proxy=http://proxy-domain:port
```

Red Hat Enterprise Linux

1. Konfigurieren Sie `yum` für die Verwendung Ihres Proxys. Erstellen Sie eine Datei `/etc/yum.conf` mit den Werten für die Proxy-Domain und den Port für Ihre Umgebung.

```
proxy=http://proxy-domain:port
```

Proxy-Konfiguration von IAM Roles Anywhere

Der Service des Anmeldeinformationsanbieters für IAM Roles Anywhere ist für die Aktualisierung der Anmeldeinformationen verantwortlich, wenn IAM Roles Anywhere mit dem `enableCredentialsFile`-Flag verwendet wird (siehe the section called “EKS Pod Identity Agent”). Wenn Sie in Ihrer On-Premises-Umgebung einen Proxy verwenden, müssen Sie den Service so konfigurieren, dass er mit den Endpunkten von IAM Roles Anywhere kommunizieren kann.

Erstellen Sie im `/etc/systemd/system/aws_signing_helper_update.service.d/`-Verzeichnis eine Datei mit dem Namen `http-proxy.conf` mit folgendem Inhalt. Ersetzen Sie `proxy-domain` und `port` durch die Werte für Ihre Umgebung.

```
[Service]
Environment="HTTP_PROXY=http://proxy-domain:port"
Environment="HTTPS_PROXY=http://proxy-domain:port"
Environment="NO_PROXY=localhost"
```

Cluster-weite Konfiguration

Die Konfigurationen in diesem Abschnitt müssen angewendet werden, nachdem Sie Ihren Amazon-EKS-Cluster erstellt haben und bevor Sie `nodeadm init` auf jedem Hybridknoten ausführen.

kube-proxy-Proxy-Konfiguration

Amazon EKS installiert `kube-proxy` automatisch als DaemonSet auf jedem Hybridknoten, wenn Ihre Hybridknoten dem Cluster beitreten. `kube-proxy` ermöglicht das Routing über Services, die von Pods auf Amazon-EKS-Clustern unterstützt werden. Um jeden Host zu konfigurieren, erfordert `kube-proxy` eine DNS-Auflösung für Ihren Amazon-EKS-Cluster-Endpunkt.

1. `kube-proxy` DaemonSet mit dem folgenden Befehl bearbeiten

   ```
   kubectl -n kube-system edit ds kube-proxy
   ```

   Dadurch wird die `kube-proxy`-DaemonSet-Definition in Ihrem konfigurierten Editor geöffnet.

2. Fügen Sie die Umgebungsvariablen für `HTTP_PROXY` und `HTTPS_PROXY` hinzu. Beachten Sie, dass die `NODE_NAME`-Umgebungsvariable bereits in Ihrer Konfiguration vorhanden sein sollte. Ersetzen Sie `proxy-domain` und `port` durch Werte für Ihre Umgebung.

   ```
   containers:
     - command:
       - kube-proxy
       - --v=2
       - --config=/var/lib/kube-proxy-config/config - --hostname-override=$(NODE_NAME)
       env:
       - name: HTTP_PROXY
         value: http://proxy-domain:port
       - name: HTTPS_PROXY
         value: http://proxy-domain:port
       - name: NODE_NAME
         valueFrom:
           fieldRef:
             apiVersion: v1
             fieldPath: spec.nodeName
   ```

Konfiguration von Cilium BGP für Hybridknoten

In diesem Thema wird beschrieben, wie Sie Cilium Border Gateway Protocol (BGP) für Amazon EKS Hybrid Nodes konfigurieren. Die BGP-Funktionalität von Cilium heißt Cilium BGP Control Plane und kann verwendet werden, um Pod- und Serviceadressen in Ihrem On-Premises-Netzwerk bekannt zu geben. Alternative Methoden, um Pod-CIDRs in Ihrem On-Premises-Netzwerk routingfähig zu machen, finden Sie unter the section called “Routingfähige Fern-Pod-CIDRs”.

Cilium BGP konfigurieren

Voraussetzungen

- Cilium wurde gemäß den Anweisungen in the section called “CNI konfigurieren” installiert.

Verfahren

1. Um BGP mit Cilium zu verwenden und Pod- oder Service-Adressen in Ihrem On-Premises-Netzwerk bekannt zu geben, muss Cilium mit `bgpControlPlane.enabled: true` installiert sein. Wenn Sie BGP für eine vorhandene Cilium-Bereitstellung aktivieren, müssen Sie den Cilium-Operator neu starten, um die BGP-Konfiguration anzuwenden, falls BGP zuvor nicht aktiviert wurde. Sie können in Ihren Helm-Werten `operator.rollOutPods` auf `true` setzen, um den Cilium-Operator als Teil des Helm-Installations-/Upgrade-Prozesses neu zu starten.

```
helm upgrade cilium oci://public.ecr.aws/eks/cilium/cilium \
  --namespace kube-system \
  --reuse-values \
  --set operator.rollOutPods=true \
  --set bgpControlPlane.enabled=true
```

2. Bestätigen Sie, dass der Cilium-Operator und die Agenten neu gestartet wurden und ausgeführt werden.

```
kubectl -n kube-system get pods --selector=app.kubernetes.io/part-of=cilium
```

```
NAME                               READY   STATUS    RESTARTS   AGE
cilium-grwlc                       1/1     Running   0          4m12s
cilium-operator-68f7766967-5nnbl   1/1     Running   0          4m20s
cilium-operator-68f7766967-7spfz   1/1     Running   0          4m20s
cilium-pnxcv                       1/1     Running   0          6m29s
cilium-r7qkj                       1/1     Running   0          4m12s
```

```
cilium-wxhfn                       1/1     Running   0          4m1s
cilium-z7hlb                       1/1     Running   0          6m30s
```

3. Erstellen Sie eine Datei mit dem Namen `cilium-bgp-cluster.yaml` mit einer `CiliumBGPClusterConfig`-Definition. Möglicherweise müssen Sie die folgenden Informationen von Ihrem Netzwerkadministrator einholen.
   - Konfigurieren Sie `localASN` mit der ASN für die Knoten, auf denen Cilium ausgeführt wird.
   - Konfigurieren Sie `peerASN` mit der ASN für Ihren On-Premises-Router.
   - Konfigurieren Sie `peerAddress` mit der On-Premises-Router-IP, mit der jeder Knoten, auf dem Cilium ausgeführt wird, eine Peering-Verbindung herstellt.

   ```
   apiVersion: cilium.io/v2alpha1
   kind: CiliumBGPClusterConfig
   metadata:
     name: cilium-bgp
   spec:
     nodeSelector:
       matchExpressions:
       - key: eks.amazonaws.com/compute-type
         operator: In
         values:
         - hybrid
     bgpInstances:
     - name: "rack0"
       localASN: NODES_ASN
       peers:
       - name: "onprem-router"
         peerASN: ONPREM_ROUTER_ASN
         peerAddress: ONPREM_ROUTER_IP
         peerConfigRef:
           name: "cilium-peer"
   ```

4. Wenden Sie die Cilium-BGP-Cluster-Konfiguration auf Ihren Cluster an.

   ```
   kubectl apply -f cilium-bgp-cluster.yaml
   ```

5. Erstellen Sie eine Datei mit dem Namen `cilium-bgp-peer.yaml` mit der `CiliumBGPPeerConfig`-Ressource, die eine BGP-Peer-Konfiguration definiert. Mehrere Peers können dieselbe Konfiguration freigeben und auf die gemeinsame `CiliumBGPPeerConfig`-Ressource verweisen. Eine vollständige Liste der Konfigurationsoptionen finden Sie in der BGP-Peer-Konfiguration in der Cilium-Dokumentation.

Die Werte für die folgenden Cilium-Peer-Einstellungen müssen mit denen des On-Premises-Routers übereinstimmen, mit dem Sie das Peering durchführen.

- Konfigurieren Sie `holdTimeSeconds`, das festlegt, wie lange ein BGP-Peer auf eine Keepalive- oder Aktualisierungsnachricht wartet, bevor die Sitzung als unterbrochen erklärt wird. Der Standardwert ist 90 Sekunden.
- Konfigurieren Sie `keepAliveTimeSeconds`, um festzustellen, ob ein BGP-Peer noch erreichbar ist und die BGP-Sitzung aktiv ist. Standardmäßig ist ein Zeitraum von 30 Sekunden festgelegt.
- Konfigurieren Sie `restartTimeSeconds`, das die Zeit bestimmt, nach der die BGP-Steuerebene von Cilium die BGP-Sitzung nach einem Neustart wiederherstellen soll. Standardmäßig sind 120 Sekunden festgelegt.

```
apiVersion: cilium.io/v2alpha1
kind: CiliumBGPPeerConfig
metadata:
  name: cilium-peer
spec:
  timers:
    holdTimeSeconds: 90
    keepAliveTimeSeconds: 30
  gracefulRestart:
    enabled: true
    restartTimeSeconds: 120
  families:
    - afi: ipv4
      safi: unicast
      advertisements:
        matchLabels:
          advertise: "bgp"
```

6. Wenden Sie die Cilium BGP-Peer-Konfiguration auf Ihren Cluster an.

```
kubectl apply -f cilium-bgp-peer.yaml
```

7. Erstellen Sie eine Datei mit dem Namen `cilium-bgp-advertisement-pods.yaml` und einer `CiliumBGPAdvertisement`-Ressource, um die Pod-CIDRs in Ihrem On-Premises-Netzwerk bekannt zu machen.
   - Die `CiliumBGPAdvertisement`-Ressource wird verwendet, um Ankündigungstypen und die zugehörigen Attribute zu definieren. Im folgenden Beispiel wird Cilium so konfiguriert, dass nur

Pod-CIDRs bekannt gemacht werden. Weitere Informationen zum Konfigurieren von Cilium zum Bekanntgeben von Service-Adressen finden Sie in den Beispielen in the section called “Servicetyp LoadBalancer” und the section called “Cilium-Load-Balancing im Cluster”.

- Jeder Hybridknoten, auf dem der Cilium-Agent ausgeführt wird, ist mit dem Upstream-BGP-fähigen Router verbunden. Jeder Knoten gibt den ihm zugehörigen Pod-CIDR-Bereich bekannt, wenn Ciliums `advertisementType` wie im folgenden Beispiel auf `PodCIDR` gesetzt ist. Weitere Informationen finden Sie in der BGP-Anzeigenkonfiguration in der Cilium-Dokumentation.

```
apiVersion: cilium.io/v2alpha1
kind: CiliumBGPAdvertisement
metadata:
  name: bgp-advertisement-pods
  labels:
    advertise: bgp
spec:
  advertisements:
    - advertisementType: "PodCIDR"
```

8. Wenden Sie die Cilium BGP-Anzeigenkonfiguration auf Ihren Cluster an.

```
kubectl apply -f cilium-bgp-advertisement-pods.yaml
```

9. Sie können mit dem `cilium bgp peers`-Befehl überprüfen, ob das BGP-Peering mit der Cilium CLI funktioniert. In der Ausgabe sollten die korrekten Werte für Ihre Umgebung und der Sitzungsstatus `established` angezeigt werden. Weitere Informationen zur Fehlerbehebung finden Sie im Handbuch zur Fehlerbehebung und zum Betrieb in der Cilium-Dokumentation.

   In den folgenden Beispielen gibt es fünf Hybridknoten, in denen der Cilium-Agent ausgeführt wird, und jeder Knoten gibt den Pod-CIDR-Bereich bekannt, den er besitzt.

```
cilium bgp peers
```

```
Node                   Local AS    Peer AS               Peer Address        Session
 State   Uptime     Family         Received   Advertised
mi-026d6a261e355fba7   NODES_ASN
                  ONPREM_ROUTER_ASN
                  ONPREM_ROUTER_IP    established     1h18m58s   ipv4/unicast   1
      2
mi-082f73826a163626e   NODES_ASN
```

```
                  ONPREM_ROUTER_ASN
                  ONPREM_ROUTER_IP    established     1h19m12s   ipv4/unicast   1
      2
mi-09183e8a3d755abf6   NODES_ASN
                  ONPREM_ROUTER_ASN
                  ONPREM_ROUTER_IP    established     1h18m47s   ipv4/unicast   1
      2
mi-0d78d815980ed202d   NODES_ASN
                  ONPREM_ROUTER_ASN
                  ONPREM_ROUTER_IP    established     1h19m12s   ipv4/unicast   1
      2
mi-0daa253999fe92daa   NODES_ASN
                  ONPREM_ROUTER_ASN
                  ONPREM_ROUTER_IP    established     1h18m58s   ipv4/unicast   1
      2
```

```
cilium bgp routes
```

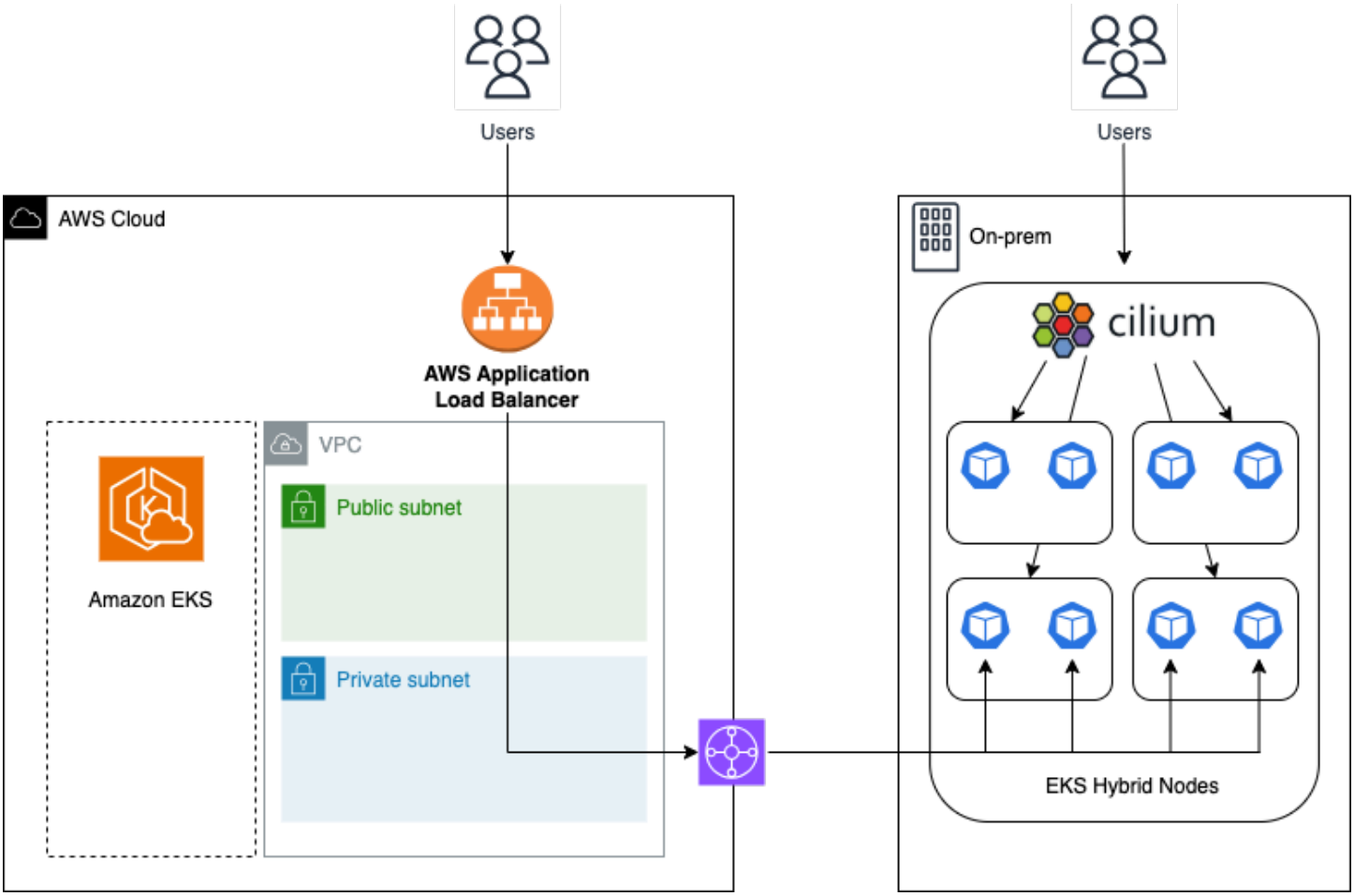
```
Node                   VRouter       Prefix           NextHop   Age          Attrs
mi-026d6a261e355fba7   NODES_ASN     10.86.2.0/26     0.0.0.0   1h16m46s   [{Origin:
 i} {Nexthop: 0.0.0.0}]
mi-082f73826a163626e   NODES_ASN     10.86.2.192/26   0.0.0.0   1h16m46s   [{Origin:
 i} {Nexthop: 0.0.0.0}]
mi-09183e8a3d755abf6   NODES_ASN     10.86.2.64/26    0.0.0.0   1h16m46s   [{Origin:
 i} {Nexthop: 0.0.0.0}]
mi-0d78d815980ed202d   NODES_ASN     10.86.2.128/26   0.0.0.0   1h16m46s   [{Origin:
 i} {Nexthop: 0.0.0.0}]
mi-0daa253999fe92daa   NODES_ASN     10.86.3.0/26     0.0.0.0   1h16m46s   [{Origin:
 i} {Nexthop: 0.0.0.0}]
```

Konfiguration von Kubernetes Ingress für Hybridknoten

In diesem Thema wird beschrieben, wie Sie Kubernetes Ingress für Workloads konfigurieren, die in Amazon EKS Hybrid Nodes ausgeführt werden. Kubernetes Ingress macht HTTP- und HTTPS-Routen von außerhalb des Clusters für Services innerhalb des Clusters verfügbar. Zur Nutzung von Ingress-Ressourcen ist ein Kubernetes-Ingress-Controller erforderlich, um die Netzwerkinfrastruktur und Komponenten für den Netzwerkverkehr einzurichten.

AWS unterstützt AWS Application Load Balancer (ALB) und Cilium für Kubernetes Ingress für Workloads, die in EKS-Hybridknoten ausgeführt werden. Die Entscheidung, ALB oder Cilium

für Ingress zu verwenden, hängt von der Quelle des Anwendungs-Datenverkehrs ab. Wenn der Anwendungsdatenverkehr aus einer AWS-Region stammt, empfiehlt AWS die Verwendung von AWS ALB und AWS Load Balancer Controller. Wenn der Anwendungs-Datenverkehr aus der lokalen On-Premises- oder Edge-Umgebung stammt, empfiehlt AWS die Verwendung der integrierten Ingress-Funktionen von Cilium, die mit oder ohne Load-Balancer-Infrastruktur in Ihrer Umgebung verwendet werden können.



AWS Application Load Balancer

Sie können den AWS Load Balancer Controller und den Application Load Balancer (ALB) mit dem Zieltyp `ip` für Workloads verwenden, die in Hybridknoten ausgeführt werden. Bei Verwendung des Zieltyps `ip` leitet ALB den Datenverkehr direkt an die Pods weiter und umgeht dabei den Netzwerkpfad der Service-Ebene. Damit ALB die Pod-IP-Ziele auf Hybridknoten erreichen kann, muss Ihr On-Premises-Pod-CIDR in Ihrem On-Premises-Netzwerk routingfähig sein. Darüber hinaus verwendet der AWS Load Balancer Controller Webhooks und erfordert eine direkte Kommunikation von der EKS-Steuerebene. Weitere Informationen finden Sie unter the section called “Webhooks konfigurieren”.

Überlegungen

- Weitere Informationen zu AWS Application Load Balancer und AWS Load Balancer Controller finden Sie unter the section called “Application Load Balancing” und the section called “Installation mit Helm”.
- Informationen zur Auswahl zwischen AWS Application Load Balancer und AWS Network Load Balancer finden Sie unter Bewährte Methoden für Load Balancing.
- Eine Liste der Annotationen, die für Ingress-Ressourcen mit AWS Application Load Balancer konfiguriert werden können, finden Sie unter Annotationen fürAWS Load Balancer Controller Ingress.

Voraussetzungen

- Cilium wurde gemäß den Anweisungen in the section called “CNI konfigurieren” installiert.
- Cilium-BGP-Steuerebene gemäß den Anweisungen in the section called “BGP konfigurieren” aktiviert. Wenn Sie BGP nicht verwenden möchten, müssen Sie eine alternative Methode verwenden, um die CIDRs Ihres On-Premises-Pods in Ihrem On-Premises-Netzwerk routingfähig zu machen. Wenn Sie Ihre On-Premises-Pod-CIDRs nicht routingfähig machen, kann ALB Ihre Pod-IP-Ziele nicht registrieren oder kontaktieren.
- Weitere Informationen finden Sie in den Anweisungen zur Einrichtung von Helm.
- eksctl ist in Ihrer Befehlszeilenumgebung installiert. Weitere Informationen finden Sie in den Installationsanweisungen für eksctl.

Verfahren

1. Laden Sie eine IAM-Richtlinie für den AWS-Lastenverteilungs-Controller herunter, die es ihm ermöglicht, in Ihrem Namen Aufrufe an AWS-APIs zu tätigen.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-
controller/refs/heads/main/docs/install/iam_policy.json
```

2. Erstellen Sie eine IAM-Richtlinie mit der im vorherigen Schritt heruntergeladenen Richtlinie.

```
aws iam create-policy \
    --policy-name AWSLoadBalancerControllerIAMPolicy \
    --policy-document file://iam_policy.json
```

3. Ersetzen Sie die Werte für den Cluster-Namen (`CLUSTER_NAME`), die AWS-Region (`AWS_REGION`) und die AWS-Konto-ID (`AWS_ACCOUNT_ID`) durch Ihre Einstellungen und führen Sie den folgenden Befehl aus.

```
eksctl create iamserviceaccount \
    --cluster=CLUSTER_NAME \
    --namespace=kube-system \
    --name=aws-load-balancer-controller \
    --attach-policy-arn=arn:aws:iam::AWS_ACCOUNT_ID:policy/
AWSLoadBalancerControllerIAMPolicy \
    --override-existing-serviceaccounts \
    --region AWS_REGION \
    --approve
```

4. Fügen Sie das Helm-Diagramm-Repository eks-charts hinzu und aktualisieren Sie Ihr lokales Helm-Repository, um sicherzustellen, dass Sie über die aktuellsten Diagramme verfügen.

```
helm repo add eks https://aws.github.io/eks-charts
```

```
helm repo update eks
```

5. Installieren Sie den AWS-Lastenverteilungs-Controller. Ersetzen Sie die Werte für Cluster-Name (`CLUSTER_NAME`), AWS-Region (`AWS_REGION`), VPC-ID (`VPC_ID`) und Helm-Chart-Version für AWS Load Balancer Controller (`AWS_LBC_HELM_VERSION`) durch Ihre Einstellungen und führen Sie den folgenden Befehl aus. Wenn Sie einen Cluster im kombinierten Modus mit Hybridknoten und Knoten in der AWS Cloud betreiben, können Sie AWS Load Balancer Controller auf Cloud-Knoten ausführen, indem Sie die Anweisungen unter the section called “AWS-Lastenverteilungs-Controller” befolgen.
   - Die neueste Version des Helm-Charts finden Sie, indem Sie `helm search repo eks/aws-load-balancer-controller --versions` ausführen.

```
helm install aws-load-balancer-controller eks/aws-load-balancer-controller \
  -n kube-system \
  --version AWS_LBC_HELM_VERSION \
  --set clusterName=CLUSTER_NAME \
  --set region=AWS_REGION \
  --set vpcId=VPC_ID \
  --set serviceAccount.create=false \
  --set serviceAccount.name=aws-load-balancer-controller
```

6. Überprüfen Sie, ob der AWS Load Balancer Controller erfolgreich installiert wurde.

```
kubectl get -n kube-system deployment aws-load-balancer-controller
```

```
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
aws-load-balancer-controller   2/2     2            2           84s
```

7. Erstellen Sie eine Beispielanwendung. Das folgende Beispiel verwendet die Beispielanwendung Istio Bookinfo für Microservices.

```
kubectl apply -f https://raw.githubusercontent.com/istio/istio/refs/heads/master/
samples/bookinfo/platform/kube/bookinfo.yaml
```

8. Erstellen Sie eine Datei mit dem Namen `my-ingress-alb.yaml` und dem folgenden Inhalt.

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-ingress
  namespace: default
  annotations:
    alb.ingress.kubernetes.io/load-balancer-name: "my-ingress-alb"
    alb.ingress.kubernetes.io/target-type: "ip"
    alb.ingress.kubernetes.io/scheme: "internet-facing"
    alb.ingress.kubernetes.io/healthcheck-path: "/details/1"
spec:
  ingressClassName: alb
  rules:
  - http:
      paths:
      - backend:
          service:
            name: details
            port:
              number: 9080
        path: /details
        pathType: Prefix
```

9. Wenden Sie die Ingress-Konfiguration auf Ihren Cluster an.

```
kubectl apply -f my-ingress-alb.yaml
```

10 Die Bereitstellung von ALB für Ihre Ingress-Ressource kann einige Minuten dauern. Sobald der ALB bereitgestellt ist, wird Ihrer Ingress-Ressource eine Adresse zugewiesen, die dem DNS-Namen der ALB-Bereitstellung entspricht. Die Adresse hat das Format `<alb-name>-<random-string>.<region>.elb.amazonaws.com`.

```
kubectl get ingress my-ingress
```

```
NAME         CLASS   HOSTS   ADDRESS
    PORTS   AGE
my-ingress   alb     *       my-ingress-alb-<random-
string>.<region>.elb.amazonaws.com   80      23m
```

11 Zugriff auf den Service über die Adresse des ALB.

```
curl -s http//my-ingress-alb-<random-string>.<region>.elb.amazonaws.com:80/details/1
 | jq
```

```
{
  "id": 1,
  "author": "William Shakespeare",
  "year": 1595,
  "type": "paperback",
  "pages": 200,
  "publisher": "PublisherA",
  "language": "English",
  "ISBN-10": "1234567890",
  "ISBN-13": "123-1234567890"
  "details": "This is the details page"
}
```

Übersicht über Cilium Ingress und Cilium Gateway

Die Ingress-Funktionen von Cilium sind in die Architektur von Cilium integriert und können über die Kubernetes Ingress API oder Gateway API verwaltet werden. Wenn Sie noch nicht über vorhandene Ingress-Ressourcen verfügen, empfiehlt AWS den Einstieg mit der Gateway-API, da diese eine ausdrucksstärkere und flexiblere Möglichkeit zur Definition und Verwaltung von Kubernetes-Netzwerkressourcen bietet. Die Kubernetes-Gateway-API zielt darauf ab, die Definition und Verwaltung von Netzwerkressourcen für Ingress, Lastenausgleich und Service Mesh in Kubernetes-Clustern zu standardisieren.

Wenn Sie die Ingress- oder Gateway-Features von Cilium aktivieren, gleicht der Cilium-Operator Ingress-/Gateway-Objekte im Cluster ab und Envoy-Proxys auf jedem Knoten verarbeiten den Layer-7-Netzwerk-Datenverkehr (L7). Cilium stellt Ingress-/Gateway-Infrastruktur wie Load Balancer nicht direkt bereit. Wenn Sie Cilium Ingress/Gateway mit einem Load Balancer verwenden möchten, müssen Sie die Tools des Load Balancers, üblicherweise einen Ingress- oder Gateway-Controller, verwenden, um die Infrastruktur des Load Balancers bereitzustellen und zu verwalten.

Für den Ingress-/Gateway-Datenverkehr übernimmt Cilium den Kernnetzwerk-Datenverkehr und die Durchsetzung der L3/L4-Richtlinien, während integrierte Envoy-Proxys den L7-Netzwerk-Datenverkehr verarbeiten. Mit Cilium Ingress/Gateway ist Envoy für die Anwendung von L7-Routing-Regeln, Richtlinien und Anforderungsmanipulation, erweitertes Datenverkehrs-Management wie Datenverkehrsaufteilung und -spiegelung sowie TLS-Beendigung und -Initiierung verantwortlich. Die Envoy-Proxys von Cilium werden standardmäßig als separates DaemonSet (`cilium-envoy`) bereitgestellt, wodurch Envoy und der Cilium-Agent separat aktualisiert, skaliert und verwaltet werden können.

Weitere Informationen zur Funktionsweise von Cilium Ingress und Cilium Gateway finden Sie auf den Seiten Cilium Ingress und Cilium Gateway in der Cilium-Dokumentation.

Vergleich zwischen Cilium Ingress und Gateway

Die folgende Tabelle fasst die Features von Cilium Ingress und Cilium Gateway in Cilium-Version 1.17.x zusammen.

Funktion	Ingress	Gateway
Servicetyp LoadBalancer	Ja	Ja
Servicetyp NodePort	Ja	Keine[1]
Host-Netzwerk	Ja	Ja
Gemeinsam genutzter Load Balancer	Ja	Ja
Dedizierter Load Balancer	Ja	Nr.[2]
Netzwerkrichtlinien	Ja	Ja
Protokolle	Layer 7 (HTTP(S), gRPC)	Layer 7 (HTTP(S), gRPC)[3]

Funktion	Ingress	Gateway
TLS Passthrough	Ja	Ja
Verwaltung des Datenverkehrs	Pfad- und Host-Routing	Pfad- und Host-Routing, URL-Weiterleitung und -Umschreibung, Datenverkehrsteilung, Header-Änderung

[1] Cilium-Gateway-Support für NodePort-Services ist für Cilium-Version 1.18.x geplant (#27273)

[2] Cilium-Gateway-Support für dedizierte Load Balancer (#25567)

[3] Cilium-Gateway-Support für TCP/UDP (#21929)

Cilium Gateway installieren

Überlegungen

- Cilium muss mit `nodePort.enabled` auf `true` konfiguriert werden, wie in den folgenden Beispielen dargestellt. Wenn Sie das kube-proxy-Ersatz-Feature von Cilium verwenden, müssen Sie `nodePort.enabled` nicht auf `true` setzen.
- Cilium muss mit `envoy.enabled` auf `true` konfiguriert werden, wie in den folgenden Beispielen dargestellt.
- Cilium Gateway kann im Load-Balancer-Modus (Standard) oder im Host-Netzwerkmodus bereitgestellt werden.
- Wenn Sie Cilium Gateway im Load Balancer-Modus verwenden, muss die `service.beta.kubernetes.io/aws-load-balancer-type: "external"`-Annotation auf der Gateway-Ressource festgelegt werden, um zu verhindern, dass der alte AWS-Cloud-Anbieter einen Classic Load Balancer für den Service vom Typ LoadBalancer erstellt, den Cilium für die Gateway-Ressource erstellt.
- Bei Verwendung von Cilium Gateway im Host-Netzwerkmodus ist der Service vom Typ LoadBalancer-Modus deaktiviert. Der Host-Netzwerkmodus ist nützlich für Umgebungen ohne Load Balancer-Infrastruktur. Weitere Informationen finden Sie unter the section called “Host-Netzwerk”.

Voraussetzungen

1. Helm ist in Ihrer Befehlszeilenumgebung installiert. Weitere Informationen finden Sie unter Anweisungen zur Einrichtung von Helm.
2. Cilium wurde gemäß den Anweisungen in the section called “CNI konfigurieren” installiert.

Verfahren

1. Installieren Sie die benutzerdefinierten Ressourcendefinitionen (CRDs) der Kubernetes-Gateway-API.

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/gateway-api/
v1.2.1/config/crd/standard/gateway.networking.k8s.io_gatewayclasses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/gateway-api/
v1.2.1/config/crd/standard/gateway.networking.k8s.io_gateways.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/gateway-api/
v1.2.1/config/crd/standard/gateway.networking.k8s.io_httproutes.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/gateway-api/
v1.2.1/config/crd/standard/gateway.networking.k8s.io_referencegrants.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/gateway-api/
v1.2.1/config/crd/standard/gateway.networking.k8s.io_grpcroutes.yaml
```

2. Erstellen Sie eine Datei mit dem Namen `cilium-gateway-values.yaml` und den folgenden Inhalten. Das folgende Beispiel konfiguriert Cilium Gateway so, dass der Standard-Load-Balancer-Modus verwendet wird und ein separates `cilium-envoy` DaemonSet für Envoy-Proxys verwendet wird, das so konfiguriert ist, dass es nur auf Hybridknoten ausgeführt wird.

```
gatewayAPI:
  enabled: true
  # uncomment to use host network mode
  # hostNetwork:
  #   enabled: true
nodePort:
  enabled: true
envoy:
  enabled: true
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
```

```
          - key: eks.amazonaws.com/compute-type
            operator: In
            values:
            - hybrid
```

3. Wenden Sie die Helm-Wertedatei auf Ihren Cluster an.

```
helm upgrade cilium oci://public.ecr.aws/eks/cilium/cilium \
  --namespace kube-system \
  --reuse-values \
  --set operator.rollOutPods=true \
  --values cilium-gateway-values.yaml
```

4. Bestätigen Sie, dass der Cilium-Operator, der Agent und die Envoy-Pods ausgeführt werden.

```
kubectl -n kube-system get pods --selector=app.kubernetes.io/part-of=cilium
```

```
NAME                               READY   STATUS    RESTARTS   AGE
cilium-envoy-5pgnd                 1/1     Running   0          6m31s
cilium-envoy-6fhg4                 1/1     Running   0          6m30s
cilium-envoy-jskrk                 1/1     Running   0          6m30s
cilium-envoy-k2xtb                 1/1     Running   0          6m31s
cilium-envoy-w5s9j                 1/1     Running   0          6m31s
cilium-grwlc                       1/1     Running   0          4m12s
cilium-operator-68f7766967-5nnbl   1/1     Running   0          4m20s
cilium-operator-68f7766967-7spfz   1/1     Running   0          4m20s
cilium-pnxcv                       1/1     Running   0          6m29s
cilium-r7qkj                       1/1     Running   0          4m12s
cilium-wxhfn                       1/1     Running   0          4m1s
cilium-z7hlb                       1/1     Running   0          6m30s
```

Cilium Gateway konfigurieren

Cilium Gateway wird für Gateway-Objekte aktiviert, indem `gatewayClassName` auf `cilium` gesetzt wird. Der Service, den Cilium für Gateway-Ressourcen erstellt, kann mit Feldern im Gateway-Objekt konfiguriert werden. Gängige Annotationen, die von Gateway-Controllern zur Konfiguration der Load-Balancer-Infrastruktur verwendet werden, können über das `infrastructure`-Feld des Gateway-Objekts konfiguriert werden. Bei Verwendung von Ciliums LoadBalancer IPAM (siehe Beispiel in the section called “Servicetyp LoadBalancer”) kann die für den Service vom Typ LoadBalancer zu verwendende IP-Adresse über das `addresses`-Feld des Gateway-Objekts konfiguriert werden.

Weitere Informationen zur Gateway-Konfiguration finden Sie in der Kubernetes-Gateway-API-Spezifikation.

```
apiVersion: gateway.networking.k8s.io/v1
kind: Gateway
metadata:
  name: my-gateway
spec:
  gatewayClassName: cilium
  infrastructure:
    annotations:
      service.beta.kubernetes.io/...
      service.kuberentes.io/...
  addresses:
  - type: IPAddress
    value: <LoadBalancer IP address>
  listeners:
  ...
```

Cilium und die Kubernetes Gateway-Spezifikation unterstützen die Ressourcen GatewayClass, Gateway, HTTPRoute, GRPCRoute und ReferenceGrant.

- Eine Liste der verfügbaren Felder finden Sie in den HTTPRoute- und GRPCRoute-Spezifikationen.
- Informationen zur Verwendung und Konfiguration dieser Ressourcen finden Sie in den Beispielen im nachfolgenden the section called “Cilium Gateway bereitstellen”-Abschnitt und in der Cilium-Dokumentation.

Cilium Gateway bereitstellen

1. Erstellen Sie eine Beispielanwendung. Das folgende Beispiel verwendet die Beispielanwendung Istio Bookinfo für Microservices.

   ```
   kubectl apply -f https://raw.githubusercontent.com/istio/istio/refs/heads/master/
   samples/bookinfo/platform/kube/bookinfo.yaml
   ```

2. Bestätigen Sie, dass die Anwendung erfolgreich ausgeführt wird.

   ```
   kubectl get pods
   ```

   ```
   NAME                              READY   STATUS    RESTARTS   AGE
   ```

```
details-v1-766844796b-9965p       1/1     Running   0          81s
productpage-v1-54bb874995-jmc8j   1/1     Running   0          80s
ratings-v1-5dc79b6bcd-smzxz       1/1     Running   0          80s
reviews-v1-598b896c9d-vj7gb       1/1     Running   0          80s
reviews-v2-556d6457d-xbt8v        1/1     Running   0          80s
reviews-v3-564544b4d6-cpmvq       1/1     Running   0          80s
```

3. Erstellen Sie eine Datei mit dem Namen `my-gateway.yaml` und dem folgenden Inhalt. Das folgende Beispiel verwendet die `service.beta.kubernetes.io/aws-load-balancer-type: "external"`-Annotation, um zu verhindern, dass der alte AWS-Cloud-Anbieter einen Classic Load Balancer für den Service vom Typ LoadBalancer erstellt, den Cilium für die Gateway-Ressource erstellt.

```
---
apiVersion: gateway.networking.k8s.io/v1
kind: Gateway
metadata:
  name: my-gateway
spec:
  gatewayClassName: cilium
  infrastructure:
    annotations:
      service.beta.kubernetes.io/aws-load-balancer-type: "external"
  listeners:
  - protocol: HTTP
    port: 80
    name: web-gw
    allowedRoutes:
      namespaces:
        from: Same
---
apiVersion: gateway.networking.k8s.io/v1
kind: HTTPRoute
metadata:
  name: http-app-1
spec:
  parentRefs:
  - name: my-gateway
    namespace: default
  rules:
  - matches:
    - path:
        type: PathPrefix
```

```
        value: /details
    backendRefs:
    - name: details
      port: 9080
```

4. Wenden Sie die Gateway-Ressource auf Ihren Cluster an.

```
kubectl apply -f my-gateway.yaml
```

5. Bestätigen Sie, dass die Gateway-Ressource und der entsprechende Service erstellt wurden. Zum gegenwärtigen Zeitpunkt wird davon ausgegangen, dass das ADDRESS-Feld der Gateway-Ressource keine IP-Adresse oder keinen Host-Namen enthält und dass dem Service vom Typ LoadBalancer für die Gateway-Ressource ebenfalls keine IP-Adresse oder kein Host-Name zugewiesen ist.

```
kubectl get gateway my-gateway
```

```
NAME         CLASS    ADDRESS   PROGRAMMED   AGE
my-gateway   cilium             True         10s
```

```
kubectl get svc cilium-gateway-my-gateway
```

```
NAME                         TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)
    AGE
cilium-gateway-my-gateway   LoadBalancer   172.16.227.247   <pending>     80:30912/
TCP   24s
```

6. Fahren Sie mit the section called “Servicetyp LoadBalancer” fort, um die Gateway-Ressource für die Verwendung einer von Cilium Load Balancer IPAM zugewiesenen IP-Adresse zu konfigurieren, und mit the section called “Servicetyp NodePort” oder the section called “Host-Netzwerk”, um die Gateway-Ressource für die Verwendung von NodePort- oder Host-Netzwerkadressen zu konfigurieren.

Cilium Ingress installieren

Überlegungen

- Cilium muss mit `nodePort.enabled` auf `true` konfiguriert werden, wie in den folgenden Beispielen dargestellt. Wenn Sie das kube-proxy-Ersatz-Feature von Cilium verwenden, müssen Sie `nodePort.enabled` nicht auf `true` setzen.
- Cilium muss mit `envoy.enabled` auf `true` konfiguriert werden, wie in den folgenden Beispielen dargestellt.
- Wenn `ingressController.loadbalancerMode` auf `dedicated` gesetzt ist, erstellt Cilium dedizierte Services für jede Ingress-Ressource. Wenn `ingressController.loadbalancerMode` auf `shared` gesetzt ist, erstellt Cilium einen gemeinsamen Service vom Typ LoadBalancer für alle Ingress-Ressourcen im Cluster. Bei Verwendung des `shared`-Load-Balancer-Modus werden die Einstellungen für den gemeinsam genutzten Service wie `labels`, `annotations`, `type` und `loadBalancerIP` im `ingressController.service`-Abschnitt der Helm-Werte konfiguriert. Weitere Informationen finden Sie in der Wertereferenz von Cilium Helm.
- Wenn `ingressController.default` auf `true` gesetzt ist, wird Cilium als Standard-Ingress-Controller für den Cluster konfiguriert und erstellt Ingress-Einträge, auch wenn `ingressClassName` bei Ingress-Ressourcen nicht angegeben ist.
- Cilium Ingress kann im Load Balancer- (Standard), Knoten-Port- oder Host-Netzwerkmodus bereitgestellt werden. Wenn Cilium im Host-Netzwerkmodus installiert ist, werden die Modi „Service vom Typ LoadBalancer“ und „Service vom Typ NodePort“ deaktiviert. Weitere Informationen finden Sie unter the section called “Host-Netzwerk”.
- Setzen Sie in den Helm-Werten `ingressController.service.annotations` immer auf `service.beta.kubernetes.io/aws-load-balancer-type: "external"`, um zu verhindern, dass der alte AWS-Cloud-Anbieter einen Classic Load Balancer für den vom Helm-Chart-Diagramm erstellten Standard-`cilium-ingress`-Service erstellt.

Voraussetzungen

1. Helm ist in Ihrer Befehlszeilenumgebung installiert. Weitere Informationen finden Sie unter Anweisungen zur Einrichtung von Helm.
2. Cilium wurde gemäß den Anweisungen in the section called “CNI konfigurieren” installiert.

Verfahren

1. Erstellen Sie eine Datei mit dem Namen `cilium-ingress-values.yaml` und den folgenden Inhalten. Das folgende Beispiel konfiguriert Cilium Ingress so, dass der Standard-`dedicated`-Modus für Load-Balancer verwendet wird und ein separates `cilium-envoy` DaemonSet für Envoy-Proxys verwendet wird, das so konfiguriert ist, dass es nur auf Hybridknoten ausgeführt wird.

```
ingressController:
  enabled: true
  loadbalancerMode: dedicated
  service:
    annotations:
      service.beta.kubernetes.io/aws-load-balancer-type: "external"
nodePort:
  enabled: true
envoy:
  enabled: true
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: eks.amazonaws.com/compute-type
            operator: In
            values:
            - hybrid
```

2. Wenden Sie die Helm-Wertedatei auf Ihren Cluster an.

```
helm upgrade cilium oci://public.ecr.aws/eks/cilium/cilium \
  --namespace kube-system \
  --reuse-values \
  --set operator.rollOutPods=true \
  --values cilium-ingress-values.yaml
```

3. Bestätigen Sie, dass der Cilium-Operator, der Agent und die Envoy-Pods ausgeführt werden.

```
kubectl -n kube-system get pods --selector=app.kubernetes.io/part-of=cilium
```

```
NAME                               READY   STATUS    RESTARTS   AGE
```

```
cilium-envoy-5pgnd                  1/1     Running   0          6m31s
cilium-envoy-6fhg4                  1/1     Running   0          6m30s
cilium-envoy-jskrk                  1/1     Running   0          6m30s
cilium-envoy-k2xtb                  1/1     Running   0          6m31s
cilium-envoy-w5s9j                  1/1     Running   0          6m31s
cilium-grwlc                        1/1     Running   0          4m12s
cilium-operator-68f7766967-5nnbl    1/1     Running   0          4m20s
cilium-operator-68f7766967-7spfz    1/1     Running   0          4m20s
cilium-pnxcv                        1/1     Running   0          6m29s
cilium-r7qkj                        1/1     Running   0          4m12s
cilium-wxhfn                        1/1     Running   0          4m1s
cilium-z7hlb                        1/1     Running   0          6m30s
```

Cilium Ingress konfigurieren

Cilium Ingress wird für Ingress-Objekte aktiviert, indem `ingressClassName` auf `cilium` gesetzt wird. Der/die Service(s), den/die Cilium für Ingress-Ressourcen erstellt, kann/können mit Annotationen zu den Ingress-Objekten konfiguriert werden, wenn der `dedicated`-Load-Balancer-Modus verwendet wird, sowie in der Cilium-/Helm-Konfiguration, wenn der `shared`-Load-Balancer-Modus verwendet wird. Diese Annotationen werden oft von Ingress-Controllern verwendet, um die Load-Balancer-Infrastruktur oder andere Attribute des Service zu konfigurieren, wie z. B. den Servicetyp, den Load-Balancer-Modus, Ports und TLS Passthrough. Die wichtigsten Annotationen werden unten beschrieben. Eine vollständige Liste der unterstützten Annotationen finden Sie in den Cilium-Ingress-Annotationen in der Cilium-Dokumentation.

Anmerkung	Beschreibung
`ingress.cilium.io/loadbalancer-mode`	`dedicated` : Dedizierter Service vom Typ LoadBalancer für jede Ingress-Ressource (Standard). `shared`: Einzelner Service vom Typ LoadBalancer für alle Ingress-Ressourcen.
`ingress.cilium.io/service-type`	`LoadBalancer` : Der Service ist vom Typ LoadBalancer (Standard). `NodePort`: Der Service ist vom Typ NodePort.

Anmerkung	Beschreibung
`service.beta.kubernetes.io/aws-load-balancer-type`	`"external"` : Verhindern Sie, dass ältere AWS-Cloud-Anbieter Classic Load Balancer für Services vom Typ LoadBalancer bereitstellen.
`lbipam.cilium.io/ips`	Liste der IP-Adressen, die vom Cilium LoadBalancer IPAM zugewiesen werden sollen

Cilium und die Kubernetes-Ingress-Spezifikation unterstützen exakte, präfix- und implementierungsspezifische Übereinstimmungsregeln für Ingress-Pfade. Cilium unterstützt Regex als implementierungsspezifische Übereinstimmungsregel. Weitere Informationen finden Sie unter Ingress-Pfadtypen und -Priorität und Beispiele für Pfadtypen in der Cilium-Dokumentation sowie in den Beispielen im the section called “Cilium Ingress bereitstellen”-Abschnitt dieser Seite.

Ein Beispiel für ein Cilium-Ingress-Objekt ist unten dargestellt.

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-ingress
  annotations:
    service.beta.kuberentes.io/...
    service.kuberentes.io/...
spec:
  ingressClassName: cilium
  rules:
  ...
```

Cilium Ingress bereitstellen

1. Erstellen Sie eine Beispielanwendung. Das folgende Beispiel verwendet die Beispielanwendung Istio Bookinfo für Microservices.

   ```
   kubectl apply -f https://raw.githubusercontent.com/istio/istio/refs/heads/master/samples/bookinfo/platform/kube/bookinfo.yaml
   ```

2. Bestätigen Sie, dass die Anwendung erfolgreich ausgeführt wird.

```
kubectl get pods
```

```
NAME                              READY   STATUS    RESTARTS   AGE
details-v1-766844796b-9965p       1/1     Running   0          81s
productpage-v1-54bb874995-jmc8j   1/1     Running   0          80s
ratings-v1-5dc79b6bcd-smzxz       1/1     Running   0          80s
reviews-v1-598b896c9d-vj7gb       1/1     Running   0          80s
reviews-v2-556d6457d-xbt8v        1/1     Running   0          80s
reviews-v3-564544b4d6-cpmvq       1/1     Running   0          80s
```

3. Erstellen Sie eine Datei mit dem Namen `my-ingress.yaml` und dem folgenden Inhalt. Das folgende Beispiel verwendet die `service.beta.kubernetes.io/aws-load-balancer-type: "external"`-Annotation, um zu verhindern, dass der alte AWS-Cloud-Anbieter einen Classic Load Balancer für den Service vom Typ LoadBalancer erstellt, den Cilium für die Ingress-Ressource erstellt.

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-ingress
  namespace: default
  annotations:
    service.beta.kubernetes.io/aws-load-balancer-type: "external"
spec:
  ingressClassName: cilium
  rules:
  - http:
      paths:
      - backend:
          service:
            name: details
            port:
              number: 9080
        path: /details
        pathType: Prefix
```

4. Wenden Sie die Ingress-Ressource auf Ihren Cluster an.

```
kubectl apply -f my-ingress.yaml
```

5. Bestätigen Sie, dass die Ingress-Ressource und der entsprechende Service erstellt wurden. Zu diesem Zeitpunkt wird davon ausgegangen, dass das ADDRESS-Feld der Ingress-Ressource keine IP-Adresse oder keinen Host-Namen enthält und dass dem gemeinsam genutzten oder dedizierten Service vom Typ LoadBalancer für die Ingress-Ressource ebenfalls keine IP-Adresse oder kein Host-Name zugewiesen ist.

```
kubectl get ingress my-ingress
```

```
NAME         CLASS    HOSTS   ADDRESS   PORTS   AGE
my-ingress   cilium   *                 80      8s
```

Für Load-Balancer-Modus `shared`

```
kubectl -n kube-system get svc cilium-ingress
```

```
NAME             TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)
      AGE
cilium-ingress   LoadBalancer   172.16.217.48   <pending>     80:32359/TCP,443:31090/
TCP   10m
```

Für Load-Balancer-Modus `dedicated`

```
kubectl -n default get svc cilium-ingress-my-ingress
```

```
NAME                         TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)
                  AGE
cilium-ingress-my-ingress   LoadBalancer   172.16.193.15   <pending>     80:32088/
TCP,443:30332/TCP   25s
```

6. Fahren Sie mit the section called “Servicetyp LoadBalancer” fort, um die Ingress-Ressource so zu konfigurieren, dass sie eine von Cilium Load Balancer IPAM zugewiesene IP-Adresse verwendet, und mit the section called “Servicetyp NodePort” oder the section called “Host-Netzwerk”, um die Ingress-Ressource so zu konfigurieren, dass sie NodePort- oder Host-Netzwerkadressen verwendet.

Servicetyp LoadBalancer

Vorhandene Load-Balancer-Infrastruktur

Standardmäßig erstellt Cilium sowohl für Cilium Ingress als auch für Cilium-Gateway-Kubernetes-Services vom Typ LoadBalancer für die Ingress-/Gateway-Ressourcen. Die Attribute der von Cilium erstellten Service(s) können über die Ingress- und Gateway-Ressourcen konfiguriert werden. Wenn Sie Ingress- oder Gateway-Ressourcen erstellen, werden die extern verfügbar gemachten IP-Adressen oder Host-Namen für Ingress oder Gateway von der Load Balancer-Infrastruktur zugewiesen, die normalerweise von einem Ingress- oder Gateway-Controller bereitgestellt wird.

Viele Ingress- und Gateway-Controller verwenden Annotationen, um die Load-Balancer-Infrastruktur zu erkennen und zu konfigurieren. Die Annotationen für diese Ingress- und Gateway-Controller werden auf den Ingress- oder Gateway-Ressourcen konfiguriert, wie in den vorherigen Beispielen oben gezeigt. Informationen zu den unterstützten Annotationen finden Sie in der Dokumentation Ihres Ingress- oder Gateway-Controllers. Eine Liste gängiger Controller finden Sie in der Kubernetes-Ingress-Dokumentation und der Kubernetes-Gateway-Dokumentation.

> **Important**
>
> Cilium Ingress und Gateway können nicht mit dem AWS Load Balancer Controller und AWS Network Load Balancern (NLBs) mit EKS-Hybridknoten verwendet werden. Der Versuch, diese zusammen zu verwenden, führt zu nicht registrierten Zielen, da der NLB versucht, eine direkte Verbindung zu den Pod-IPs herzustellen, die den Service vom Typ LoadBalancer unterstützen, wenn der `target-type` des NLB auf `ip` eingestellt ist (Voraussetzung für die Verwendung von NLB mit Workloads, die auf EKS-Hybridknoten ausgeführt werden).

Keine Load-Balancer-Infrastruktur

Wenn Sie in Ihrer Umgebung keine Load Balancer-Infrastruktur und keinen entsprechenden Ingress-/Gateway-Controller haben, können Ingress-/Gateway-Ressourcen und entsprechende Services vom Typ LoadBalancer so konfiguriert werden, dass sie IP-Adressen verwenden, die von der Load-Balancer-IP-Adressverwaltung (LB IPAM) von Cilium zugewiesen werden. Cilium LB IPAM kann mit bekannten IP-Adressbereichen aus Ihrer lokalen Umgebung konfiguriert werden. Mit dem integrierten Border Gateway Protocol (BGP)-Support von Cilium oder L2-Ankündigungen können die LoadBalancer-IP-Adressen in Ihrem On-Premises-Netzwerk bekannt gegeben werden.

Das folgende Beispiel veranschaulicht, wie Sie Ciliums LB IPAM mit einer IP-Adresse für Ihre Ingress-/Gateway-Ressourcen konfigurieren und wie Sie die Cilium-BGP-Steuerebene konfigurieren, um die LoadBalancer-IP-Adresse im On-Premises-Netzwerk bekannt zu geben. Das LB-IPAM-Feature von Cilium ist standardmäßig aktiviert, wird jedoch erst nach der Erstellung einer `CiliumLoadBalancerIPPool`-Ressource aktiviert.

Voraussetzungen

- Cilium Ingress oder Gateway wurde gemäß den Anweisungen in the section called “Cilium Ingress installieren” oder the section called “Cilium Gateway installieren” installiert.
- Cilium-Ingress- oder Gateway-Ressourcen mit Beispielanwendung bereitgestellt gemäß den Anweisungen in the section called “Cilium Ingress bereitstellen” oder the section called “Cilium Gateway bereitstellen”.
- Cilium-BGP-Steuerebene gemäß den Anweisungen in the section called “BGP konfigurieren” aktiviert. Wenn Sie BGP nicht verwenden möchten, können Sie diese Voraussetzung überspringen. Sie können jedoch erst dann auf Ihre Ingress- oder Gateway-Ressource zugreifen, wenn die von Cilium LB IPAM zugewiesene LoadBalancer-IP-Adresse in Ihrem On-Premises-Netzwerk routingfähig sind.

Verfahren

1. Patchen Sie optional die Ingress- oder Gateway-Ressource, um eine bestimmte IP-Adresse für den Service vom Typ LoadBalancer anzufordern. Wenn Sie keine bestimmte IP-Adresse anfordern, weist Cilium im nächsten Schritt eine IP-Adresse aus dem in der `CiliumLoadBalancerIPPool`-Ressource konfigurierten IP-Adressbereich zu. Ersetzen Sie in den folgenden Befehlen `LB_IP_ADDRESS` durch die anzufordernde IP-Adresse für den Service vom Typ LoadBalancer.

   Gateway

   ```
   kubectl patch gateway -n default my-gateway --type=merge -p '{
     "spec": {
       "addresses": [{"type": "IPAddress", "value": "LB_IP_ADDRESS"}]
     }
   }'
   ```

   Ingress

```
kubectl patch ingress my-ingress --type=merge -p '{
  "metadata": {"annotations": {"lbipam.cilium.io/ips": "LB_IP_ADDRESS"}}
}'
```

2. Erstellen Sie eine Datei mit dem Namen `cilium-lbip-pool-ingress.yaml` mit einer `CiliumLoadBalancerIPPool`-Ressource, um den IP-Adressbereich des Load Balancers für Ihre Ingress-/Gateway-Ressourcen zu konfigurieren.
   - Wenn Sie Cilium Ingress verwenden, wendet Cilium das `cilium.io/ingress: "true"`-Label automatisch auf die Services an, die es für Ingress-Ressourcen erstellt. Sie können dieses Label im Feld `serviceSelector` der `CiliumLoadBalancerIPPool`-Ressourcendefinition verwenden, um die für LB IPAM geeigneten Services auszuwählen.
   - Wenn Sie Cilium-Gateway verwenden, können Sie das `gateway.networking.k8s.io/gateway-name`-Label in den Feldern `serviceSelector` der `CiliumLoadBalancerIPPool`-Ressourcendefinition verwenden, um die für LB IPAM geeigneten Gateway-Ressourcen auszuwählen.
   - Ersetzen Sie `LB_IP_CIDR` durch den IP-Adressbereich, der für die IP-Adressen des Load Balancers verwendet werden soll. Um eine einzelne IP-Adresse auszuwählen, verwenden Sie ein `/32` CIDR. Weitere Informationen finden Sie unter LoadBalancer-IP-Adressverwaltung in der Cilium-Dokumentation.

```
apiVersion: cilium.io/v2alpha1
kind: CiliumLoadBalancerIPPool
metadata:
  name: bookinfo-pool
spec:
  blocks:
  - cidr: "LB_IP_CIDR"
  serviceSelector:
    # if using Cilium Gateway
    matchExpressions:
      - { key: gateway.networking.k8s.io/gateway-name, operator: In, values: [ my-
gateway ] }
    # if using Cilium Ingress
    matchLabels:
      cilium.io/ingress: "true"
```

3. Wenden Sie die `CiliumLoadBalancerIPPool`-Ressource auf Ihren Cluster an.

```
kubectl apply -f cilium-lbip-pool-ingress.yaml
```

4. Bestätigen Sie, dass von Cilium LB IPAM eine IP-Adresse für die Ingress-/Gateway-Ressource zugewiesen wurde.

   Gateway

   ```
   kubectl get gateway my-gateway
   ```

   ```
   NAME         CLASS    ADDRESS        PROGRAMMED   AGE
   my-gateway   cilium   LB_IP_ADDRESS    True         6m41s
   ```

   Ingress

   ```
   kubectl get ingress my-ingress
   ```

   ```
   NAME         CLASS    HOSTS   ADDRESS        PORTS   AGE
   my-ingress   cilium   *       LB_IP_ADDRESS   80      10m
   ```

5. Erstellen Sie eine Datei mit dem Namen `cilium-bgp-advertisement-ingress.yaml` und einer `CiliumBGPAdvertisement`-Ressource, um die LoadBalancer-IP-Adresse für die Ingress-/Gateway-Ressourcen bekannt zu geben. Wenn Sie Cilium BGP nicht verwenden, können Sie diesen Schritt überspringen. Die für Ihre Ingress-/Gateway-Ressource verwendete LoadBalancer-IP-Adresse muss in Ihrem On-Premises-Netzwerk routingfähig sein, damit Sie den Service im nächsten Schritt abfragen können.

   ```
   apiVersion: cilium.io/v2alpha1
   kind: CiliumBGPAdvertisement
   metadata:
     name: bgp-advertisement-lb-ip
     labels:
       advertise: bgp
   spec:
     advertisements:
       - advertisementType: "Service"
         service:
           addresses:
             - LoadBalancerIP
         selector:
   ```

```
        # if using Cilium Gateway
        matchExpressions:
          - { key: gateway.networking.k8s.io/gateway-name, operator: In, values:
 [ my-gateway ] }
        # if using Cilium Ingress
        matchLabels:
          cilium.io/ingress: "true"
```

6. Wenden Sie die `CiliumBGPAdvertisement`-Ressource auf Ihren Cluster an.

```
kubectl apply -f cilium-bgp-advertisement-ingress.yaml
```

7. Greifen Sie über die von Cilium LB IPAM zugewiesene IP-Adresse auf den Service zu.

```
curl -s http://LB_IP_ADDRESS:80/details/1 | jq
```

```
{
  "id": 1,
  "author": "William Shakespeare",
  "year": 1595,
  "type": "paperback",
  "pages": 200,
  "publisher": "PublisherA",
  "language": "English",
  "ISBN-10": "1234567890",
  "ISBN-13": "123-1234567890"
}
```

Servicetyp NodePort

Wenn Sie in Ihrer Umgebung keine Load Balancer-Infrastruktur und keinen entsprechenden Ingress-Controller haben oder wenn Sie Ihre Load Balancer-Infrastruktur selbst verwalten oder DNS-basiertes Load Balancing verwenden, können Sie Cilium Ingress so konfigurieren, dass Service vom Typ NodePort für die Ingress-Ressourcen erstellt werden Bei Verwendung von NodePort mit Cilium Ingress wird der Service vom Typ NodePort auf einem Port auf jedem Knoten im Port-Bereich 30000-32767 verfügbar gemacht. In diesem Modus wird der Datenverkehr, sobald er einen beliebigen Knoten im Cluster auf dem NodePort erreicht, an einen Pod weitergeleitet, der den Service unterstützt. Dieser Pod kann sich auf demselben Knoten oder auf einem anderen Knoten befinden.

> **Note**
>
> Der Support für Cilium Gateway für NodePort-Services ist für Cilium Version 1.18.x geplant (#27273).

Voraussetzungen

- Cilium Ingress wurde gemäß den Anweisungen in the section called “Cilium Ingress installieren” installiert.
- Cilium-Ingress-Ressourcen mit Beispielanwendung wurden gemäß den Anweisungen in the section called “Cilium Ingress bereitstellen” bereitgestellt.

Verfahren

1. Patchen Sie die vorhandene Ingress-Ressource `my-ingress`, um sie vom Servicetyp LoadBalancer in NodePort zu ändern.

   ```
   kubectl patch ingress my-ingress --type=merge -p '{
       "metadata": {"annotations": {"ingress.cilium.io/service-type": "NodePort"}}
   }'
   ```

   Falls Sie die Ingress-Ressource noch nicht erstellt haben, können Sie sie erstellen, indem Sie die folgende Ingress-Definition auf Ihren Cluster anwenden. Beachten Sie, dass die folgende Ingress-Definition die in the section called “Cilium Ingress bereitstellen” beschriebene Istio-Bookinfo-Beispielanwendung verwendet.

   ```
   apiVersion: networking.k8s.io/v1
   kind: Ingress
   metadata:
     name: my-ingress
     namespace: default
     annotations:
       service.beta.kubernetes.io/aws-load-balancer-type: "external"
       "ingress.cilium.io/service-type": "NodePort"
   spec:
     ingressClassName: cilium
     rules:
     - http:
   ```

```
      paths:
      - backend:
          service:
            name: details
            port:
              number: 9080
        path: /details
        pathType: Prefix
```

2. Bestätigen Sie, dass der Service für die Ingress-Ressource aktualisiert wurde, um den Servicetyp NodePort zu verwenden. Notieren Sie sich den Port für das HTTP-Protokoll in der Ausgabe. Im folgenden Beispiel lautet dieser HTTP-Port 32353 und wird in einem späteren Schritt zur Abfrage des Services verwendet. Der Vorteil der Verwendung von Cilium Ingress mit einem Servic. vom Typ NodePort besteht darin, dass Sie pfad- und Host-basiertes Routing sowie Netzwerkrichtlinien für den Ingress-Datenverkehr anwenden können. Dies ist bei einem Standard-Service vom Typ NodePort ohne Ingress nicht möglich.

```
kubectl -n default get svc cilium-ingress-my-ingress
```

```
NAME                         TYPE       CLUSTER-IP      EXTERNAL-IP   PORT(S)
             AGE
cilium-ingress-my-ingress    NodePort   172.16.47.153   <none>        80:32353/
TCP,443:30253/TCP   27m
```

3. Rufen Sie die IP-Adressen Ihrer Knoten in Ihrem Cluster ab.

```
kubectl get nodes -o wide
```

```
NAME                   STATUS   ROLES    AGE   VERSION              INTERNAL-IP
 EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION       CONTAINER-RUNTIME
mi-026d6a261e355fba7   Ready    <none>   23h   v1.32.3-eks-473151a   10.80.146.150
 <none>        Ubuntu 22.04.5 LTS   5.15.0-142-generic   containerd://1.7.27
mi-082f73826a163626e   Ready    <none>   23h   v1.32.3-eks-473151a   10.80.146.32
 <none>        Ubuntu 22.04.4 LTS   5.15.0-142-generic   containerd://1.7.27
mi-09183e8a3d755abf6   Ready    <none>   23h   v1.32.3-eks-473151a   10.80.146.33
 <none>        Ubuntu 22.04.4 LTS   5.15.0-142-generic   containerd://1.7.27
mi-0d78d815980ed202d   Ready    <none>   23h   v1.32.3-eks-473151a   10.80.146.97
 <none>        Ubuntu 22.04.4 LTS   5.15.0-142-generic   containerd://1.7.27
mi-0daa253999fe92daa   Ready    <none>   23h   v1.32.3-eks-473151a   10.80.146.100
 <none>        Ubuntu 22.04.4 LTS   5.15.0-142-generic   containerd://1.7.27
```

4. Greifen Sie mit den IP-Adressen Ihrer Knoten und dem oben erfassten NodePort auf den Service vom Typ NodePort zu. Im folgenden Beispiel ist die verwendete Knoten-IP-Adresse `10.80.146.32` und der Knoten-Port ist `32353`. Ersetzen Sie diese durch die Werte für Ihre Umgebung.

```
curl -s http://10.80.146.32:32353/details/1 | jq
```

```
{
  "id": 1,
  "author": "William Shakespeare",
  "year": 1595,
  "type": "paperback",
  "pages": 200,
  "publisher": "PublisherA",
  "language": "English",
  "ISBN-10": "1234567890",
  "ISBN-13": "123-1234567890"
}
```

Host-Netzwerk

Ähnlich wie beim Service vom Typ NodePort können Sie – wenn Sie nicht über eine Load-Balancer-Infrastruktur und einen Ingress- oder Gateway-Controller verfügen oder wenn Sie Ihren Load Balancing mit einem externen Load Balancer selbst verwalten – Cilium Ingress und Cilium Gateway so konfigurieren, dass Ingress- und Gateway-Ressourcen direkt im Host-Netzwerk verfügbar gemacht werden. Wenn der Host-Netzwerkmodus für eine Ingress- oder Gateway-Ressource aktiviert ist, werden die Modi „Service vom Typ LoadBalancer“ und „NodePort“ automatisch deaktiviert. Der Host-Netzwerkmodus und diese alternativen Modi schließen sich für jede Ingress- oder Gateway-Ressource gegenseitig aus. Im Vergleich zum Modus „Service vom Typ NodePort“ bietet der Host-Netzwerkmodus zusätzliche Flexibilität hinsichtlich des verwendbaren Portbereichs (er ist nicht auf den NodePort-Bereich 30000–32767 beschränkt). Außerdem können Sie eine Teilmenge von Knoten konfigurieren, auf denen die Envoy-Proxys im Host-Netzwerk ausgeführt werden.

Voraussetzungen

- Cilium Ingress oder Gateway wurde gemäß den Anweisungen in the section called “Cilium Ingress installieren” oder the section called “Cilium Gateway installieren” installiert.

Verfahren

Gateway

1. Erstellen Sie eine Datei mit dem Namen `cilium-gateway-host-network.yaml` und dem folgenden Inhalt.

```
gatewayAPI:
  enabled: true
  hostNetwork:
    enabled: true
    # uncomment to restrict nodes where Envoy proxies run on the host network
    # nodes:
    #   matchLabels:
    #     role: gateway
```

2. Wenden Sie die Cilium Gateway-Konfiguration des Host-Netzwerks auf Ihren Cluster an.

```
helm upgrade cilium oci://public.ecr.aws/eks/cilium/cilium \
  --namespace kube-system \
  --reuse-values \
  --set operator.rollOutPods=true \
  -f cilium-gateway-host-network.yaml
```

Wenn Sie die Gateway-Ressource noch nicht erstellt haben, können Sie sie erstellen, indem Sie die folgende Gateway-Definition auf Ihren Cluster anwenden. Die nachfolgende Gateway-Definition verwendet die in the section called “Cilium Gateway bereitstellen” beschriebene Istio-Bookinfo-Beispielanwendung. Im folgenden Beispiel ist die Gateway-Ressource so konfiguriert, dass sie den 8111-Port für den HTTP-Listener verwendet, den gemeinsamen Listener-Port für die im Host-Netzwerk ausgeführten Envoy-Proxys. Wenn Sie einen bevorzugten Port (unter 1023) für die Gateway-Ressource verwenden, finden Sie Anweisungen dazu in der Cilium-Dokumentation.

```
---
apiVersion: gateway.networking.k8s.io/v1
kind: Gateway
metadata:
  name: my-gateway
spec:
  gatewayClassName: cilium
  listeners:
  - protocol: HTTP
```

```
    port: 8111
    name: web-gw
    allowedRoutes:
      namespaces:
        from: Same
---
apiVersion: gateway.networking.k8s.io/v1
kind: HTTPRoute
metadata:
  name: http-app-1
spec:
  parentRefs:
  - name: my-gateway
    namespace: default
  rules:
  - matches:
    - path:
        type: PathPrefix
        value: /details
    backendRefs:
    - name: details
      port: 9080
```

Mit dem folgenden Befehl können Sie die angewendete Cilium-Envoy-Konfiguration anzeigen.

```
kubectl get cec cilium-gateway-my-gateway -o yaml
```

Mit dem folgenden Befehl können Sie den Envoy-Listener-Port für den `cilium-gateway-my-gateway`-Service abrufen. In diesem Beispiel lautet der gemeinsame Listener-Port `8111`.

```
kubectl get cec cilium-gateway-my-gateway -o jsonpath={.spec.services[0].ports[0]}
```

3. Rufen Sie die IP-Adressen Ihrer Knoten in Ihrem Cluster ab.

```
kubectl get nodes -o wide
```

```
NAME                   STATUS   ROLES    AGE   VERSION               INTERNAL-IP
 EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION       CONTAINER-RUNTIME
mi-026d6a261e355fba7   Ready    <none>   23h   v1.32.3-eks-473151a   10.80.146.150
 <none>        Ubuntu 22.04.5 LTS   5.15.0-142-generic   containerd://1.7.27
```

```
mi-082f73826a163626e   Ready    <none>   23h   v1.32.3-eks-473151a   10.80.146.32
 <none>        Ubuntu 22.04.4 LTS   5.15.0-142-generic   containerd://1.7.27
mi-09183e8a3d755abf6   Ready    <none>   23h   v1.32.3-eks-473151a   10.80.146.33
 <none>        Ubuntu 22.04.4 LTS   5.15.0-142-generic   containerd://1.7.27
mi-0d78d815980ed202d   Ready    <none>   23h   v1.32.3-eks-473151a   10.80.146.97
 <none>        Ubuntu 22.04.4 LTS   5.15.0-142-generic   containerd://1.7.27
mi-0daa253999fe92daa   Ready    <none>   23h   v1.32.3-eks-473151a   10.80.146.100
 <none>        Ubuntu 22.04.4 LTS   5.15.0-142-generic   containerd://1.7.27
```

4. Greifen Sie über die IP-Adressen Ihrer Knoten und den Listener-Port für die `cilium-gateway-my-gateway`-Ressource auf den Service zu. Im nachfolgenden Beispiel lautet die verwendete Knoten-IP-Adresse `10.80.146.32` und der Listener-Port `8111`. Ersetzen Sie diese durch die Werte für Ihre Umgebung.

```
curl -s http://10.80.146.32:8111/details/1 | jq
```

```
{
  "id": 1,
  "author": "William Shakespeare",
  "year": 1595,
  "type": "paperback",
  "pages": 200,
  "publisher": "PublisherA",
  "language": "English",
  "ISBN-10": "1234567890",
  "ISBN-13": "123-1234567890"
}
```

Ingress

Aufgrund eines vorgelagerten Cilium-Problems (#34028) erfordert Cilium Ingress im Host-Netzwerkmodus die Verwendung von `loadbalancerMode: shared`, wodurch ein einzelner Service vom Typ ClusterIP für alle Ingress-Ressourcen im Cluster erstellt wird. Wenn Sie einen bevorzugten Port (unter 1023) für die Ingress-Ressource verwenden, finden Sie Anweisungen dazu in der Cilium-Dokumentation.

1. Erstellen Sie eine Datei mit dem Namen `cilium-ingress-host-network.yaml` und dem folgenden Inhalt.

```
ingressController:
```

```
  enabled: true
  loadbalancerMode: shared
  # This is a workaround for the upstream Cilium issue
  service:
    externalTrafficPolicy: null
    type: ClusterIP
  hostNetwork:
    enabled: true
    # ensure the port does not conflict with other services on the node
    sharedListenerPort: 8111
    # uncomment to restrict nodes where Envoy proxies run on the host network
    # nodes:
    #   matchLabels:
    #     role: ingress
```

2. Wenden Sie die Cilium Ingress-Konfiguration des Host-Netzwerks auf Ihren Cluster an.

```
helm upgrade cilium oci://public.ecr.aws/eks/cilium/cilium \
  --namespace kube-system \
  --reuse-values \
  --set operator.rollOutPods=true \
  -f cilium-ingress-host-network.yaml
```

Falls Sie die Ingress-Ressource noch nicht erstellt haben, können Sie sie erstellen, indem Sie die folgende Ingress-Definition auf Ihren Cluster anwenden. Die nachfolgende Ingress-Definition verwendet die in the section called “Cilium Ingress bereitstellen” beschriebene Istio-Bookinfo-Beispielanwendung.

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-ingress
  namespace: default
spec:
  ingressClassName: cilium
  rules:
  - http:
      paths:
      - backend:
          service:
            name: details
            port:
```

```
              number: 9080
        path: /details
        pathType: Prefix
```

Mit dem folgenden Befehl können Sie die angewendete Cilium-Envoy-Konfiguration anzeigen.

```
kubectl get cec -n kube-system cilium-ingress -o yaml
```

Mit dem folgenden Befehl können Sie den Envoy-Listener-Port für den `cilium-ingress`-Service abrufen. In diesem Beispiel lautet der gemeinsame Listener-Port `8111`.

```
kubectl get cec -n kube-system cilium-ingress -o
 jsonpath={.spec.services[0].ports[0]}
```

3. Rufen Sie die IP-Adressen Ihrer Knoten in Ihrem Cluster ab.

```
kubectl get nodes -o wide
```

```
NAME                   STATUS   ROLES    AGE   VERSION               INTERNAL-IP
 EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION       CONTAINER-RUNTIME
mi-026d6a261e355fba7   Ready    <none>   23h   v1.32.3-eks-473151a   10.80.146.150
 <none>        Ubuntu 22.04.5 LTS   5.15.0-142-generic   containerd://1.7.27
mi-082f73826a163626e   Ready    <none>   23h   v1.32.3-eks-473151a   10.80.146.32
 <none>        Ubuntu 22.04.4 LTS   5.15.0-142-generic   containerd://1.7.27
mi-09183e8a3d755abf6   Ready    <none>   23h   v1.32.3-eks-473151a   10.80.146.33
 <none>        Ubuntu 22.04.4 LTS   5.15.0-142-generic   containerd://1.7.27
mi-0d78d815980ed202d   Ready    <none>   23h   v1.32.3-eks-473151a   10.80.146.97
 <none>        Ubuntu 22.04.4 LTS   5.15.0-142-generic   containerd://1.7.27
mi-0daa253999fe92daa   Ready    <none>   23h   v1.32.3-eks-473151a   10.80.146.100
 <none>        Ubuntu 22.04.4 LTS   5.15.0-142-generic   containerd://1.7.27
```

4. Greifen Sie über die IP-Adressen Ihrer Knoten und das `sharedListenerPort` für die `cilium-ingress`-Ressource auf den Service zu. Im nachfolgenden Beispiel lautet die verwendete Knoten-IP-Adresse `10.80.146.32` und der Listener-Port `8111`. Ersetzen Sie diese durch die Werte für Ihre Umgebung.

```
curl -s http://10.80.146.32:8111/details/1 | jq
```

```
{
  "id": 1,
```

```
  "author": "William Shakespeare",
  "year": 1595,
  "type": "paperback",
  "pages": 200,
  "publisher": "PublisherA",
  "language": "English",
  "ISBN-10": "1234567890",
  "ISBN-13": "123-1234567890"
}
```

Konfiguration von Services vom Typ LoadBalancer für Hybridknoten

In diesem Thema wird beschrieben, wie Sie den Layer 4 (L4)-Lastenausgleich für Anwendungen konfigurieren, die auf Amazon EKS Hybrid Nodes ausgeführt werden. Kubernetes-Services vom Typ LoadBalancer werden verwendet, um Kubernetes-Anwendungen außerhalb des Clusters verfügbar zu machen. Services vom Typ LoadBalancer werden oft mit einer physischen Load-Balancer-Infrastruktur in der Cloud oder in einer On-Premises-Umgebung verwendet, um den Datenverkehr der Workload zu bedienen. Diese Load-Balancer-Infrastruktur wird in der Regel mit einem umgebungsspezifischen Controller bereitgestellt.

AWS unterstützt AWS Network Load Balancer (NLB) und Cilium für Services vom Typ LoadBalancer, die in EKS-Hybridknoten ausgeführt werden. Die Entscheidung, ob NLB oder Cilium verwendet wird, hängt von der Quelle des Anwendungs-Datenverkehrs ab. Wenn der Anwendungsverkehr aus einer AWS-Region stammt, empfiehlt AWS die Verwendung von AWS NLB und dem AWS Load Balancer Controller. Wenn der Anwendungs-Datenverkehr aus der lokalen On-Premises- oder Edge-Umgebung stammt, empfiehlt AWS die Verwendung der integrierten Load-Balancing-Funktionen von Cilium, die mit oder ohne Load Balancer-Infrastruktur in Ihrer Umgebung verwendet werden können.

Informationen zum Load Balancing für den Anwendungs-Datenverkehr der Layer 7 (L7) finden Sie unter the section called “Ingress konfigurieren”. Allgemeine Informationen zum Load Balancing mit EKS finden Sie unter Bewährte Methoden für den Load Balancing.

AWS Network Load Balancer

Sie können den AWS Load Balancer Controller und NLB mit dem Zieltyp `ip` für Workloads verwenden, die in Hybridknoten ausgeführt werden. Bei Verwendung des Zieltyps `ip` leitet NLB den Datenverkehr direkt an die Pods weiter und umgeht dabei den Netzwerkpfad der Service-Ebene. Damit NLB die Pod-IP-Ziele auf Hybridknoten erreichen kann, müssen Ihre lokalen Pod-CIDRs in Ihrem On-Premises-Netzwerk routingfähig sein. Darüber hinaus verwendet der AWS Load Balancer

Controller Webhooks und erfordert eine direkte Kommunikation von der EKS-Steuerebene. Weitere Informationen finden Sie unter the section called “Webhooks konfigurieren”.

- Weitere Informationen zu den Anforderungen für die Subnetzkonfiguration finden Sie unter the section called “Netzwerk-Load-Balancer”. Weitere Informationen zum AWS Network Load Balancer und AWS Load Balancer Controller finden Sie unter the section called “Installation mit Helm” und Bewährte Methoden für Load Balancing.
- Informationen zu Konfigurationen, die auf Services vom Typ `LoadBalancer` mit AWS Network Load Balancer angewendet werden können, finden Sie unter Konfigurationen für AWS Load Balancer Controller NLB.

Voraussetzungen

- Cilium wurde gemäß den Anweisungen in the section called “CNI konfigurieren” installiert.
- Cilium-BGP-Steuerebene gemäß den Anweisungen in the section called “BGP konfigurieren” aktiviert. Wenn Sie BGP nicht verwenden möchten, müssen Sie eine alternative Methode verwenden, um die CIDRs Ihres On-Premises-Pods in Ihrem On-Premises-Netzwerk routingfähig zu machen. Weitere Informationen finden Sie unter the section called “Routingfähige Fern-Pod-CIDRs”.
- Helm ist in Ihrer Befehlszeilenumgebung installiert. Weitere Informationen finden Sie unter Anweisungen zur Einrichtung von Helm.
- eksctl in Ihrer Befehlszeilenumgebung installiert ist. Lesen Sie die Anweisungen zur Einrichtung von eksctl.

Verfahren

1. Laden Sie eine IAM-Richtlinie für den AWS-Lastenverteilungs-Controller herunter, die es ihm ermöglicht, in Ihrem Namen Aufrufe an AWS-APIs zu tätigen.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-
controller/refs/heads/main/docs/install/iam_policy.json
```

2. Erstellen Sie eine IAM-Richtlinie mit der im vorherigen Schritt heruntergeladenen Richtlinie.

```
aws iam create-policy \
    --policy-name AWSLoadBalancerControllerIAMPolicy \
    --policy-document file://iam_policy.json
```

3. Ersetzen Sie die Werte für Cluster-Name (`CLUSTER_NAME`), AWS-Region (`AWS_REGION`) und AWS-Konto-ID (`AWS_ACCOUNT_ID`) durch Ihre Einstellungen und führen Sie den folgenden Befehl aus.

```
eksctl create iamserviceaccount \
    --cluster=CLUSTER_NAME \
    --namespace=kube-system \
    --name=aws-load-balancer-controller \
    --attach-policy-arn=arn:aws:iam::AWS_ACCOUNT_ID:policy/
AWSLoadBalancerControllerIAMPolicy \
    --override-existing-serviceaccounts \
    --region AWS_REGION \
    --approve
```

4. Fügen Sie das Helm-Chart-Repository eks-charts hinzu. AWS verwaltet dieses Repository auf GitHub.

```
helm repo add eks https://aws.github.io/eks-charts
```

5. Aktualisieren Sie Ihr lokales Helm-Repository, um sicherzustellen, dass Sie über die aktuellsten Diagramme verfügen.

```
helm repo update eks
```

6. Installieren Sie den AWS-Lastenverteilungs-Controller. Ersetzen Sie die Werte für Cluster-Name (`CLUSTER_NAME`), AWS-Region (`AWS_REGION`), VPC-ID (`VPC_ID`), und die Helm-Chart-Version des AWS Load Balancer Controller (`AWS_LBC_HELM_VERSION`) durch Ihre Einstellungen. Die neueste Version des Helm-Charts finden Sie, indem Sie `helm search repo eks/aws-load-balancer-controller --versions` ausführen. Wenn Sie einen Cluster im kombinierten Modus mit Hybridknoten und Knoten in der AWS Cloud betreiben, können Sie AWS Load Balancer Controller auf Cloud-Knoten ausführen, indem Sie die Anweisungen unter the section called “AWS-Lastenverteilungs-Controller” befolgen.

```
helm install aws-load-balancer-controller eks/aws-load-balancer-controller \
  -n kube-system \
  --version AWS_LBC_HELM_VERSION \
  --set clusterName=CLUSTER_NAME \
  --set region=AWS_REGION \
  --set vpcId=VPC_ID \
  --set serviceAccount.create=false \
```

```
  --set serviceAccount.name=aws-load-balancer-controller
```

7. Überprüfen Sie, ob der AWS Load Balancer Controller erfolgreich installiert wurde.

```
kubectl get -n kube-system deployment aws-load-balancer-controller
```

```
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
aws-load-balancer-controller   2/2     2            2           84s
```

8. Definieren Sie eine Beispielanwendung in einer Datei mit dem Namen `tcp-sample-app.yaml`. Das folgende Beispiel verwendet eine einfache NGINX-Bereitstellung mit einem TCP-Port.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: tcp-sample-app
  namespace: default
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: public.ecr.aws/nginx/nginx:1.23
          ports:
            - name: tcp
              containerPort: 80
```

9. Wenden Sie die Bereitstellung auf Ihren Cluster an.

```
kubectl apply -f tcp-sample-app.yaml
```

10. Definieren Sie einen Service vom Typ LoadBalancer für die Bereitstellung in einer Datei mit dem Namen `tcp-sample-service.yaml`.

```
apiVersion: v1
```

```
kind: Service
metadata:
  name: tcp-sample-service
  namespace: default
  annotations:
    service.beta.kubernetes.io/aws-load-balancer-type: external
    service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: ip
    service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing
spec:
  ports:
    - port: 80
      targetPort: 80
      protocol: TCP
  type: LoadBalancer
  selector:
    app: nginx
```

11. Wenden Sie die Servicekonfiguration auf Ihren Cluster an.

```
kubectl apply -f tcp-sample-service.yaml
```

12. Die Bereitstellung des NLB für den Service kann einige Minuten dauern. Sobald der NLB bereitgestellt ist, wird dem Service eine Adresse zugewiesen, die dem DNS-Namen der NLB-Bereitstellung entspricht.

```
kubectl get svc tcp-sample-service
```

```
NAME                TYPE           CLUSTER-IP       EXTERNAL-IP
                                                PORT(S)        AGE
tcp-sample-service   LoadBalancer   172.16.115.212   k8s-default-tcpsampl-xxxxxxxxxx-
xxxxxxxxxxxxxxxx.elb.<region>.amazonaws.com   80:30396/TCP   8s
```

13. Greifen Sie über die Adresse des NLB auf den Service zu.

```
curl k8s-default-tcpsampl-xxxxxxxxxx-xxxxxxxxxxxxxxxx.elb.<region>.amazonaws.com
```

Unten sehen Sie eine Beispielausgabe.

```
<!DOCTYPE html>
<html>
<head>
```

```
<title>Welcome to nginx!</title>
[...]
```

14 Bereinigen Sie die -Ressourcen, die Sie erstellt haben.

```
kubectl delete -f tcp-sample-service.yaml
kubectl delete -f tcp-sample-app.yaml
```

Cilium-Load-Balancing im Cluster

Cilium kann als Cluster-interner Load Balancer für Workloads verwendet werden, die in EKS-Hybridknoten ausgeführt werden. Dies kann für Umgebungen nützlich sein, die nicht über eine Load-Balancer-Infrastruktur verfügen. Die Load-Balancing-Funktionen von Cilium basieren auf einer Kombination von Cilium-Features, darunter kube-proxy-Ersatz, Load Balancer IP Address Management (IPAM) und BGP-Steuerebene. Die Aufgaben dieser Feature werden nachfolgend näher erläutert:

- Cilium-kube-proxy-Ersatz: Verarbeitet die Weiterleitung des Service-Datenverkehrs an Backend-Pods.
- Cilium Load Balancer IPAM: Verwaltet IP-Adressen, die Services vom Typ `LoadBalancer` zugewiesen werden können.
- Cilium-BGP-Steuerebene: Gibt vom Load Balancer IPAM dem On-Premises-Netzwerk zugewiesene IP-Adressen bekannt.

Wenn Sie den kube-proxy-Ersatz von Cilium nicht verwenden, können Sie dennoch Cilium Load Balancer IPAM und BGP-Steuerebene nutzen, um IP-Adressen für Services vom Typ LoadBalancer zuzuweisen und zuzuordnen. Wenn Sie den kube-proxy-Ersatz von Cilium nicht verwenden, wird der Lastenausgleich für Services zu Backend-Pods in EKS standardmäßig durch kube-proxy- und iptables-Regeln übernommen.

Voraussetzungen

- Cilium wurde gemäß den Anweisungen in the section called “CNI konfigurieren” mit oder ohne aktiviertem kube-proxy-Ersatz installiert. Der kube-proxy-Ersatz von Cilium erfordert ein Betriebssystem mit einem Linux-Kernel mindestens der aktuellen Version v4.19.57, v5.1.16 oder v5.2.0. Alle aktuellen Versionen der für die Verwendung mit Hybridknoten unterstützten Betriebssysteme erfüllen dieses Kriterium, mit Ausnahme von Red Hat Enterprise Linux (RHEL) 8.x.

- Cilium-BGP-Steuerebene gemäß den Anweisungen in the section called “BGP konfigurieren” aktiviert. Wenn Sie BGP nicht verwenden möchten, müssen Sie eine alternative Methode verwenden, um die CIDRs Ihres On-Premises-Pods in Ihrem On-Premises-Netzwerk routingfähig zu machen. Weitere Informationen finden Sie unter the section called “Routingfähige Fern-Pod-CIDRs”.
- Helm ist in Ihrer Befehlszeilenumgebung installiert. Weitere Informationen finden Sie unter Anweisungen zur Einrichtung von Helm.

Verfahren

1. Erstellen Sie eine Datei mit dem Namen `cilium-lbip-pool-loadbalancer.yaml` mit einer `CiliumLoadBalancerIPPool`-Ressource, um den Load Balancer-IP-Adressbereich für Ihre Service vom Typ LoadBalancer zu konfigurieren.
   - Ersetzen Sie `LB_IP_CIDR` durch den IP-Adressbereich, der für die IP-Adressen des Load Balancers verwendet werden soll. Um eine einzelne IP-Adresse auszuwählen, verwenden Sie ein `/32` CIDR. Weitere Informationen finden Sie unter LoadBalancer-IP-Adressverwaltung in der Cilium-Dokumentation.
   - Das `serviceSelector`-Feld wird so konfiguriert, dass es mit dem Namen des Service übereinstimmt, den Sie in einem späteren Schritt erstellen. Mit dieser Konfiguration werden IPs aus diesem Pool nur Services mit dem Namen `tcp-sample-service` zugewiesen.

   ```
   apiVersion: cilium.io/v2alpha1
   kind: CiliumLoadBalancerIPPool
   metadata:
     name: tcp-service-pool
   spec:
     blocks:
     - cidr: "LB_IP_CIDR"
     serviceSelector:
       matchLabels:
         io.kubernetes.service.name: tcp-sample-service
   ```

2. Wenden Sie die `CiliumLoadBalancerIPPool`-Ressource auf Ihren Cluster an.

   ```
   kubectl apply -f cilium-lbip-pool-loadbalancer.yaml
   ```

3. Stellen Sie sicher, dass im Pool mindestens eine IP-Adresse verfügbar ist.

```
kubectl get ciliumloadbalancerippools.cilium.io
```

```
NAME               DISABLED   CONFLICTING   IPS AVAILABLE   AGE
tcp-service-pool   false      False         1               24m
```

4. Erstellen Sie eine Datei mit dem Namen `cilium-bgp-advertisement-loadbalancer.yaml` mit einer `CiliumBGPAdvertisement`-Ressource, um die IP-Adresse des Load Balancers für den Service bekannt zu geben, den Sie im nächsten Schritt erstellen. Wenn Sie Cilium BGP nicht verwenden, können Sie diesen Schritt überspringen. Die für Ihren Service verwendete IP-Adresse des Load Balancers muss in Ihrem On-Premises-Netzwerk routingfähig sein, damit Sie den Service im letzten Schritt abfragen können.
   - Das `advertisementType`-Feld ist auf `Service` gesetzt und `service.addresses` ist auf `LoadBalancerIP` gesetzt, um den `LoadBalancerIP` für Services vom Typ `LoadBalancer` bekannt zu geben.
   - Das `selector`-Feld wird so konfiguriert, dass es mit dem Namen des Service übereinstimmt, den Sie in einem späteren Schritt erstellen. Mit dieser Konfiguration werden nur `LoadBalancerIP` für Services mit dem Namen `tcp-sample-service` bekannt gegeben.

   ```
   apiVersion: cilium.io/v2alpha1
   kind: CiliumBGPAdvertisement
   metadata:
     name: bgp-advertisement-tcp-service
     labels:
       advertise: bgp
   spec:
     advertisements:
       - advertisementType: "Service"
         service:
           addresses:
             - LoadBalancerIP
         selector:
           matchLabels:
             io.kubernetes.service.name: tcp-sample-service
   ```

5. Wenden Sie die `CiliumBGPAdvertisement`-Ressource auf Ihren Cluster an. Wenn Sie Cilium BGP nicht verwenden, können Sie diesen Schritt überspringen.

   ```
   kubectl apply -f cilium-bgp-advertisement-loadbalancer.yaml
   ```

6. Definieren Sie eine Beispielanwendung in einer Datei mit dem Namen `tcp-sample-app.yaml`. Das folgende Beispiel verwendet eine einfache NGINX-Bereitstellung mit einem TCP-Port.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: tcp-sample-app
  namespace: default
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: public.ecr.aws/nginx/nginx:1.23
          ports:
            - name: tcp
              containerPort: 80
```

7. Wenden Sie die Bereitstellung auf Ihren Cluster an.

```
kubectl apply -f tcp-sample-app.yaml
```

8. Definieren Sie einen Service vom Typ LoadBalancer für die Bereitstellung in einer Datei mit dem Namen `tcp-sample-service.yaml`.
   - Mit der `lbipam.cilium.io/ips`-Annotation zum Serviceobjekt können Sie eine bestimmte IP-Adresse aus dem IP-Pool des Load Balancers anfordern. Sie können diese Annotation entfernen, wenn Sie keine bestimmte IP-Adresse für den Service anfordern möchten.
   - Das Spezifikationsfeld `loadBalancerClass` ist erforderlich, um zu verhindern, dass der alte AWS-Cloud-Anbieter einen Classic Load Balancer für den Service erstellt. Im nachfolgenden Beispiel ist dies auf `io.cilium/bgp-control-plane` konfiguriert, um die BGP-Steuerebene von Cilium als Load-Balancer-Klasse zu verwenden. Dieses Feld kann alternativ auf `io.cilium/l2-announcer` konfiguriert werden, um das L2-Ankündigungsfeature von Cilium zu verwenden (derzeit in der Betaphase und nicht offiziell von AWS unterstützt).

```
apiVersion: v1
kind: Service
metadata:
  name: tcp-sample-service
  namespace: default
  annotations:
    lbipam.cilium.io/ips: "LB_IP_ADDRESS"
spec:
  loadBalancerClass: io.cilium/bgp-control-plane
  ports:
    - port: 80
      targetPort: 80
      protocol: TCP
  type: LoadBalancer
  selector:
    app: nginx
```

9. Wenden Sie den Service auf Ihren Cluster an. Der Service wird mit einer externen IP-Adresse erstellt, die Sie für den Zugriff auf die Anwendung verwenden können.

```
kubectl apply -f tcp-sample-service.yaml
```

10. Überprüfen Sie, ob der Service erfolgreich erstellt wurde und ihm eine IP aus dem im vorherigen Schritt erstellten `CiliumLoadBalancerIPPool` zugewiesen wurde.

```
kubectl get svc tcp-sample-service
```

```
NAME                TYPE           CLUSTER-IP      EXTERNAL-IP     PORT(S)
 AGE
tcp-sample-service   LoadBalancer   172.16.117.76   LB_IP_ADDRESS   80:31129/TCP
 14m
```

11. Wenn Sie Cilium im kube-proxy-Ersatzmodus verwenden, können Sie mit dem folgenden Befehl bestätigen, dass Cilium das Load Balancing für den Service übernimmt. In der folgenden Ausgabe sind die `10.86.2.`x-Adressen die Pod-IP-Adressen der Backend-Pods für den Service.

```
kubectl -n kube-system exec ds/cilium -- cilium-dbg service list
```

```
ID   Frontend              Service Type   Backend
...
```

```
41   LB_IP_ADDRESS:80/TCP   LoadBalancer   1 => 10.86.2.76:80/TCP (active)
                                           2 => 10.86.2.130:80/TCP (active)
                                           3 => 10.86.2.141:80/TCP (active)
```

12. Stellen Sie sicher, dass Cilium die IP-Adresse über BGP an das On-Premises-Netzwerk bekannt gibt. Im folgenden Beispiel gibt es fünf Hybridknoten, die jeweils die `LB_IP_ADDRESS` für den `tcp-sample-service`-Service im On-Premises-Netzwerk bekannt geben.

```
Node                    VRouter      Prefix             NextHop   Age     Attrs
mi-026d6a261e355fba7    NODES_ASN
                   LB_IP_ADDRESS/32   0.0.0.0   12m3s   [{Origin: i} {Nexthop:
 0.0.0.0}]
mi-082f73826a163626e    NODES_ASN
                   LB_IP_ADDRESS/32   0.0.0.0   12m3s   [{Origin: i} {Nexthop:
 0.0.0.0}]
mi-09183e8a3d755abf6    NODES_ASN
                   LB_IP_ADDRESS/32   0.0.0.0   12m3s   [{Origin: i} {Nexthop:
 0.0.0.0}]
mi-0d78d815980ed202d    NODES_ASN
                   LB_IP_ADDRESS/32   0.0.0.0   12m3s   [{Origin: i} {Nexthop:
 0.0.0.0}]
mi-0daa253999fe92daa    NODES_ASN
                   LB_IP_ADDRESS/32   0.0.0.0   12m3s   [{Origin: i} {Nexthop:
 0.0.0.0}]
```

13. Greifen Sie über die zugewiesene IP-Adresse des Load Balancers auf den Service zu.

```
curl LB_IP_ADDRESS
```

Unten sehen Sie eine Beispielausgabe.

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
[...]
```

14. Bereinigen Sie die -Ressourcen, die Sie erstellt haben.

```
kubectl delete -f tcp-sample-service.yaml
kubectl delete -f tcp-sample-app.yaml
```

```
kubectl delete -f cilium-lb-ip-pool.yaml
kubectl delete -f cilium-bgp-advertisement.yaml
```

Konfiguration von Kubernetes-Netzwerkrichtlinien für Hybridknoten

AWS unterstützt Kubernetes-Netzwerkrichtlinien (Layer 3/Layer 4) für Pod-Ein- und -Ausgangs-Datenverkehr bei Verwendung von Cilium als CNI mit EKS-Hybridknoten. Wenn Sie EKS-Cluster mit Knoten in der AWS Cloud ausführen, unterstützt AWS das Amazon VPC CNI für Kubernetes-Netzwerkrichtlinien.

Dieses Thema beschreibt die Konfiguration von Cilium- und Kubernetes-Netzwerkrichtlinien mit EKS-Hybridknoten. Ausführliche Informationen zu Kubernetes-Netzwerkrichtlinien finden Sie unter Kubernetes-Netzwerkrichtlinien in der Kubernetes-Dokumentation.

Netzwerkrichtlinien konfigurieren

Überlegungen

- AWS unterstützt die vorgelagerten Kubernetes-Netzwerkrichtlinien und -Spezifikationen für Pod-Ingress und -Egress. AWS unterstützt derzeit `CiliumNetworkPolicy` oder `CiliumClusterwideNetworkPolicy` nicht.
- Der `policyEnforcementMode`-Helm-Wert kann verwendet werden, um das Standardverhalten der Cilium-Richtliniendurchsetzung zu steuern. Das Standardverhalten lässt den gesamten ausgehenden und eingehenden Datenverkehr zu. Wenn ein Endpunkt durch eine Netzwerkrichtlinie ausgewählt wird, wechselt er in einen Standard-Verweigerungszustand, in dem nur ausdrücklich zugelassener Datenverkehr zugelassen wird. Weitere Informationen zum Standard-Richtlinienmodus und zu den Modi zur Durchsetzung von Richtlinien finden Sie in der Cilium-Dokumentation.
- Wenn Sie `policyEnforcementMode` für eine vorhandene Cilium-Installation ändern, müssen Sie den Cilium Agent DaemonSet neu starten, um den neuen Modus zur Durchsetzung von Richtlinien anzuwenden.
- Verwenden Sie `namespaceSelector` und `podSelector`, um Datenverkehr zu/von Namespaces und Pods mit übereinstimmenden Labels zuzulassen oder zu verweigern. Der `namespaceSelector` und `podSelector` kann mit `matchLabels` oder `matchExpressions` verwendet werden, um Namespaces und Pods basierend auf ihren Labels auszuwählen.
- Verwenden Sie `ingress.ports` und `egress.ports`, um Datenverkehr zu/von Ports und Protokollen zuzulassen oder zu verweigern.

- Das `ipBlock`-Feld kann nicht verwendet werden, um Datenverkehr zu/von Pod-IP-Adressen selektiv zuzulassen oder zu verweigern (#9209). Die Verwendung von `ipBlock`-Selektoren für Knoten-IPs ist ein Beta-Feature in Cilium und wird von AWS nicht unterstützt.
- Informationen zu den verfügbaren Feldern für Kubernetes-Netzwerkrichtlinien finden Sie in der NetworkPolicy-Ressource in der Kubernetes-Dokumentation.

Voraussetzungen

- Cilium wurde gemäß den Anweisungen in the section called “CNI konfigurieren” installiert.
- Helm ist in Ihrer Befehlszeilenumgebung installiert. Weitere Informationen finden Sie unter Anweisungen zur Einrichtung von Helm.

Verfahren

Mit dem folgenden Verfahren werden Netzwerkrichtlinien für eine Beispiel-Microservices-Anwendung eingerichtet, sodass Komponenten nur mit anderen Komponenten kommunizieren können, die für die Funktion der Anwendung erforderlich sind. Das Verfahren verwendet die Istio Bookinfo-Beispielanwendung für Microservices.

Die Bookinfo-Anwendung besteht aus vier separaten Microservices mit den folgenden Beziehungen:

- productpage. Der Microservice „productpage“ ruft die Details auf und überprüft die Microservices, um die Seite zu füllen.
- details. Der Microservice „details“ enthält Buchinformationen.
- reviews. Der Microservice „reviews“ enthält Buchbewertungen. Er ruft außerdem den Microservice „ratings“ auf.
- ratings. Der Microservice „ratings“ enthält Informationen zur Buchbewertung, die einer Buchrezension beigefügt sind.

1. Erstellen Sie die Beispielanwendung.

```
kubectl apply -f https://raw.githubusercontent.com/istio/istio/refs/heads/master/
samples/bookinfo/platform/kube/bookinfo.yaml
```

2. Stellen Sie sicher, dass die Anwendung erfolgreich ausgeführt wird, und notieren Sie sich die Pod-IP-Adresse für den Mikroservice „productpage“. Diese Pod-IP-Adresse werden Sie in den folgenden Schritten verwenden, um jeden Microservice abzufragen.

```
kubectl get pods -o wide
```

```
NAME                              READY   STATUS    RESTARTS   AGE   IP
 NODE
details-v1-766844796b-9wff2       1/1     Running   0          7s    10.86.3.7
 mi-0daa253999fe92daa
productpage-v1-54bb874995-lwfgg   1/1     Running   0          7s    10.86.2.193
 mi-082f73826a163626e
ratings-v1-5dc79b6bcd-59njm       1/1     Running   0          7s    10.86.2.232
 mi-082f73826a163626e
reviews-v1-598b896c9d-p2289       1/1     Running   0          7s    10.86.2.47
 mi-026d6a261e355fba7
reviews-v2-556d6457d-djktc        1/1     Running   0          7s    10.86.3.58
 mi-0daa253999fe92daa
reviews-v3-564544b4d6-g8hh4       1/1     Running   0          7s    10.86.2.69
 mi-09183e8a3d755abf6
```

3. Erstellen Sie einen Pod, der durchgehend zum Testen der Netzwerkrichtlinien verwendet wird. Beachten Sie, dass der Pod im `default`-Namespace mit dem Label `access: true` erstellt wird.

```
kubectl run curl-pod --image=curlimages/curl -i --tty --labels=access=true --
namespace=default --overrides='{"spec": { "nodeSelector": {"eks.amazonaws.com/
compute-type": "hybrid"}}}' -- /bin/sh
```

4. Testen Sie den Zugriff auf den Microservice „productpage“. Im nachfolgenden Beispiel verwenden wir die Pod-IP-Adresse des productpage-Pods (`10.86.2.193`), um den Microservice abzufragen. Ersetzen Sie dies durch die Pod-IP-Adresse des productpage-Pods in Ihrer Umgebung.

```
curl -s http://10.86.2.193:9080/productpage | grep -o "<title>.*</title>"
```

```
<title>Simple Bookstore App</title>
```

5. Sie können den Test-Curl-Pod durch Eingabe von `exit` verlassen und sich erneut mit dem Pod verbinden, indem Sie den folgenden Befehl ausführen.

```
kubectl attach curl-pod -c curl-pod -i -t
```

6. Um die Auswirkungen der Netzwerkrichtlinien in den folgenden Schritten zu demonstrieren, erstellen wir zunächst eine Netzwerkrichtlinie, die den gesamten Datenverkehr für die BookInfo-Microservices verweigert. Erstellen Sie eine Datei mit dem Namen `network-policy-deny-bookinfo.yaml`, welche die Netzwerk-Verweigerungsrichtlinie definiert.

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: deny-bookinfo
  namespace: default
spec:
  podSelector:
    matchExpressions:
    - key: app
      operator: In
      values: ["productpage", "details", "reviews", "ratings"]
  policyTypes:
  - Ingress
  - Egress
```

7. Wenden Sie die Netzwerk-Verweigerungsrichtlinie auf Ihren Cluster an.

```
kubectl apply -f network-policy-default-deny-bookinfo.yaml
```

8. Testen Sie den Zugriff auf die Anwendung BookInfo. Im nachfolgenden Beispiel verwenden wir die Pod-IP-Adresse des productpage-Pods (`10.86.2.193`), um den Microservice abzufragen. Ersetzen Sie dies durch die Pod-IP-Adresse des productpage-Pods in Ihrer Umgebung.

```
curl http://10.86.2.193:9080/productpage --max-time 10
```

```
curl: (28) Connection timed out after 10001 milliseconds
```

9. Erstellen Sie eine Datei mit dem Namen `network-policy-productpage.yaml`, welche die Netzwerkrichtlinie der Produktseite definiert. Die Richtlinie enthält die folgenden Regeln:
   - erlaubt eingehenden Datenverkehr von Pods mit dem Label `access: true` (der im vorherigen Schritt erstellte Curl-Pod).
   - erlaubt ausgehenden TCP-Datenverkehr auf Port `9080` für die Microservices „details“, „reviews“ und „ratings“.

- erlaubt ausgehenden TCP/UDP-Datenverkehr auf Port 53 für CoreDNS, das im `kube-system`-Namespace ausgeführt wird.

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: productpage-policy
  namespace: default
spec:
  podSelector:
    matchLabels:
      app: productpage
  policyTypes:
  - Ingress
  - Egress
  ingress:
  - from:
    - podSelector:
        matchLabels:
          access: "true"
  egress:
  - to:
    - podSelector:
        matchExpressions:
        - key: app
          operator: In
          values: ["details", "reviews", "ratings"]
    ports:
    - port: 9080
      protocol: TCP
  - to:
    - namespaceSelector:
        matchLabels:
          kubernetes.io/metadata.name: kube-system
      podSelector:
        matchLabels:
          k8s-app: kube-dns
    ports:
    - port: 53
      protocol: UDP
    - port: 53
      protocol: TCP
```

10. Wenden Sie die Netzwerkrichtlinie „productpage“ auf Ihren Cluster an.

```
kubectl apply -f network-policy-productpage.yaml
```

11. Stellen Sie eine Verbindung zum Curl-Pod her und testen Sie den Zugriff auf die Bookinfo-Anwendung. Der Zugriff auf den Microservice „productpage“ ist nun erlaubt. Die anderen Microservices werden jedoch weiterhin verweigert, da sie weiterhin der Richtlinie „Netzwerk verweigern“ unterliegen. In den folgenden Beispielen verwenden wir die Pod-IP-Adresse des Productpage-Pods (`10.86.2.193`), um den Microservice abzufragen. Ersetzen Sie dies durch die Pod-IP-Adresse des productpage-Pods in Ihrer Umgebung.

```
kubectl attach curl-pod -c curl-pod -i -t
```

```
curl -s http://10.86.2.193:9080/productpage | grep -o "<title>.*</title>"
<title>Simple Bookstore App</title>
```

```
curl -s http://10.86.2.193:9080/api/v1/products/1
{"error": "Sorry, product details are currently unavailable for this book."}
```

```
curl -s http://10.86.2.193:9080/api/v1/products/1/reviews
{"error": "Sorry, product reviews are currently unavailable for this book."}
```

```
curl -s http://10.86.2.193:9080/api/v1/products/1/ratings
{"error": "Sorry, product ratings are currently unavailable for this book."}
```

12. Erstellen Sie eine Datei mit dem Namen `network-policy-details.yaml`, welche die detaillierte Netzwerkrichtlinie definiert. Die Richtlinie lässt nur eingehenden Datenverkehr vom Microservice „productpage“ zu.

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: details-policy
  namespace: default
spec:
  podSelector:
    matchLabels:
      app: details
```

```
  policyTypes:
  - Ingress
  ingress:
  - from:
    - podSelector:
        matchLabels:
          app: productpage
```

13. Erstellen Sie eine Datei mit dem Namen `network-policy-reviews.yaml`, welche die Richtlinie für das Bewertungsnetzwerk definiert. Die Richtlinie lässt nur eingehenden Datenverkehr vom Microservice „productpage“ zu und nur ausgehenden Datenverkehr zum Microservice „ratings“ und zu CoreDNS.

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: reviews-policy
  namespace: default
spec:
  podSelector:
    matchLabels:
      app: reviews
  policyTypes:
  - Ingress
  - Egress
  ingress:
  - from:
    - podSelector:
        matchLabels:
          app: productpage
  egress:
  - to:
    - podSelector:
        matchLabels:
          app: ratings
  - to:
    - namespaceSelector:
        matchLabels:
          kubernetes.io/metadata.name: kube-system
      podSelector:
        matchLabels:
          k8s-app: kube-dns
    ports:
```

```
    - port: 53
      protocol: UDP
    - port: 53
      protocol: TCP
```

14. Erstellen Sie eine Datei mit dem Namen `network-policy-ratings.yaml`, welche die Richtlinie für das Bewertungsnetzwerk definiert. Die Richtlinie lässt nur eingehenden Datenverkehr von der Produktseite und den Microservices „reviews“ zu.

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: ratings-policy
  namespace: default
spec:
  podSelector:
    matchLabels:
      app: ratings
  policyTypes:
  - Ingress
  ingress:
  - from:
    - podSelector:
        matchExpressions:
        - key: app
          operator: In
          values: ["productpage", "reviews"]
```

15. Wenden Sie die Netzwerkrichtlinien für Details, Rezensionen und Bewertungen auf Ihren Cluster an.

```
kubectl apply -f network-policy-details.yaml
kubectl apply -f network-policy-reviews.yaml
kubectl apply -f network-policy-ratings.yaml
```

16. Stellen Sie eine Verbindung zum Curl-Pod her und testen Sie den Zugriff auf die Bookinfo-Anwendung. In den folgenden Beispielen verwenden wir die Pod-IP-Adresse des Productpage-Pods (`10.86.2.193`), um den Microservice abzufragen. Ersetzen Sie dies durch die Pod-IP-Adresse des productpage-Pods in Ihrer Umgebung.

```
kubectl attach curl-pod -c curl-pod -i -t
```

Testen Sie den Microservice „details“.

```
curl -s http://10.86.2.193:9080/api/v1/products/1
```

```
{"id": 1, "author": "William Shakespeare", "year": 1595, "type": "paperback",
 "pages": 200, "publisher": "PublisherA", "language": "English", "ISBN-10":
 "1234567890", "ISBN-13": "123-1234567890"}
```

Testen Sie den Microservice „reviews“.

```
curl -s http://10.86.2.193:9080/api/v1/products/1/reviews
```

```
{"id": "1", "podname": "reviews-v1-598b896c9d-p2289", "clustername": "null",
 "reviews": [{"reviewer": "Reviewer1", "text": "An extremely entertaining play
 by Shakespeare. The slapstick humour is refreshing!"}, {"reviewer": "Reviewer2",
 "text": "Absolutely fun and entertaining. The play lacks thematic depth when
 compared to other plays by Shakespeare."}]}
```

Testen Sie den Microservice „ratings“.

```
curl -s http://10.86.2.193:9080/api/v1/products/1/ratings
```

```
{"id": 1, "ratings": {"Reviewer1": 5, "Reviewer2": 4}}
```

17. Bereinigen Sie die Ressourcen, die Sie in diesem Verfahren erstellt haben.

```
kubectl delete -f network-policy-deny-bookinfo.yaml
kubectl delete -f network-policy-productpage.yaml
kubectl delete -f network-policy-details.yaml
kubectl delete -f network-policy-reviews.yaml
kubectl delete -f network-policy-ratings.yaml
kubectl delete -f https://raw.githubusercontent.com/istio/istio/refs/heads/master/
samples/bookinfo/platform/kube/bookinfo.yaml
kubectl delete pod curl-pod
```

Konzepte für Hybridknoten

Mit Amazon EKS Hybrid Nodes verbinden Sie physische oder virtuelle Rechner, die in On-Premises- oder Edge-Umgebungen ausgeführt werden, mit Amazon-EKS-Clustern, die in der AWS Cloud ausgeführt werden. Dieser Ansatz bietet zahlreiche Vorteile, führt jedoch auch neue Netzwerkkonzepte und -architekturen für diejenigen ein, die mit der Ausführung von Kubernetes-Clustern in einer einzigen Netzwerkumgebung vertraut sind.

Die folgenden Abschnitte befassen sich eingehend mit den Kubernetes- und Netzwerkkonzepten für EKS-Hybridknoten und erläutern detailliert, wie der Datenverkehr durch die Hybridarchitektur verläuft. Für diese Abschnitte sind grundlegende Kenntnisse über Kubernetes-Netzwerke erforderlich, beispielsweise über die Konzepte von Pods, Knoten, Services, der Kubernetes-Steuerebene, kubelet und kube-proxy.

Wir empfehlen, diese Seiten der Reihe nach zu lesen, beginnend mit the section called “Netzwerkkonzepte”, dann the section called “Kubernetes-Konzepte” und schließlich mit the section called “Datenverkehr-Abläufe”.

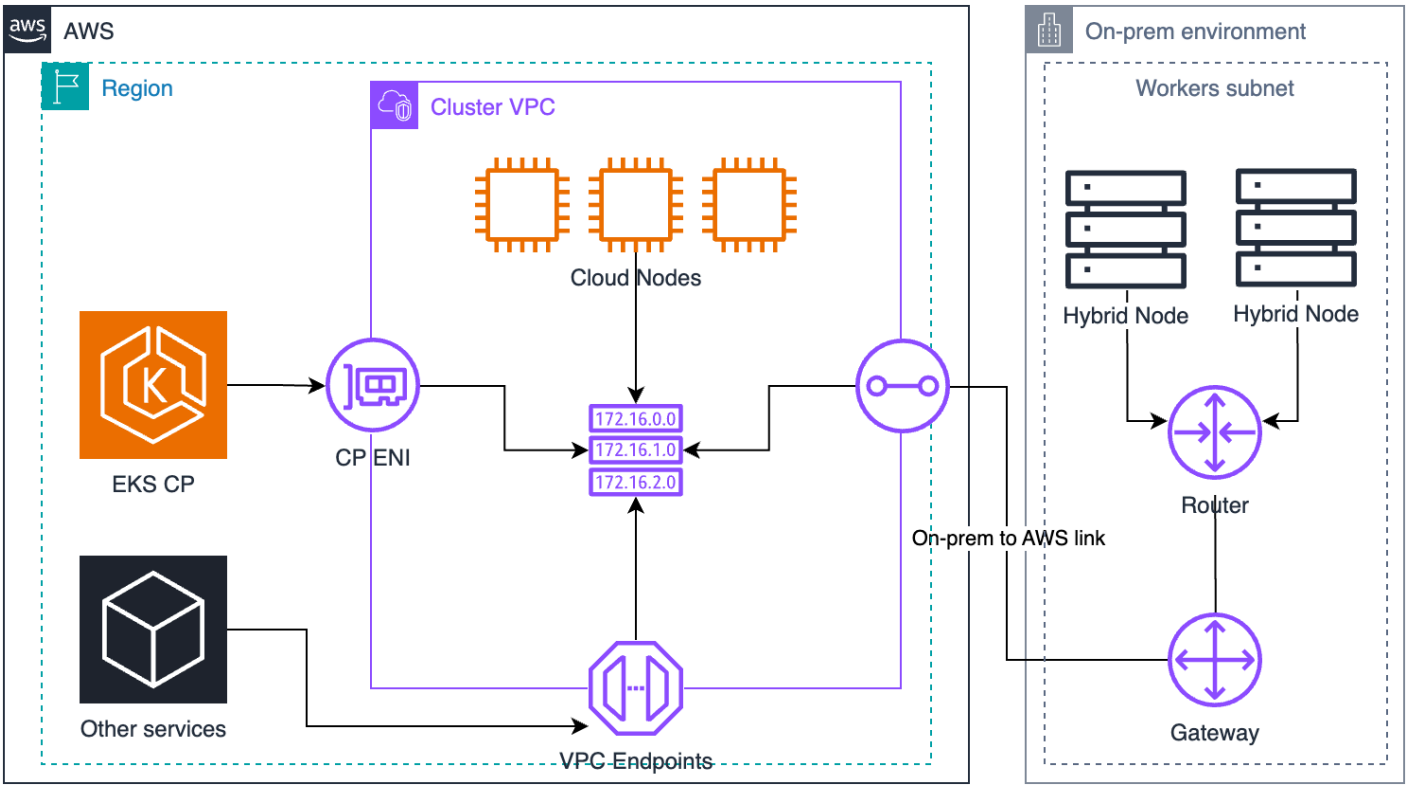
Themen

- Netzwerkkonzepte für Hybridknoten
- Kubernetes-Konzepte für Hybridknoten
- Abläufe des Netzwerk-Datenverkehrs für Hybridknoten

Netzwerkkonzepte für Hybridknoten

In diesem Abschnitt werden die wichtigsten Netzwerkkonzepte und die Einschränkungen erläutert, die Sie bei der Gestaltung Ihrer Netzwerk-Topologie für EKS-Hybridknoten berücksichtigen müssen.

Netzwerkkonzepte für EKS-Hybridknoten



VPC als Netzwerk-Bereich

Der gesamte Datenverkehr, der die Cloud-Grenze überschreitet, wird über Ihre VPC geleitet. Dazu gehört der Verkehr zwischen der EKS-Steuerungsebene oder den Pods, die in AWS Hybridknoten oder Pods laufen, die auf ihnen ausgeführt werden. Sie können sich die VPC Ihres Clusters als Netzwerk-Bereich zwischen Ihren Hybridknoten und dem Rest des Clusters vorstellen. Diese Architektur ermöglicht Ihnen die vollständige Kontrolle über den Datenverkehr und dessen Weiterleitung, überträgt Ihnen jedoch auch die Verantwortung für die korrekte Konfiguration von Routen, Sicherheitsgruppen und Firewalls für die VPC.

EKS-Steuerebene zur VPC

Die EKS-Kontrollebene verbindet Elastic Network Interfaces (ENIs) mit Ihrer VPC. Diese ENIs verarbeiten den Verkehr zum und vom EKS-API-Server. Sie steuern die Platzierung der EKS-Steuerungsebene ENIs bei der Konfiguration Ihres Clusters, da EKS eine Verbindung ENIs zu den Subnetzen herstellt, die Sie bei der Clustererstellung übergeben.

EKS ordnet Sicherheitsgruppen den Gruppen zu ENIs , die EKS Ihren Subnetzen anhängt. Diese Sicherheitsgruppen ermöglichen den Verkehr zur und von der EKS-Steuerebene über die. ENIs Dies

ist wichtig für EKS-Hybridknoten, da Sie den Datenverkehr von den Hybridknoten und den darauf laufenden Pods zur EKS-Steuerebene zulassen müssen ENIs.

Fern-Knoten-Netzwerke

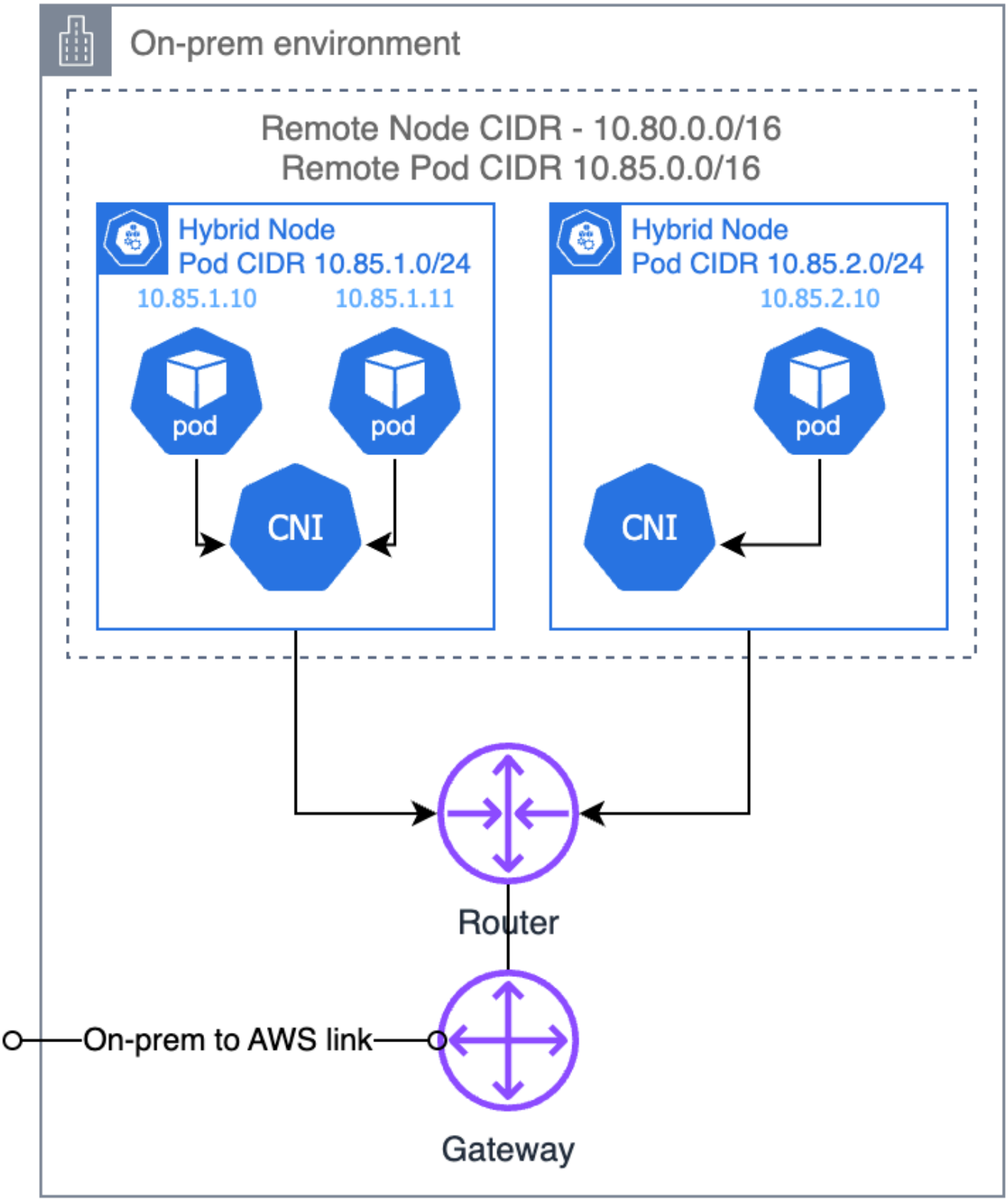
Die Remote-Knoten-Netzwerke, insbesondere der Remote-Knoten CIDRs, sind die Bereiche, die den Maschinen IPs zugewiesen sind, die Sie als Hybridknoten verwenden. Wenn Sie Hybridknoten bereitstellen, befinden sich diese in Ihrem On-Premises-Rechenzentrum oder Edge-Standort, der sich in einer anderen Netzwerk-Domain als die EKS-Steuerebene und die VPC befindet. Jeder Hybridknoten verfügt über eine oder mehrere IP-Adressen aus einem Fern-Knoten-CIDR, der sich von den Subnetzen in Ihrer VPC unterscheidet.

Sie konfigurieren den EKS-Cluster mit diesen Remote-Knoten, CIDRs sodass EKS den gesamten für die Hybridknoten bestimmten Datenverkehr IPs über Ihre Cluster-VPC weiterleiten kann, z. B. Anfragen an die Kubelet-API. Die Verbindungen zur `kubelet` API werden in den Befehlen`kubectl attach`,, `kubectl cp`kubectl exec, `kubectl logs` und verwendet. `kubectl port-forward`

Fern-Pod-Netzwerke

Die Remote-Pod-Netzwerke sind die Bereiche, die den Pods IPs zugewiesen wurden, die auf den Hybridknoten ausgeführt werden. In der Regel konfigurieren Sie Ihr CNI mit diesen Bereichen, und die IP-Adressverwaltungsfunktion (IPAM) des CNI übernimmt die Zuweisung eines Teils dieser Segmente zu jedem Hybridknoten. Wenn Sie einen Pod erstellen, weist das CNI dem Pod eine IP-Adresse aus dem Segment zu, das dem Knoten zugewiesen wurde, auf dem der Pod geplant wurde.

Sie konfigurieren den EKS-Cluster mit diesen Remote-Pods, CIDRs sodass die EKS-Steuerebene weiß, dass sie den gesamten Datenverkehr, der für die Pods bestimmt ist, die auf den Hybridknoten ausgeführt werden, über die VPC Ihres Clusters weiterleitet, z. B. die Kommunikation mit Webhooks.



On-premises zur VPC

Das On-Premises-Netzwerk, das Sie für Hybridknoten verwenden, muss an die VPC weiterleiten, die Sie für Ihren EKS-Cluster verwenden. Es stehen mehrere Network-to-Amazon VPC-Konnektivitätsoptionen zur Verfügung, um Ihr lokales Netzwerk mit einer VPC zu verbinden. Sie können auch Ihre eigene VPN-Lösung verwenden.

Es ist wichtig, dass Sie das Routing auf der AWS Cloud-Seite in der VPC und in Ihrem lokalen Netzwerk korrekt konfigurieren, sodass beide Netzwerke den richtigen Datenverkehr über die Verbindung für die beiden Netzwerke weiterleiten.

In der VPC muss der gesamte Datenverkehr, der an die Fern-Knoten- und Fern-Pod-Netzwerke geht, über die Verbindung zu Ihrem On-Premises-Netzwerk (als „Gateway“ bezeichnet) weitergeleitet werden. Wenn einige Ihrer Subnetze unterschiedliche Routen-Tabellen aufweisen, müssen Sie jede Routen-Tabelle mit den Routen für Ihre Hybridknoten und die darauf ausgeführten Pods konfigurieren. Dies gilt für die Subnetze, an die die EKS-Steuerebene ENIs angeschlossen ist, und für Subnetze, die EC2 Knoten oder Pods enthalten, die mit Hybridknoten kommunizieren müssen.

In Ihrem lokalen Netzwerk müssen Sie Ihr Netzwerk so konfigurieren, dass der Datenverkehr zur und von der VPC Ihres EKS-Clusters und den anderen für Hybridknoten erforderlichen AWS Diensten zugelassen wird. Der Datenverkehr für den EKS-Cluster durchläuft das Gateway in beide Richtungen.

Netzwerkbeschränkungen

Vollständig weitergeleitetes Netzwerk

Die wichtigste Einschränkung besteht darin, dass die EKS-Steuerebene und alle Knoten, Cloud- oder Hybridknoten, ein vollständig weitergeleitetes Netzwerk bilden müssen. Dies bedeutet, dass alle Knoten in der Lage sein müssen, sich gegenseitig auf Layer 3 über die IP-Adresse zu erreichen.

Die EKS-Steuerebene und die Cloud-Knoten sind bereits voneinander erreichbar, da sie sich in einem flachen Netzwerk (dem VPC) befinden. Die Hybridknoten befinden sich jedoch in einer anderen Netzwerk-Domain. Aus diesem Grund müssen Sie zusätzliches Routing in der VPC und in Ihrem On-Premises-Netzwerk konfigurieren, um den Datenverkehr zwischen den Hybridknoten und dem Rest des Clusters weiterzuleiten. Wenn die Hybridknoten untereinander und von der VPC aus erreichbar sind, können sich Ihre Hybridknoten in einem einzigen flachen Netzwerk oder in mehreren segmentierten Netzwerken befinden.

Routbarer Remote-Pod CIDRs

Damit die EKS-Steuerebene mit Pods kommunizieren kann, die in Hybridknoten ausgeführt werden (z. B. Webhooks oder der Metriken-Server), oder damit Pods, die auf Cloud-Knoten ausgeführt

werden, mit Pods kommunizieren können, die in Hybridknoten ausgeführt werden (Ost-West-Kommunikation der Workloads), muss Ihr Fern-Pod-CIDR vom VPC aus routingfähig sein. Das bedeutet, dass die VPC in der Lage sein muss, den Datenverkehr zum Pod CIDRs über das Gateway zu Ihrem lokalen Netzwerk weiterzuleiten, und dass Ihr lokales Netzwerk in der Lage sein muss, den Datenverkehr für einen Pod an den richtigen Knoten weiterzuleiten.

Es ist wichtig, den Unterschied zwischen den Pod-Routing-Anforderungen im VPC und On-Premises zu beachten. Die VPC muss lediglich wissen, dass der gesamte Datenverkehr, der an einen Fern-Pod gesendet wird, über das Gateway geleitet werden soll. Wenn Sie nur über eine Fern-Pod-CIDR verfügen, benötigen Sie nur eine Route.

Diese Anforderung gilt für alle Hops in Ihrem On-Premises-Netzwerk bis zum lokalen Router im selben Subnetz wie Ihre Hybridknoten. Dies ist der einzige Router, der den jedem Knoten zugewiesenen Pod-CIDR-Segment kennen muss, um sicherzustellen, dass der Datenverkehr für einen bestimmten Pod an den Knoten übermittelt wird, für den der Pod geplant wurde.

Sie können sich dafür entscheiden, diese Routen für den lokalen Pod CIDRs von Ihrem lokalen Router an die VPC-Routentabellen weiterzugeben, dies ist jedoch nicht erforderlich. Wenn sich Ihr On-Premises-Pod häufig CIDRs ändert und Ihre VPC-Routing-Tabellen aktualisiert werden müssen, um den sich ändernden Pod widerzuspiegeln CIDRs, empfehlen wir, den lokalen Pod an die VPC-Routing-Tabellen CIDRs weiterzugeben. Dies ist jedoch ungewöhnlich.

Beachten Sie, dass die Einschränkung, Ihren lokalen Pod routingfähig zu machen, optional ist. CIDRs Wenn Sie keine Webhooks auf Ihren Hybridknoten ausführen oder Pods auf Cloud-Knoten nicht mit Pods auf Hybridknoten kommunizieren lassen müssen, müssen Sie das Routing für den Pod in CIDRs Ihrem lokalen Netzwerk nicht konfigurieren.

Warum muss der lokale Pod mit CIDRs Hybridknoten routingfähig sein?

Wenn Sie EKS mit dem VPC-CNI für Ihre Cloud-Knoten verwenden, weist das VPC-CNI den Pods IPs direkt von der VPC zu. Das bedeutet, dass kein spezielles Routing erforderlich ist, da sowohl Cloud-Pods als auch die EKS-Steuerebene den Pod direkt erreichen können. IPs

Bei der Ausführung vor Ort (und mit anderen CNIs in der Cloud) werden die Pods normalerweise in einem isolierten Overlay-Netzwerk ausgeführt, und das CNI kümmert sich um die Übertragung des Datenverkehrs zwischen den Pods. Dies erfolgt üblicherweise durch Kapselung: Das CNI wandelt den Datenverkehr in pod-to-pod Datenverkehr um und kümmert sich dabei um die Kapselung und Entkapselung an beiden Enden. node-to-node Auf diese Weise sind keine zusätzlichen Konfigurationen an den Knoten und Routern erforderlich.

Die Vernetzung mit Hybridknoten ist einzigartig, da sie eine Kombination beider Topologien darstellt – die EKS-Steuerebene und die Cloud-Knoten (mit dem VPC CNI) erwarten ein flaches Netzwerk einschließlich Knoten und Pods, während die in Hybridknoten ausgeführten Pods sich in einem Überlagerungsnetzwerk befinden, das VXLAN für die Kapselung verwendet (standardmäßig in Cilium). In Hybridknoten ausgeführte Pods können die EKS-Steuerebene erreichen, und auf Cloudknoten ausgeführte Pods können dies erreichen, vorausgesetzt, das On-Premises-Netzwerk kann zum VPC weiterleiten. Ohne Routing für den Pod CIDRs im lokalen Netzwerk wird jedoch jeglicher Datenverkehr, der zu einer lokalen Pod-IP zurückkehrt, irgendwann verworfen, wenn das Netzwerk nicht weiß, wie es das Overlay-Netzwerk erreichen und zu den richtigen Knoten weiterleiten kann.

Kubernetes-Konzepte für Hybridknoten

Auf dieser Seite werden die wichtigsten Kubernetes-Konzepte erläutert, die der Systemarchitektur von EKS-Hybridknoten zugrunde liegen.

EKS-Steuerebene in der VPC

Die IPs der ENIs der EKS-Steuerebene werden im `kubernetes` `Endpoints`-Objekt im `default`-Namespace gespeichert. Wenn EKS neue ENIs erstellt oder ältere entfernt, aktualisiert EKS dieses Objekt, sodass die Liste der IPs immer auf dem neuesten Stand ist.

Sie können diese Endpunkte über den `kubernetes`-Service auch im `default`-Namespace verwenden. Diesem Service vom Typ `ClusterIP` wird immer die erste IP des Service-CIDR des Clusters zugewiesen. Beispielsweise lautet die Service-IP für den Service CIDR `172.16.0.0/16` `172.16.0.1`.

Im Allgemeinen greifen Pods (unabhängig davon, ob sie in der Cloud oder in Hybridknoten ausgeführt werden) auf diese Weise auf den EKS-Kubernetes-API-Server zu Pods verwenden die Service-IP als Ziel-IP, die in die tatsächlichen IPs einer der EKS-Steuerebenen-ENIs übersetzt wird. Die wichtigste Ausnahme ist `kube-proxy`, da es die Übersetzung einrichtet.

EKS-API-Server-Endpunkt

Die `kubernetes`-Service-IP ist nicht die einzige Möglichkeit, auf den EKS-API-Server zuzugreifen. EKS erstellt außerdem einen Route53-DNS-Namen, wenn Sie Ihren Cluster erstellen. Dies ist das Feld `endpoint` Ihres EKS-Clusters beim Aufruf der EKS-`DescribeCluster`-API-Aktion.

```
{
```

```
    "cluster": {
        "endpoint": "https://xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx.gr7.us-
west-2.eks.amazonaws.com",
        "name": "my-cluster",
        "status": "ACTIVE"
    }
}
```

In einem Cluster mit öffentlichem Endpunktzugriff oder öffentlichem und privatem Endpunktzugriff lösen Ihre Hybridknoten diesen DNS-Namen standardmäßig in eine öffentliche IP-Adresse auf, die über das Internet routingfähig ist. In einem Cluster mit privatem Endpunktzugriff wird der DNS-Name in die privaten IP-Adressen der EKS-Control-Plane-ENIs aufgelöst.

Auf diese Weise greifen `kubelet` und `kube-proxy` auf den Kubernetes-API-Server zu. Wenn Sie möchten, dass der gesamte Datenverkehr Ihres Kubernetes-Clusters über die VPC läuft, müssen Sie entweder Ihren Cluster im privaten Zugriffsmodus konfigurieren oder Ihren On-Premises-DNS-Server so ändern, dass der EKS-Cluster-Endpunkt zu den privaten IP-Adressen der EKS-Steuerebenen-ENIs aufgelöst wird.

`kubelet`-Endpunkt

Das `kubelet` stellt mehrere REST-Endpunkte bereit, sodass andere Teile des Systems mit jedem Knoten interagieren und Informationen von diesem erfassen können. In den meisten Clustern stammt der Großteil des Datenverkehrs zum `kubelet`-Server von der Steuerebene, aber auch bestimmte Überwachungsagenten können mit ihr interagieren.

Über diese Schnittstelle verarbeitet `kubelet` verschiedene Anfragen: das Abrufen von Protokollen (`kubectl logs`), das Ausführen von Befehlen in Containern (`kubectl exec`) und die Portweiterleitung des Datenverkehrs (`kubectl port-forward`). Jede dieser Anfragen interagiert über das `kubelet` mit der zugrunde liegenden Container-Laufzeitumgebung, was für Cluster-Administratoren und Entwickler nahtlos erscheint.

Der häufigste Nutzer dieser API ist der Kubernetes-API-Server. Wenn Sie einen der zuvor erwähnten `kubectl`-Befehle verwenden, sendet `kubectl` eine API-Anfrage an den API-Server, der dann die `kubelet`-API des Knotens aufruft, auf dem der Pod ausgeführt wird. Dies ist der Hauptgrund, warum die Knoten-IP von der EKS-Steuerebene aus erreichbar sein muss und warum Sie selbst bei ausgeführten Pods nicht auf deren Protokolle oder `exec` zugreifen können, wenn die Knoten-Route falsch konfiguriert ist.

Knoten-IPs

Wenn die EKS-Steuerebene mit einem Knoten kommuniziert, verwendet sie eine der im Node-Objektstatus (`status.addresses`) gemeldeten Adressen.

Bei EKS-Cloud-Knoten ist es üblich, dass der Kubelet die private IP-Adresse der EC2-Instance während der Knotenregistrierung als `InternalIP` meldet. Diese IP-Adresse wird dann vom Cloud Controller Manager (CCM) überprüft, um sicherzustellen, dass sie zur EC2-Instance gehört. Darüber hinaus fügt der CCM in der Regel die öffentlichen IP-Adressen (als `ExternalIP`) und DNS-Namen (`InternalDNS` und `ExternalDNS`) der Instance zum Knotenstatus hinzu.

Für Hybridknoten gibt es jedoch kein CCM. Wenn Sie einen Hybridknoten mit der EKS-Hybridknoten-CLI (`nodeadm`) registrieren, wird das kubelet so konfiguriert, dass es die IP-Adresse Ihres Rechners direkt im Status des Knotens meldet, ohne das CCM.

```
apiVersion: v1
kind: Node
metadata:
  name: my-node-1
spec:
  providerID: eks-hybrid:///us-west-2/my-cluster/my-node-1
status:
  addresses:
  - address: 10.1.1.236
    type: InternalIP
  - address: my-node-1
    type: Hostname
```

Wenn Ihr Rechner über mehrere IP-Adressen verfügt, wählt das kubelet nach seiner eigenen Logik eine davon aus. Sie können die ausgewählte IP mit dem `--node-ip`-Flag steuern, das Sie in der `nodeadm`-Konfiguration in `spec.kubelet.flags` übergeben können. Nur die im Node-Objekt gemeldete IP benötigt eine Route von der VPC. Ihre Rechner können über andere IP-Adressen verfügen, die von der Cloud aus nicht erreichbar sind.

kube-proxy

`kube-proxy` ist für die Implementierung der Service-Abstraktion auf der Netzwerkebene jedes Knotens verantwortlich. Es agiert als Netzwerk-Proxy und Load Balancer für den Datenverkehr, der für Kubernetes-Services bestimmt ist. Durch die kontinuierliche Überwachung des Kubernetes-API-Servers auf Änderungen in Bezug auf Services und Endpunkte aktualisiert `kube-proxy` dynamisch die Netzwerkregeln des zugrunde liegenden Hosts, um sicherzustellen, dass der Datenverkehr richtig geleitet wird.

Im iptables-Modus programmiert kube-proxy mehrere netfilter-Ketten zur Handhabung des Service-Datenverkehrs. Die Regeln bilden die folgende Hierarchie:

1. KUBE-SERVICES-Kette: Der Einstiegspunkt für den gesamten Service-Datenverkehr. Sie verfügt über Regeln, die mit dem ClusterIP und dem Port jedes Services übereinstimmen.
2. KUBE-SVC-XXX-Ketten: Service-spezifische Ketten verfügen über Load-Balancing-Regeln für jeden einzelnen Service.
3. KUBE-SEP-XXX-Ketten: Endpunktspezifische Ketten enthalten die aktuellen DNAT-Regeln.

Betrachten wir, was für einen Service-test-server im default-Namespace geschieht: * Service ClusterIP: 172.16.31.14 * Service-Port: 80 * Unterstützende Pods: 10.2.0.110, 10.2.1.39 und 10.2.2.254

Bei Überprüfung der iptables-Regeln (mit iptables-save 0 grep -A10 KUBE-SERVICES):

1. In der KUBE-SERVICES-Kette finden wir eine Regel, die zum Service passt:

```
-A KUBE-SERVICES -d 172.16.31.14/32 -p tcp -m comment --comment "default/test-server
 cluster IP" -m tcp --dport 80 -j KUBE-SVC-XYZABC123456
```

   - Diese Regel gilt für Pakete, die für 172.16.31.14:80 bestimmt sind.
   - Der Kommentar gibt an, wozu diese Regel dient: default/test-server cluster IP
   - Übereinstimmende Pakete werden zur KUBE-SVC-XYZABC123456-Kette weitergeleitet
2. Die KUBE-SVC-XYZABC123456-Kette verfügt über wahrscheinlichkeitsbasierte Lastausgleichsregeln:

```
-A KUBE-SVC-XYZABC123456 -m statistic --mode random --probability 0.33333333349 -j
 KUBE-SEP-POD1XYZABC
-A KUBE-SVC-XYZABC123456 -m statistic --mode random --probability 0.50000000000 -j
 KUBE-SEP-POD2XYZABC
-A KUBE-SVC-XYZABC123456 -j KUBE-SEP-POD3XYZABC
```

   - Erste Regel: 33,3 % Wahrscheinlichkeit, dass zu KUBE-SEP-POD1XYZABC weitergeleitet wird
   - Zweite Regel: 50 % Wahrscheinlichkeit, dass der verbleibende Datenverkehr (33,3 % der Gesamtmenge) zu KUBE-SEP-POD2XYZABC weitergeleitet wird
   - Letzte Regel: Der gesamte verbleibende Datenverkehr (33,3 % der Gesamtmenge) wird zu KUBE-SEP-POD3XYZABC weitergeleitet

3. Die einzelnen KUBE-SEP-XXX-Ketten führen das DNAT (Destination NAT) durch:

```
-A KUBE-SEP-POD1XYZABC -p tcp -m tcp -j DNAT --to-destination 10.2.0.110:80
-A KUBE-SEP-POD2XYZABC -p tcp -m tcp -j DNAT --to-destination 10.2.1.39:80
-A KUBE-SEP-POD3XYZABC -p tcp -m tcp -j DNAT --to-destination 10.2.2.254:80
```

- Diese DNAT-Regeln schreiben die Ziel-IP und den Ziel-Port neu, um den Datenverkehr an bestimmte Pods weiterzuleiten.
- Jede Regel verarbeitet etwa 33,3 % des Datenverkehrs und sorgt für eine gleichmäßige Lastverteilung zwischen `10.2.0.110`, `10.2.1.39` und `10.2.2.254`.

Diese mehrstufige Kettenstruktur ermöglicht `kube-proxy` die effiziente Implementierung des Service-Load-Balancings und der Umleitung durch Paketmanipulation auf Kernelebene, ohne dass ein Proxy-Prozess im Datenpfad erforderlich ist.

Auswirkungen auf den Kubernetes-Betrieb

Ein defekter `kube-proxy` in einem Knoten verhindert, dass dieser Knoten den Service-Datenverkehr ordnungsgemäß weiterleitet. Dies führt zu Zeitüberschreitungen oder Verbindungsfehlern für Pods, die auf Cluster-Services angewiesen sind. Dies kann insbesondere bei der erstmaligen Registrierung eines Knotens zu Störungen führen. Das CNI muss mit dem Kubernetes-API-Server kommunizieren, um Informationen wie die Pod-CIDR des Knotens abzurufen, bevor es die Pod-Vernetzung konfigurieren kann. Dazu verwendet es die `kubernetes`-Service-IP. Wenn `kube-proxy` jedoch nicht gestartet werden konnte oder die richtigen `iptables`-Regeln nicht festgelegt wurden, werden die an die `kubernetes`-Service-IP gesendeten Anfragen nicht in die tatsächlichen IPs der ENIs der EKS-Steuerebene übersetzt. Infolgedessen gerät das CNI in eine Absturzsituation, und keiner der Pods kann ordnungsgemäß ausgeführt werden.

Es ist bekannt, dass Pods die `kubernetes`-Service-IP verwenden, um mit dem Kubernetes-API-Server zu kommunizieren, jedoch muss `kube-proxy` zunächst `iptables`-Regeln festlegen, damit dies funktioniert.

Wie kommuniziert `kube-proxy` mit dem API-Server?

`kube-proxy` muss so konfiguriert werden, dass es die tatsächlichen IPs des Kubernetes-API-Servers oder einen DNS-Namen verwendet, der auf diese aufgelöst wird. Im Fall von EKS konfiguriert EKS die Standard-`kube-proxy` so, dass es auf den Route53-DNS-Namen verweist, den EKS beim Erstellen des Clusters erstellt. Sie können diesen Wert in der `kube-proxy`-ConfigMap im `kube-system`-Namespace sehen. Der Inhalt dieser ConfigMap ist ein `kubeconfig`, das in den `kube-`

proxy-Pod eingefügt wird. Suchen Sie also nach dem clusters0.cluster.server-Feld. Dieser Wert entspricht dem endpoint-Feld Ihres EKS-Clusters (beim Aufruf der EKS-DescribeCluster-API).

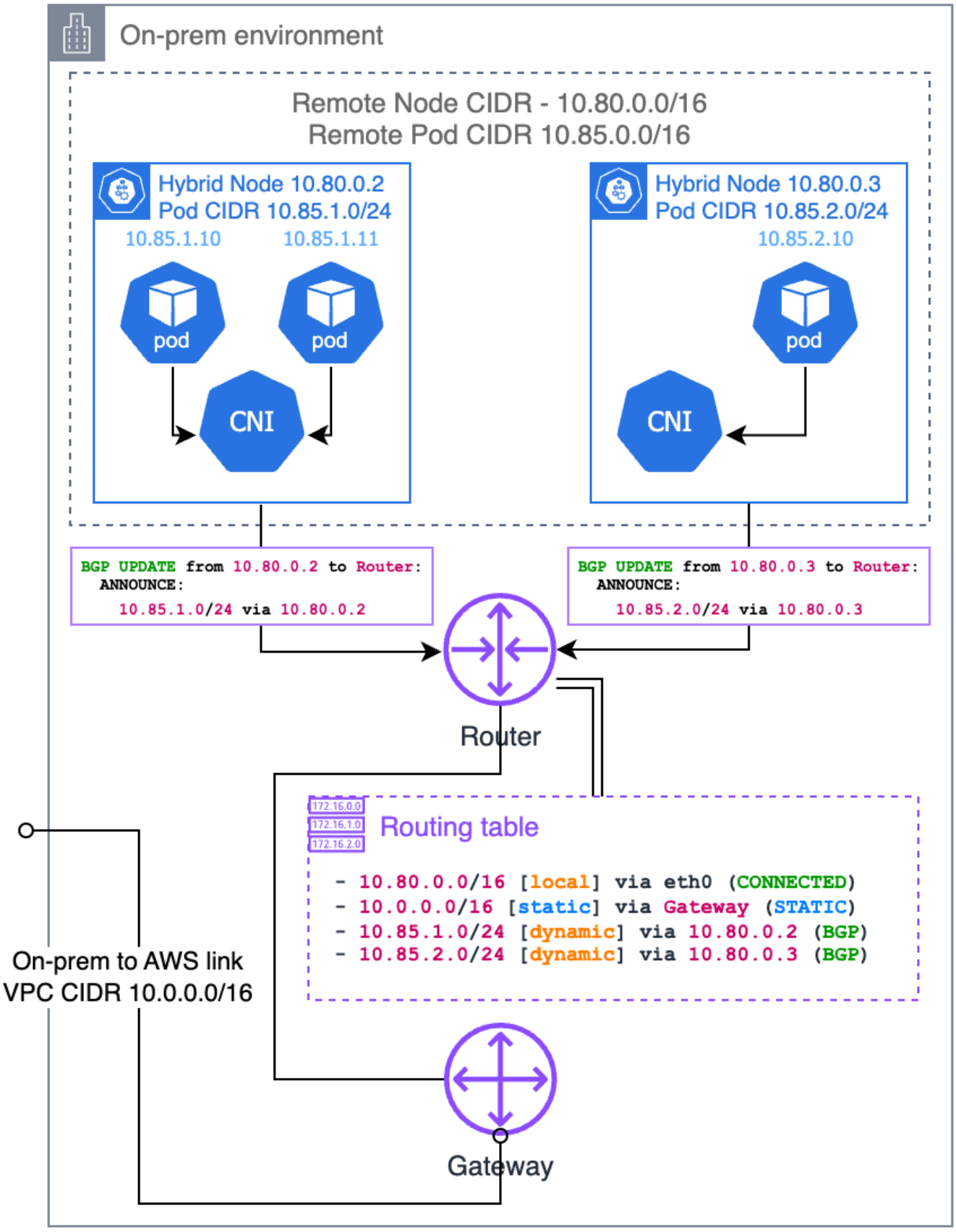
```
apiVersion: v1
data:
  kubeconfig: |-
    kind: Config
    apiVersion: v1
    clusters:
    - cluster:
        certificate-authority: /var/run/secrets/kubernetes.io/serviceaccount/ca.crt
        server: https://xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx.gr7.us-
west-2.eks.amazonaws.com
      name: default
    contexts:
    - context:
        cluster: default
        namespace: default
        user: default
      name: default
    current-context: default
    users:
    - name: default
      user:
        tokenFile: /var/run/secrets/kubernetes.io/serviceaccount/token
kind: ConfigMap
metadata:
  name: kube-proxy
  namespace: kube-system
```

Routingfähige Fern-Pod-CIDRs

Auf der Seite the section called “Netzwerkkonzepte” werden die Anforderungen zum Ausführen von Webhooks in Hybridknoten oder zum Kommunizieren von Pods in Cloud-Knoten mit Pods in Hybridknoten detailliert beschrieben. Die wichtigste Voraussetzung ist, dass der On-Premises-Router weiß, welcher Knoten für eine bestimmte Pod-IP zuständig ist. Es gibt mehrere Möglichkeiten, dies zu erreichen, darunter Border Gateway Protocol (BGP), statische Routen und Address Resolution Protocol (ARP)-Proxying. Diese werden in den folgenden Abschnitten behandelt.

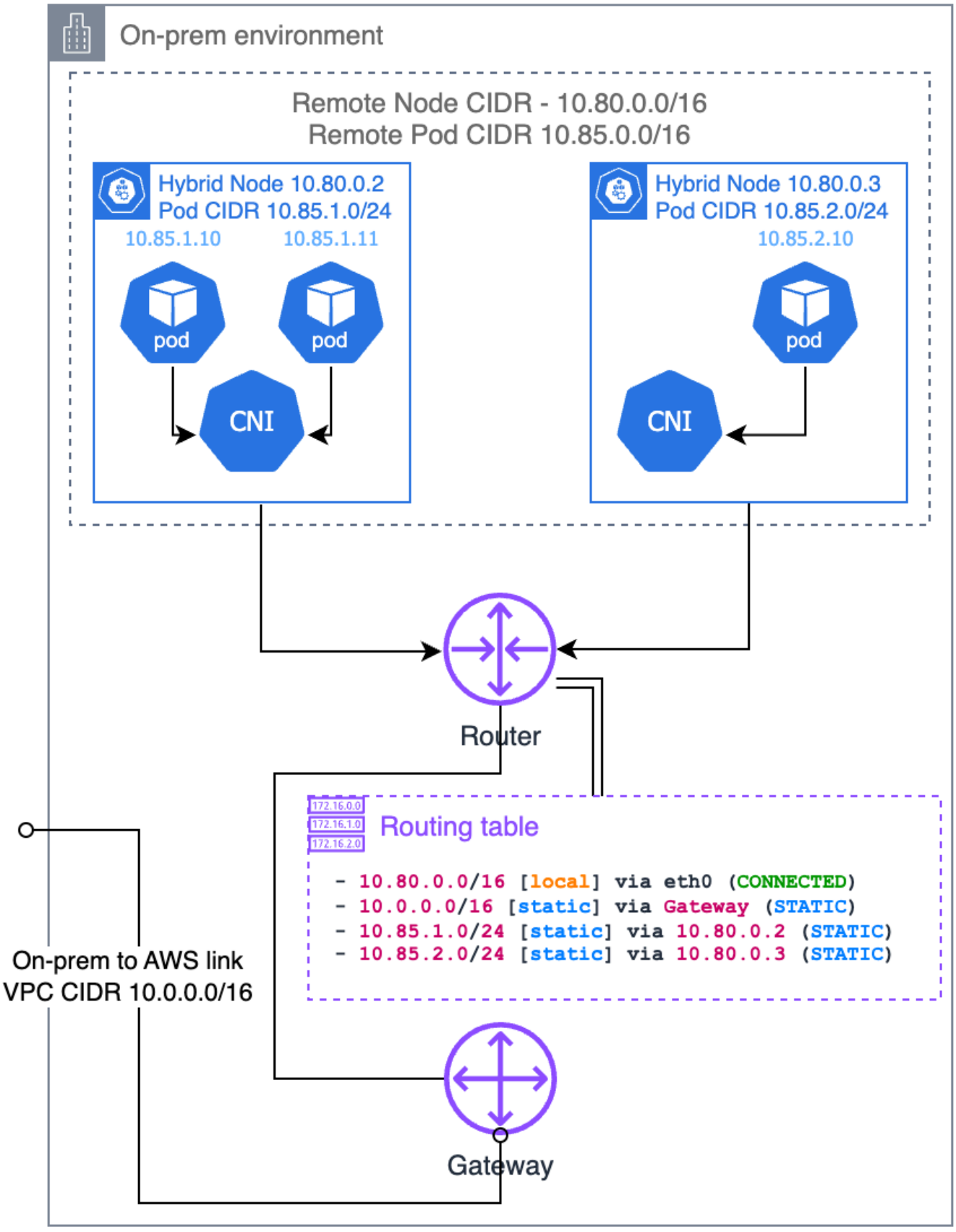
Border Gateway Protocol (BGP)

Wenn Ihr CNI dies unterstützt (z. B. Cilium und Calico), können Sie den BGP-Modus Ihres CNI verwenden, um Routen zu Ihren Pod-CIDRs pro Knoten von Ihren Knoten an Ihren lokalen Router weiterzuleiten. Bei Verwendung des BGP-Modus des CNI agiert Ihr CNI als virtueller Router, sodass Ihr lokaler Router davon ausgeht, dass die Pod-CIDR zu einem anderen Subnetz gehört und Ihr Knoten das Gateway zu diesem Subnetz ist.


On-prem environment
Remote Node CIDR - 10.80.0.0/16
Remote Pod CIDR 10.85.0.0/16
Hybrid Node 10.80.0.2
Pod CIDR 10.85.1.0/24
10.85.1.10
10.85.1.11
pod
pod
CNI
Hybrid Node 10.80.0.3
Pod CIDR 10.85.2.0/24
10.85.2.10
pod
CNI
BGP UPDATE from 10.80.0.2 to Router:
ANNOUNCE:
10.85.1.0/24 via 10.80.0.2
BGP UPDATE from 10.80.0.3 to Router:
ANNOUNCE:
10.85.2.0/24 via 10.80.0.3
Router
172.16.0.0
172.16.1.0
172.16.2.0
Routing table
- 10.80.0.0/16 [local] via eth0 (CONNECTED)
- 10.0.0.0/16 [static] via Gateway (STATIC)
- 10.85.1.0/24 [dynamic] via 10.80.0.2 (BGP)
- 10.85.2.0/24 [dynamic] via 10.80.0.3 (BGP)
On-prem to AWS link
VPC CIDR 10.0.0.0/16
Gateway

Statische Routen

Oder Sie können statische Routen in Ihrem lokalen Router konfigurieren. Dies ist die einfachste Methode, um die CIDR des On-Premises-Pods zu Ihrer VPC weiterzuleiten. Allerdings ist sie auch die fehleranfälligste und am schwierigsten zu wartende Methode. Sie müssen sicherstellen, dass die Routen stets mit den vorhandenen Knoten und den ihnen zugewiesenen Pod-CIDRs auf dem neuesten Stand sind. Wenn die Anzahl Ihrer Knoten gering und die Infrastruktur statisch ist, ist dies eine sinnvolle Option und macht die BGP-Unterstützung in Ihrem Router überflüssig. Wenn Sie sich dafür entscheiden, empfehlen wir Ihnen, Ihr CNI mit dem Pod-CIDR-Segment zu konfigurieren, das Sie jedem Knoten zuweisen möchten, anstatt dies Ihrem IPAM zu überlassen.


On-prem environment
Remote Node CIDR - 10.80.0.0/16
Remote Pod CIDR 10.85.0.0/16
Hybrid Node 10.80.0.2
Pod CIDR 10.85.1.0/24
10.85.1.10
10.85.1.11
pod
pod
CNI
Hybrid Node 10.80.0.3
Pod CIDR 10.85.2.0/24
10.85.2.10
pod
CNI
Router
Routing table
- 10.80.0.0/16 [local] via eth0 (CONNECTED)
- 10.0.0.0/16 [static] via Gateway (STATIC)
- 10.85.1.0/24 [static] via 10.80.0.2 (STATIC)
- 10.85.2.0/24 [static] via 10.80.0.3 (STATIC)
On-prem to AWS link
VPC CIDR 10.0.0.0/16
Gateway

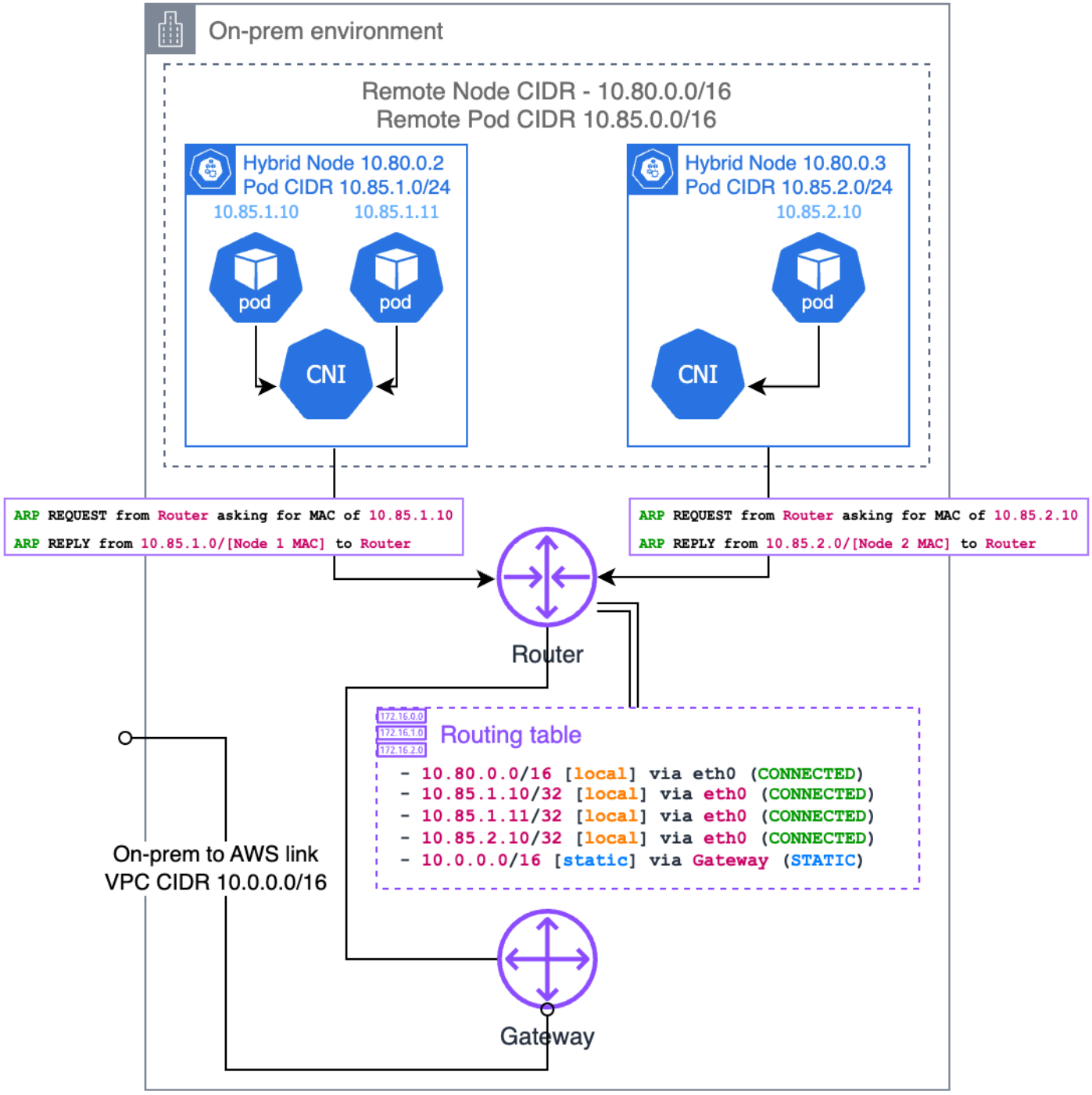
Address Resolution Protocol (ARP)-Proxying

ARP-Proxying ist ein weiterer Ansatz, um On-Premises-Pod-IPs routingfähig zu machen. Dies ist besonders nützlich, wenn sich Ihre Hybridknoten im selben Layer-2-Netzwerk wie Ihr lokaler Router befinden. Wenn ARP-Proxying aktiviert ist, antwortet ein Knoten auf ARP-Anfragen für Pod-IPs, die er hostet, auch wenn diese IPs zu einem anderen Subnetz gehören.

Wenn ein Gerät in Ihrem lokalen Netzwerk versucht, eine Pod-IP zu erreichen, sendet es zunächst eine ARP-Anfrage mit der Frage „Wer hat diese IP?“. Der Hybridknoten, der diesen Pod hostet, antwortet mit seiner eigenen MAC-Adresse und sagt: „Ich kann den Datenverkehr für diese IP verarbeiten.“ Dadurch wird eine direkte Verbindung zwischen den Geräten in Ihrem lokalen Netzwerk und den Pods hergestellt, ohne dass eine Router-Konfiguration erforderlich ist.

Damit dies funktioniert, muss Ihr CNI die Proxy-ARP-Funktionalität unterstützen. Cilium verfügt über integrierten Support für Proxy-ARP, den Sie über die Konfiguration aktivieren können. Der wichtigste Aspekt besteht darin, dass sich das Pod-CIDR nicht mit anderen Netzwerken in Ihrer Umgebung überschneiden darf, da dies zu Routing-Konflikten führen könnte.

Dieser Ansatz bietet mehrere Vorteile: * Sie müssen Ihren Router nicht mit BGP konfigurieren oder statische Routen verwalten. * Funktioniert gut in Umgebungen, in denen Sie keine Kontrolle über Ihre Router-Konfiguration haben.



Pod-zu-Pod-Kapselung

In On-Premises-Umgebungen verwenden CNIs in der Regel Kapselungsprotokolle, um Überlagerungsnetzwerke zu erstellen, die auf dem physischen Netzwerk betrieben werden können, ohne dass dieses neu konfiguriert werden muss. In diesem Abschnitt wird erläutert, wie diese

Kapselung funktioniert. Beachten Sie, dass einige Details je nach dem von Ihnen verwendeten CNI variieren können.

Bei der Kapselung werden ursprüngliche Pod-Netzwerkpakete in ein anderes Netzwerkpaket eingebunden, das durch das zugrunde liegende physische Netzwerk geleitet werden kann. Dadurch können Pods über Knoten hinweg kommunizieren, auf denen dasselbe CNI ausgeführt wird, ohne dass das physische Netzwerk wissen muss, wie diese Pod-CIDRs weitergeleitet werden sollen.

Das am häufigsten verwendete Kapselungsprotokoll mit Kubernetes ist Virtual Extensible LAN (VXLAN). Abhängig von Ihrem CNI sind jedoch auch andere Protokolle verfügbar (z. B. `Geneve`).

VXLAN-Kapselung

VXLAN kapselt Layer-2-Ethernet-Frames in UDP-Paketen. Wenn ein Pod Datenverkehr an einen anderen Pod auf einem anderen Knoten sendet, führt das CNI Folgendes aus:

1. Das CNI fängt Pakete von Pod A ab
2. Das CNI verpackt das ursprüngliche Paket in einen VXLAN-Header
3. Dieses verpackte Paket wird dann über den regulären Netzwerk-Stack des Knotens an den Zielknoten gesendet
4. Das CNI auf dem Zielknoten entpackt das Paket und übermittelt es an Pod B

Folgendes passiert mit der Paketstruktur während der VXLAN-Kapselung:

Ursprüngliches Pod-zu-Pod-Paket:

```
+-----------------+---------------+-------------+-----------------+
| Ethernet Header | IP Header     | TCP/UDP     | Payload         |
| Src: Pod A MAC  | Src: Pod A IP | Src Port    |                 |
| Dst: Pod B MAC  | Dst: Pod B IP | Dst Port    |                 |
+-----------------+---------------+-------------+-----------------+
```

Nach der VXLAN-Kapselung:

```
+-----------------+-------------+--------------+------------
+--------------------------+
| Outer Ethernet  | Outer IP    | Outer UDP    | VXLAN      | Original Pod-to-Pod
 |
| Src: Node A MAC | Src: Node A | Src: Random  | VNI: xx    | Packet (unchanged
 |
```

```
| Dst: Node B MAC | Dst: Node B | Dst: 4789    |            | from above)
 |
+-----------------+-------------+--------------+------------
+--------------------------+
```

Der VXLAN Network Identifier (VNI) unterscheidet zwischen verschiedenen Überlagerungsnetzwerken.

Szenarien für die Pod-Kommunikation

Pods auf demselben Hybridknoten

Wenn Pods auf demselben Hybridknoten kommunizieren, ist normalerweise keine Kapselung erforderlich. Das CNI richtet lokale Routen ein, die den Datenverkehr zwischen Pods über die internen virtuellen Schnittstellen des Knotens leiten:

```
Pod A -> veth0 -> node's bridge/routing table -> veth1 -> Pod B
```

Das Paket verlässt den Knoten nie und erfordert keine Kapselung.

Pods auf verschiedenen Hybridknoten

Die Kommunikation zwischen Pods auf verschiedenen Hybridknoten erfordert eine Kapselung:

```
Pod A -> CNI -> [VXLAN encapsulation] -> Node A network -> router or gateway -> Node B
 network -> [VXLAN decapsulation] -> CNI -> Pod B
```

Dadurch kann der Pod-Datenverkehr die physische Netzwerkinfrastruktur durchlaufen, ohne dass das physische Netzwerk das Pod-IP-Routing verstehen muss.

Abläufe des Netzwerk-Datenverkehrs für Hybridknoten

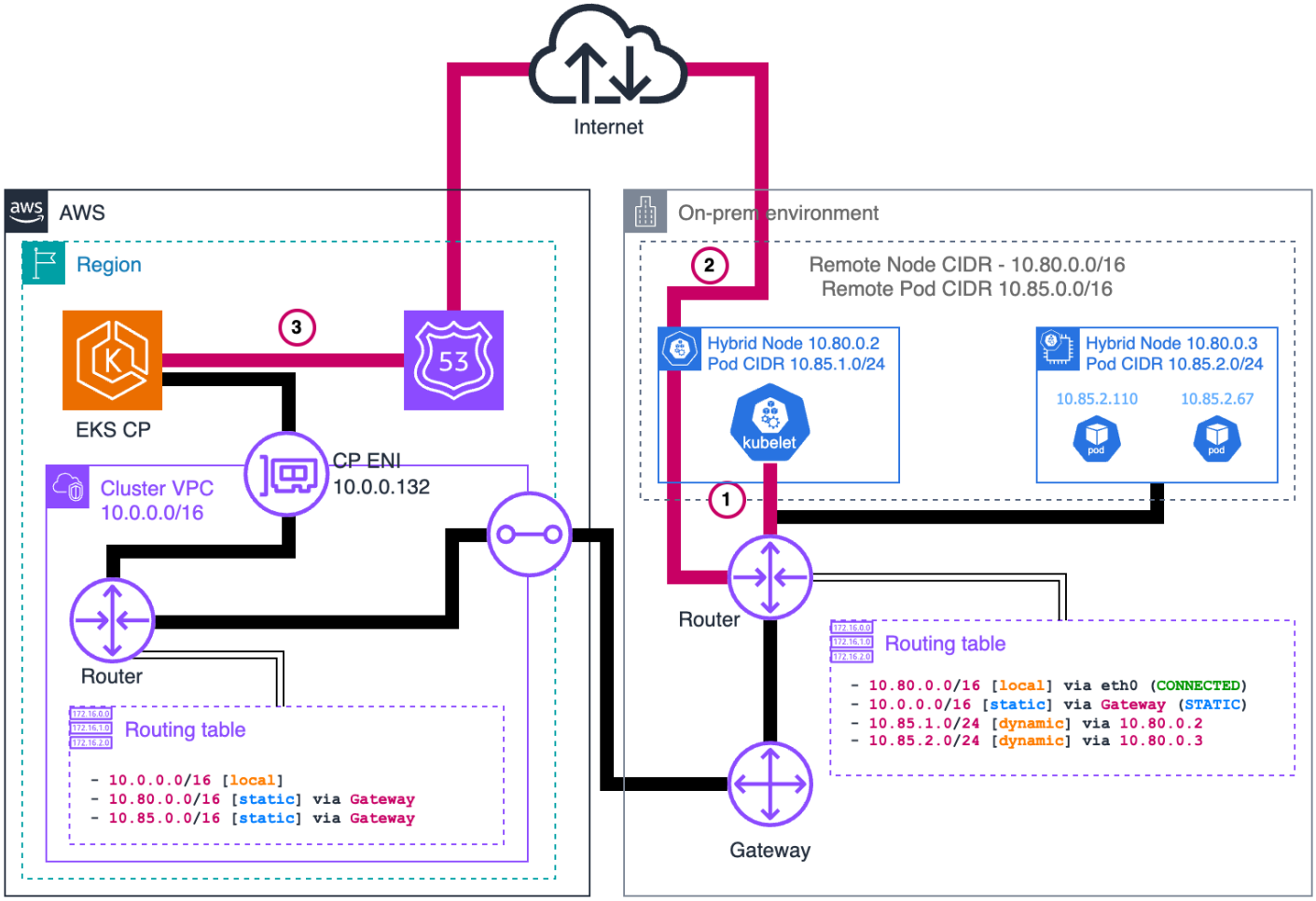
Auf dieser Seite werden die Netzwerkverkehrsflüsse für EKS-Hybridknoten mit Diagrammen beschrieben, die die end-to-end Netzwerkpfade für die verschiedenen Verkehrstypen zeigen.

Die folgenden Datenverkehr-Abläufe werden abgedeckt:

- the section called “Hybridknoten `kubelet` zur EKS-Steuerebene”
- the section called “EKS-Steuerebene zum Hybridknoten (`kubelet`-Server)”

- the section called “Pods, die in Hybridknoten zur EKS-Steuerebene ausgeführt werden”
- the section called “EKS-Steuerebene für Pods, die in einem Hybridknoten ausgeführt werden (Webhooks)”
- the section called “Pod-to-Pod läuft auf Hybridknoten”
- the section called “Pods auf Cloud-Knoten zu Pods auf Hybridknoten (Ost-West–Datenverkehr)”

Hybridknoten **`kubelet`** zur EKS-Steuerebene



Anforderung

1**`kubelet`**. Initiiert Anforderung

Wenn `kubelet` auf einem Hybridknoten mit der EKS-Steuerebene kommunizieren muss (z. B. um den Knotenstatus zu melden oder Pod-Spezifikationen abzurufen), verwendet es die während der Knotenregistrierung bereitgestellte `kubeconfig`-Datei. Dieses `kubeconfig` hat die URL des API-Server-Endpunkts (den Route53-DNS-Namen) und keine direkten IP-Adressen.

Das `kubelet` führt eine DNS-Suche für den Endpunkt durch (z. B. `https://xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx.gr7.us-west-2.eks.amazonaws.com`). In einem Cluster mit öffentlichem Zugriff wird daraus eine öffentliche IP-Adresse (z. B. `54.239.118.52`) abgeleitet, die zum in AWS ausgeführten EKS-Service gehört. `kubelet` erstellt dann eine sichere HTTPS-Anfrage an diesen Endpunkt. Das ursprüngliche Paket sieht folgendermaßen aus:

```
+--------------------+---------------------+-----------------+
| IP Header          | TCP Header          | Payload         |
| Src: 10.80.0.2     | Src: 52390 (random) |                 |
| Dst: 54.239.118.52 | Dst: 443            |                 |
+--------------------+---------------------+-----------------+
```

2. Lokales Router-Routing

Da es sich bei der Ziel-IP um eine öffentliche IP-Adresse handelt und sie nicht Teil des lokalen Netzwerks ist, sendet `kubelet` dieses Paket an sein Standard-Gateway (den On-Premises-Router vor Ort). Der Router prüft die Ziel-IP und stellt fest, dass es sich um eine öffentliche IP-Adresse handelt.

Bei öffentlichem Datenverkehr leitet der Router das Paket normalerweise an ein Internet-Gateway oder einen Border-Router weiter, der den ausgehenden Datenverkehr ins Internet abwickelt. Dies ist im Diagramm nicht dargestellt und hängt von der Konfiguration Ihres On-Premises-Netzwerks ab. Das Paket durchläuft Ihre On-Premises-Netzwerkinfrastruktur und erreicht schließlich das Netzwerk Ihres Internet-Serviceanbieters.

3. Übermittlung an die EKS-Steuerebene

Das Paket wird über das öffentliche Internet und die Transitnetzwerke übertragen, bis es AWS das Netzwerk erreicht. AWS Das Netzwerk leitet das Paket an den EKS-Dienstendpunkt in der entsprechenden Region weiter. Wenn das Paket den EKS-Service erreicht, wird es an die eigentliche EKS-Steuerebene für Ihren Cluster weitergeleitet.

Dieses Routing über das öffentliche Internet unterscheidet sich vom privaten VPC-weitergeleiteten Pfad, den wir in anderen Datenverkehrsabläufen sehen werden. Der Hauptunterschied besteht darin, dass im öffentlichen Zugriffsmodus der Datenverkehr von On-Premises-`kubelet` (jedoch nicht von Pods) zur EKS-Steuerebene nicht über Ihre VPC läuft, sondern stattdessen die globale Internetinfrastruktur verwendet.

Antwort

Nachdem die EKS-Steuerebene die `kubelet`-Anfrage verarbeitet hat, sendet sie eine Antwort zurück:

3. EKS-Steuerebene sendet Antwort

Die EKS-Steuerebene erstellt ein Antwortpaket. Dieses Paket hat die öffentliche IP als Quelle und die IP des Hybridknotens als Ziel:

```
+-------------------+--------------------+-----------------+
| IP Header         | TCP Header         | Payload         |
| Src: 54.239.118.52 | Src: 443           |                 |
| Dst: 10.80.0.2    | Dst: 52390         |                 |
+-------------------+--------------------+-----------------+
```

2. Internet-Routing

Das Antwortpaket wird über das Internet zurückgesendet und folgt dabei dem von den Internet-Serviceanbietern festgelegten Routing-Pfad, bis es Ihren On-Premises-Netzwerk-Edge-Router erreicht.

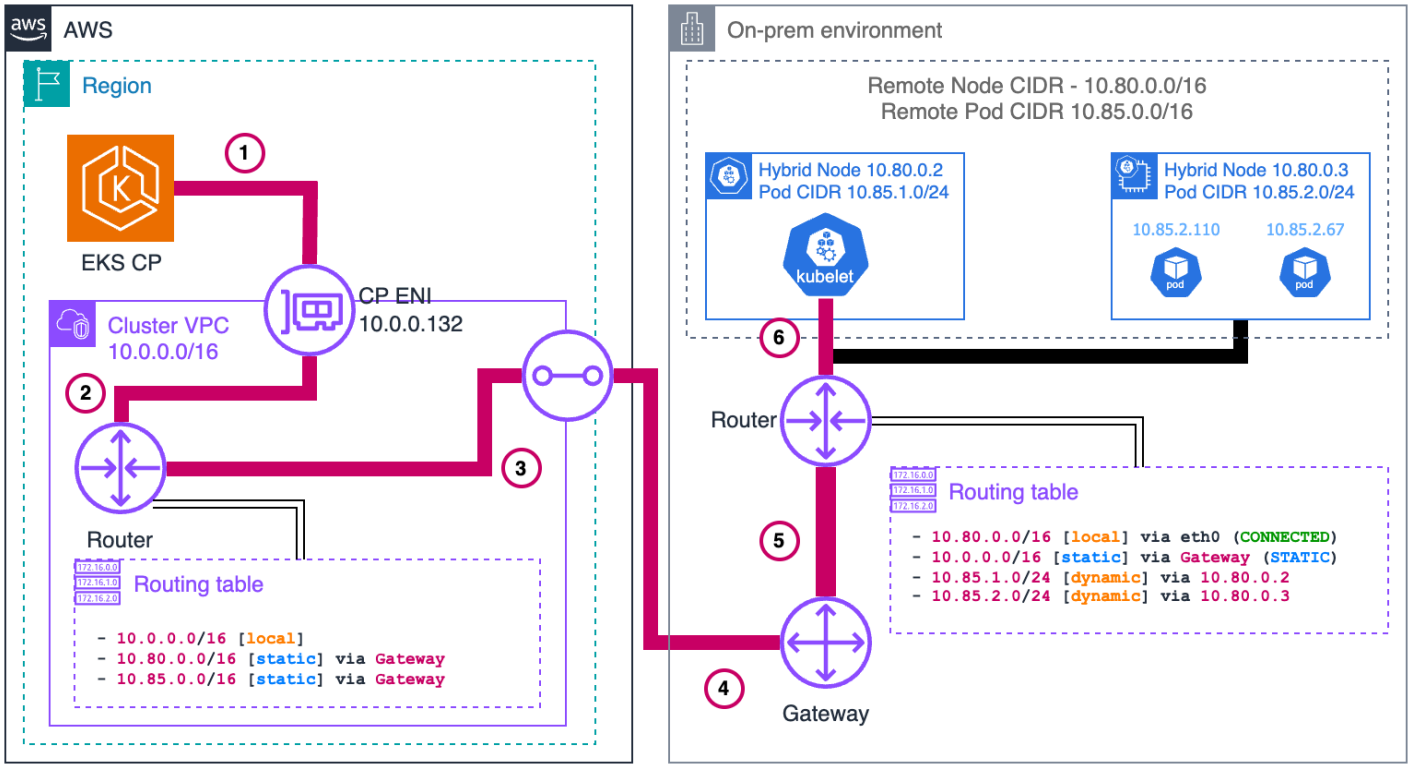
1. Lokale Übermittlung

Ihr On-Premises-Router empfängt das Paket und erkennt die Ziel-IP (`10.80.0.2`) als zu Ihrem lokalen Netzwerk gehörig. Er leitet das Paket über Ihre lokale Netzwerkinfrastruktur weiter, bis es den Ziel-Hybridknoten erreicht, wo es von `kubelet` empfangen und verarbeitet wird.

Hybridknoten **`kube-proxy`** zur EKS-Steuerebene

Wenn Sie den öffentlichen Endpunktzugriff für den Cluster aktivieren, wird für den Rück-Datenverkehr das öffentliche Internet verwendet. Dieser Datenverkehr stammt vom `kube-proxy` auf dem Hybridknoten zur EKS-Steuerebene und folgt dem gleichen Pfad wie der Datenverkehr vom `kubelet` zur EKS-Steuerebene.

EKS-Steuerebene zum Hybridknoten (**`kubelet`**-Server)



Anforderung

1. EKS-Kubernetes-API-Server initiiert Anfrage

Der EKS-Kubernetes-API-Server ruft die IP-Adresse (`10.80.0.2`) des Knotens aus dem Status des Knotenobjekts ab. Anschließend leitet er diese Anfrage über seine ENI in der VPC weiter, da die Ziel-IP zum konfigurierten CIDR (`10.80.0.0/16`) des Knotens aus der Ferne gehört. Das ursprüngliche Paket sieht folgendermaßen aus:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.0.0.132 | Src: 67493 (random) |                 |
| Dst: 10.80.0.2  | Dst: 10250          |                 |
+-----------------+---------------------+-----------------+
```

2. VPC-Netzwerkverarbeitung

Das Paket verlässt die ENI und gelangt in die VPC- Netzwerkebene, wo es zur weiteren Weiterleitung an das Gateway des Subnetzes weitergeleitet wird.

3. Abfrage der VPC-Routing-Tabelle

Die VPC-Routing-Tabelle für das Subnetz, das die EKS-Steuerebene ENI enthält, verfügt über eine bestimmte Route (die zweite im Diagramm) für den CIDR des Knotens aus der Ferne. Basierend auf dieser Routing-Regel wird das Paket an das VPC-to-onprem Gateway weitergeleitet.

4. Grenzüberschreitender Transit

Das Gateway überträgt das Paket über die Cloud-Grenze hinweg über Ihre vorhandene Verbindung (z. B. Direct Connect oder VPN) an Ihr On-Premises-Netzwerk.

5. On-Premises-Netzwerkempfang

Das Paket wird an Ihren On-Premises-Router vor Ort weitergeleitet, der den Datenverkehr für das Subnetz verwaltet, in dem sich Ihre Hybridknoten befinden.

6. Endgültige Übermittlung

Der lokale Router erkennt, dass die Ziel IP (`10.80.0.2`)-Adresse zu seinem direkt verbundenen Netzwerk gehört, und leitet das Paket direkt an den Ziel-Hybridknoten weiter, wo `kubelet` die Anfrage empfängt und verarbeitet.

Antwort

Nachdem das `kubelet` des Hybridknotens die Anfrage verarbeitet hat, sendet er eine Antwort auf dem gleichen Weg in umgekehrter Richtung zurück:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.80.0.2  | Src: 10250          |                 |
| Dst: 10.0.0.132 | Dst: 67493          |                 |
+-----------------+---------------------+-----------------+
```

6. **`kubelet`** Sendet Antwort

Das `kubelet` auf dem Hybridknoten (`10.80.0.2`) erstellt ein Antwortpaket mit der ursprünglichen Quell-IP als Ziel. Das Ziel gehört nicht zum lokalen Netzwerk, daher wird es an das Standard-Gateway des Hosts gesendet, das der lokale Router ist.

5. Lokales Router-Routing

Der Router stellt fest, dass die Ziel-IP (`10.0.0.132`) zu `10.0.0.0/16` gehört, das über eine Route verfügt, die auf das Gateway zeigt, das mit AWS verbunden ist.

4. Grenzüberschreitende Rückgabe

Das Paket wird über dieselbe On-Premises-VPC-Verbindung (z. B. Direct Connect oder VPN) zurückgesendet und durchquert die Cloud-Grenze in umgekehrter Richtung.

3. VPC-Routing

Wenn das Paket in der VPC ankommt, erkennen die Routing-Tabellen, dass die Ziel-IP zu einer VPC-CIDR gehört. Das Paket wird innerhalb der VPC weitergeleitet.

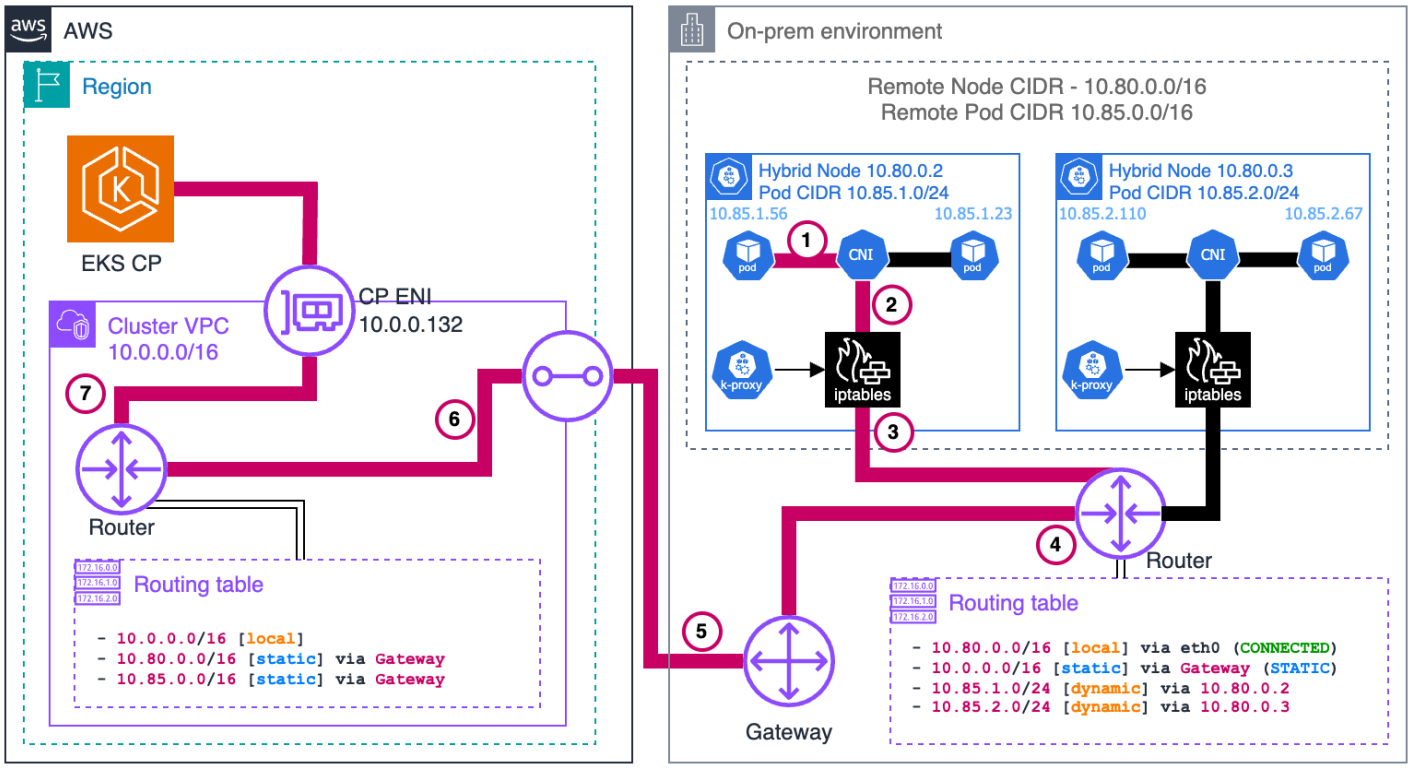
2. VPC-Netzwerkzustellung

Die VPC-Netzwerkschicht leitet das Paket an das Subnetz mit der EKS-Steuerebene ENI (`10.0.0.132`) weiter.

1. ENI-Empfang

Das Paket erreicht die EKS-Steuerebene ENI, die an den Kubernetes-API-Server angefügt ist, und schließt damit den Rundlauf ab.

Pods, die in Hybridknoten zur EKS-Steuerebene ausgeführt werden



Ohne CNI NAT

Anforderung

Pods kommunizieren in der Regel über den `kubernetes`-Service mit dem Kubernetes-API-Server. Die Service-IP ist die erste IP-Adresse des CIDR-Adressbereichs des Clusters. Diese Konvention ermöglicht es Pods, die vor der Verfügbarkeit von CoreDNS ausgeführt werden müssen, den API-Server zu erreichen, beispielsweise das CNI. Anfragen verlassen den Pod mit der Service-IP als Ziel. Wenn beispielsweise die CIDR des Services `172.16.0.0/16` ist, lautet die IP-Adresse des Services `172.16.0.1`.

1. Pod initiiert Anfrage

Der Pod sendet von einem zufälligen Quell-Port eine Anfrage an die `kubernetes`-Service-IP (`172.16.0.1`) auf dem API-Server-Port (443). Das Paket sieht wie folgt aus:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.85.1.56 | Src: 67493 (random) |                 |
| Dst: 172.16.0.1 | Dst: 443            |                 |
+-----------------+---------------------+-----------------+
```

2. CNI-Verarbeitung

Das CNI erkennt, dass die Ziel-IP zu keinem von ihm verwalteten Pod-CIDR gehört. Da ausgehendes NAT deaktiviert ist, leitet das CNI das Paket unverändert an den Host-Netzwerk-Stack weiter.

3. Knoten-Netzwerkverarbeitung

Das Paket gelangt in den Netzwerk-Stack des Knotens, wo `netfilter`-Hooks die vom Kube-Proxy festgelegten `iptables`-Regeln auslösen. Es gelten mehrere Regeln in der folgenden Reihenfolge:

1. Das Paket erreicht zunächst die `KUBE-SERVICES`-Kette, die Regeln enthält, die mit der ClusterIP und dem Port jedes Services übereinstimmen.
2. Die übereinstimmende Regel springt zur `KUBE-SVC-XXX`-Kette für den `kubernetes`-Service (Pakete mit dem Ziel für `172.16.0.1:443`), die Regeln zum Load Balancing enthält.
3. Die Lastenausgleichsregel wählt zufällig eine der `KUBE-SEP-XXX` Ketten für die Steuerungsebene ENI IPs (`10.0.0.132`oder`10.0.1.23`) aus.

4. Die ausgewählte KUBE-SEP-XXX-Kette verfügt über die eigentliche Regel, welche die Ziel-IP von der Service-IP in die ausgewählte IP ändert. Dies wird als Destination Network Address Translation (DNAT) bezeichnet.

Nachdem diese Regeln angewendet wurden, sieht das Paket wie folgt aus, vorausgesetzt, dass die IP der ausgewählten EKS-Steuerebene ENI 10.0.0.132 ist:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.85.1.56 | Src: 67493 (random) |                 |
| Dst: 10.0.0.132 | Dst: 443            |                 |
+-----------------+---------------------+-----------------+
```

Der Knoten leitet das Paket an sein Standard-Gateway weiter, da sich die Ziel-IP nicht im lokalen Netzwerk befindet.

4. Lokales Router-Routing

Der lokale Router stellt fest, dass die Ziel-IP (10.0.0.132) zur VPC-CIDR (10.0.0.0/16) gehört, und leitet sie an das Gateway weiter, das mit AWS verbunden ist.

5. Grenzüberschreitender Transit

Das Paket wird über Ihre eingerichtete Verbindung (z. B. Direct Connect oder VPN) über die Cloud-Grenze hinweg an die VPC weitergeleitet.

6. VPC-Netzwerkzustellung

Die VPC-Netzwerkebene leitet das Paket an das richtige Subnetz weiter, in dem sich die EKS-Steuerebene ENI (10.0.0.132) befindet.

7. ENI-Empfang

Das Paket erreicht die EKS-Steuerebene ENI, die an den Kubernetes-API-Server angefügt ist.

Antwort

Nachdem die EKS-Steuerebene die Anfrage verarbeitet hat, sendet sie eine Antwort zurück an den Pod:

7. API-Server übermittelt Antwort

Der EKS-Kubernetes-API-Server erstellt ein Antwortpaket mit der ursprünglichen Quell-IP als Ziel. Das Paket sieht wie folgt aus:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.0.0.132 | Src: 443            |                 |
| Dst: 10.85.1.56 | Dst: 67493          |                 |
+-----------------+---------------------+-----------------+
```

Da die Ziel-IP zum konfigurierten Fern-Pod-CIDR (`10.85.0.0/16`) gehört, sendet sie diese über ihre ENI in der VPC mit dem Router des Subnetzes als nächstem Hop.

6. VPC-Routing

Die VPC-Routentabelle enthält einen Eintrag für den Remote-Pod CIDR (`10.85.0.0/16`), der diesen Datenverkehr an das Gateway weiterleitet. VPC-to-onprem

5. Grenzüberschreitender Transit

Das Gateway überträgt das Paket über die Cloud-Grenze hinweg über Ihre vorhandene Verbindung (z. B. Direct Connect oder VPN) an Ihr On-Premises-Netzwerk.

4. On-Premises-Netzwerkempfang

Das Paket erreicht Ihren lokalen On-Premises-Router.

3. Zustellung an den Knoten

Die Tabelle des Routers enthält einen Eintrag für `10.85.1.0/24` mit `10.80.0.2` als nächstem Hop, wodurch das Paket an unseren Knoten übermittelt wird.

2. Knoten-Netzwerkverarbeitung

Während das Paket vom Netzwerk-Stack des Knotens verarbeitet wird, ordnet `conntrack` (ein Teil von `netfilter`) das Paket der Verbindung zu, die der Pod ursprünglich hergestellt hat. Da ursprünglich DNAT angewendet wurde, kehrt `conntrack` das DNAT um, indem die Quell-IP von der IP der ENI der EKS-Steuerebene in die `kubernetes`-Service-IP umgeschrieben wird:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 172.16.0.1 | Src: 443            |                 |
| Dst: 10.85.1.56 | Dst: 67493          |                 |
```

```
+-----------------+---------------------+-----------------+
```

1. CNI-Verarbeitung

Das CNI erkennt, dass die Ziel-IP zu einem Pod in seinem Netzwerk gehört, und liefert das Paket an den richtigen Pod-Netzwerk-Namespace.

Dieser Ablauf zeigt, warum Remote Pod von der VPC bis zu dem spezifischen Knoten, der jeden Pod hostet, ordnungsgemäß routbar sein CIDRs muss. Der gesamte Rückpfad hängt vom ordnungsgemäßen Routing des Pods sowohl IPs in der Cloud als auch in lokalen Netzwerken ab.

Mit CNI NAT

Dieser Ablauf ist dem ohne CNI NAT sehr ähnlich, jedoch mit einem wesentlichen Unterschied: Das CNI wendet Source NAT (SNAT) auf das Paket an, bevor es an den Netzwerk-Stack des Knotens gesendet wird. Dadurch wird die Quell-IP des Pakets in die IP des Knotens geändert, sodass das Paket ohne zusätzliche Routing-Konfiguration zurück zum Knoten geleitet werden kann.

Anforderung

1. Pod initiiert Anfrage

Der Pod sendet von einem zufälligen Quell-Port eine Anfrage an die `kubernetes`-Service-IP (`172.16.0.1`) auf dem EKS-Kubernetes-API-Server-Port (443). Das Paket sieht wie folgt aus:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.85.1.56 | Src: 67493 (random) |                 |
| Dst: 172.16.0.1 | Dst: 443            |                 |
+-----------------+---------------------+-----------------+
```

2. CNI-Verarbeitung

Das CNI erkennt, dass die Ziel-IP zu keinem von ihm verwalteten Pod-CIDR gehört. Da ausgehendes NAT aktiviert ist, wendet das CNI SNAT auf das Paket an und ändert die Quell-IP in die IP des Knotens, bevor es an den Netzwerk-Stack des Knotens weitergeleitet wird:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.80.0.2  | Src: 67493 (random) |                 |
| Dst: 172.16.0.1 | Dst: 443            |                 |
```

```
+-----------------+---------------------+-----------------+
```

Hinweis: CNI und `iptables` werden im Beispiel aus Gründen der Übersichtlichkeit als separate Blöcke dargestellt. In der Praxis ist es jedoch möglich, dass einige NAT CNIs verwenden`iptables`.

3. Knoten-Netzwerkverarbeitung

Hier `kube-proxy` verhalten sich die von festgelegten `iptables` Regeln genauso wie im vorherigen Beispiel, indem sie den Lastenausgleich zwischen dem Paket und einer der EKS-Steuerungsebenen durchführen ENIs. Das Paket sieht nun so aus:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.80.0.2  | Src: 67493 (random) |                 |
| Dst: 10.0.0.132 | Dst: 443            |                 |
+-----------------+---------------------+-----------------+
```

Der Knoten leitet das Paket an sein Standard-Gateway weiter, da sich die Ziel-IP nicht im lokalen Netzwerk befindet.

4. Lokales Router-Routing

Der lokale Router stellt fest, dass die Ziel-IP (`10.0.0.132`) zur VPC-CIDR (`10.0.0.0/16`) gehört, und leitet sie an das Gateway weiter, das mit AWS verbunden ist.

5. Grenzüberschreitender Transit

Das Paket wird über Ihre eingerichtete Verbindung (z. B. Direct Connect oder VPN) über die Cloud-Grenze hinweg an die VPC weitergeleitet.

6. VPC-Netzwerkzustellung

Die VPC-Netzwerkebene leitet das Paket an das richtige Subnetz weiter, in dem sich die EKS-Steuerebene ENI (`10.0.0.132`) befindet.

7. ENI-Empfang

Das Paket erreicht die EKS-Steuerebene ENI, die an den Kubernetes-API-Server angefügt ist.

Antwort

Nachdem die EKS-Steuerebene die Anfrage verarbeitet hat, sendet sie eine Antwort zurück an den Pod:

7. API-Server übermittelt Antwort

Der EKS-Kubernetes-API-Server erstellt ein Antwortpaket mit der ursprünglichen Quell-IP als Ziel. Das Paket sieht wie folgt aus:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.0.0.132 | Src: 443            |                 |
| Dst: 10.80.0.2  | Dst: 67493          |                 |
+-----------------+---------------------+-----------------+
```

Da die Ziel-IP zum konfigurierten Fern-Knoten-CIDR (`10.80.0.0/16`) gehört, sendet sie diese über ihre ENI in der VPC mit dem Router des Subnetzes als nächstem Hop.

6. VPC-Routing

Die VPC-Routentabelle enthält einen Eintrag für den Remote-Knoten CIDR (`10.80.0.0/16`), der diesen Verkehr an das Gateway weiterleitet. VPC-to-onprem

5. Grenzüberschreitender Transit

Das Gateway überträgt das Paket über die Cloud-Grenze hinweg über Ihre vorhandene Verbindung (z. B. Direct Connect oder VPN) an Ihr On-Premises-Netzwerk.

4. On-Premises-Netzwerkempfang

Das Paket erreicht Ihren lokalen On-Premises-Router.

3. Zustellung an den Knoten

Der lokale Router erkennt, dass die Ziel-IP-Adresse (`10.80.0.2`) zu seinem direkt verbundenen Netzwerk gehört, und leitet das Paket direkt an den Ziel-Hybridknoten weiter.

2. Knoten-Netzwerkverarbeitung

Während das Paket vom Netzwerkstack des Knotens verarbeitet wird, gleicht `conntrack` (ein Teil von `netfilter`) das Paket mit der ursprünglich vom Pod hergestellten Verbindung ab. Da ursprünglich DNAT angewendet wurde, wird dies rückgängig gemacht, indem die Quell-IP von der IP der EKS-Steuerebene ENI in die `kubernetes`-Service-IP umgeschrieben wird:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
```

```
| Src: 172.16.0.1 | Src: 443            |                 |
| Dst: 10.80.0.2  | Dst: 67493          |                 |
+-----------------+---------------------+-----------------+
```

1. CNI-Verarbeitung

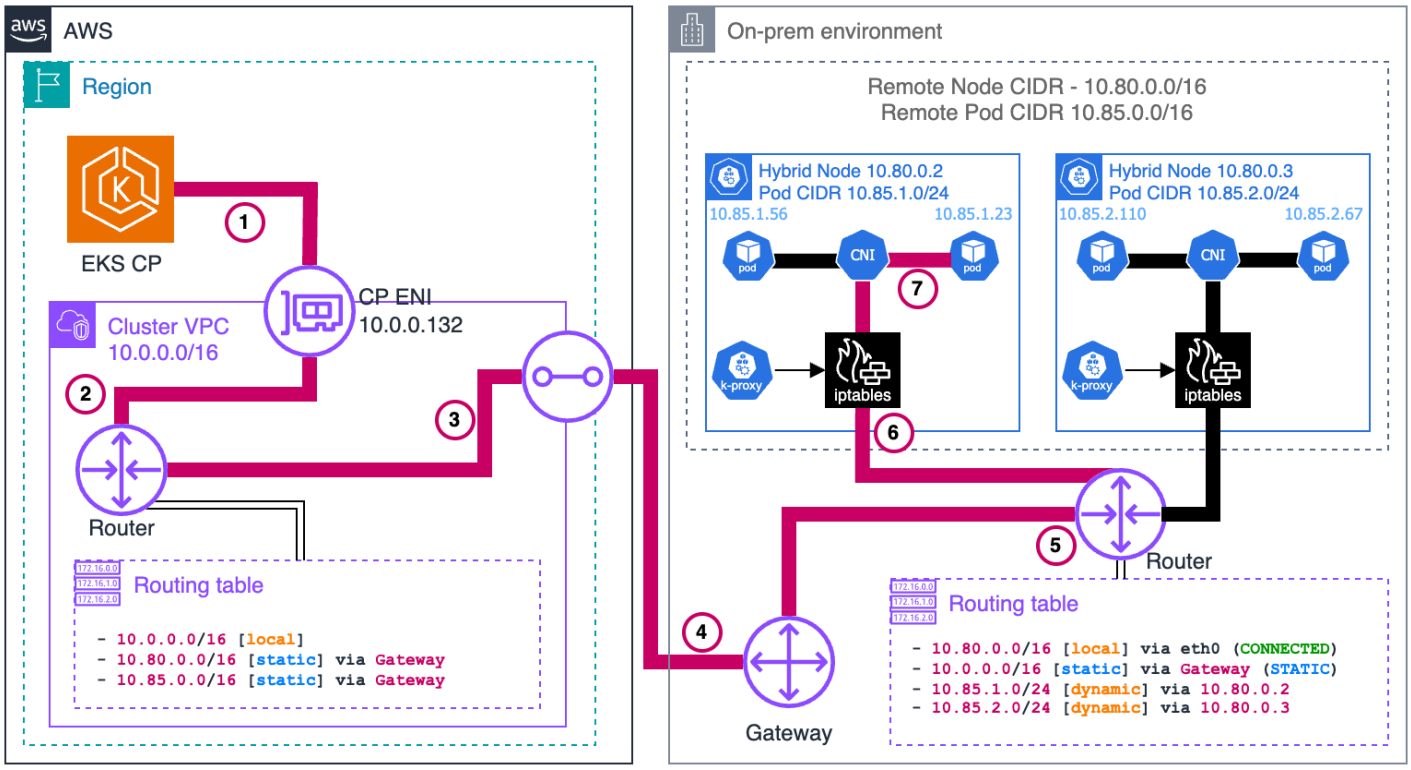
Das CNI identifiziert dieses Paket als Teil einer Verbindung, für die zuvor SNAT angewendet wurde. Es kehrt das SNAT um und ändert die Ziel-IP wieder in die IP des Pods:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 172.16.0.1 | Src: 443            |                 |
| Dst: 10.85.1.56 | Dst: 67493          |                 |
+-----------------+---------------------+-----------------+
```

Das CNI erkennt, dass die Ziel-IP zu einem Pod in seinem Netzwerk gehört, und liefert das Paket an den korrekten Pod-Netzwerk-Namespace.

Dieser Ablauf zeigt, wie CNI NAT-ing die Konfiguration vereinfachen kann, indem Pakete zurück zum Knoten geleitet werden können, ohne dass zusätzliches Routing für den Pod erforderlich ist. CIDRs

EKS-Steuerebene für Pods, die in einem Hybridknoten ausgeführt werden (Webhooks)



Dieses Datenverkehrsmuster tritt am häufigsten bei Webhooks auf, bei denen die EKS-Steuerebene direkt Verbindungen zu Webhook-Servern herstellen muss, die in Pods auf Hybridknoten ausgeführt werden. Beispiele hierfür sind die Validierung und Mutation von Zulassungs-Webhooks, die vom API-Server während der Ressourcenvalidierung oder Mutationsprozesse aufgerufen werden.

Anforderung

1. EKS-Kubernetes-API-Server initiiert Anfrage

Wenn ein Webhook im Cluster konfiguriert ist und durch einen entsprechenden API-Vorgang ausgelöst wird, muss der EKS-Kubernetes-API-Server eine direkte Verbindung zum Webhook-Server-Pod herstellen. Der API-Server sucht zunächst die IP-Adresse des Pods aus der dem Webhook zugeordneten Service- oder Endpunkt-Ressource.

Vorausgesetzt, der Webhook-Pod wird in einem Hybridknoten mit IP `10.85.1.23` ausgeführt, erstellt der EKS-Kubernetes-API-Server eine HTTPS-Anfrage an den Webhook-Endpunkt. Das erste Paket wird über die EKS-Steuerebene ENI in Ihrem VPC gesendet, da die Ziel-IP `10.85.1.23` zum konfigurierten Fern-Pod-CIDR (`10.85.0.0/16`) gehört. Das Paket sieht wie folgt aus:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.0.0.132 | Src: 41892 (random) |                 |
| Dst: 10.85.1.23 | Dst: 8443           |                 |
+-----------------+---------------------+-----------------+
```

2. VPC-Netzwerkverarbeitung

Das Paket verlässt die EKS-Steuerebene (ENI) und gelangt mit dem Router des Subnetzes als nächstem Hop in die VPC-Netzwerkebene.

3. Abfrage der VPC-Routing-Tabelle

Die VPC-Routing-Tabelle für das Subnetz, das die EKS-Steuerebene ENI enthält, enthält eine bestimmte Route für den Fern-Pod-CIDR (`10.85.0.0/16`). Diese Routing-Regel leitet das Paket an das VPC-to-onprem Gateway weiter (z. B. ein Virtual Private Gateway für Direct Connect- oder VPN-Verbindungen):

```
Destination     Target
10.0.0.0/16     local
10.85.0.0/16    vgw-id (VPC-to-onprem gateway)
```

4. Grenzüberschreitender Transit

Das Gateway überträgt das Paket über die Cloud-Grenze hinweg über Ihre vorhandene Verbindung (z. B. Direct Connect oder VPN) an Ihr On-Premises-Netzwerk. Das Paket behält seine ursprünglichen Quell- und Ziel-IP-Adressen bei, während es diese Verbindung durchläuft.

5. On-Premises-Netzwerkempfang

Das Paket erreicht Ihren lokalen On-Premises-Router. Der Router fragt seine Routing-Tabelle ab, um zu ermitteln, wie die Adresse 10.85.1.23 erreicht werden kann. Damit dies funktioniert, muss Ihr lokales Netzwerk über Routen für den Pod verfügen CIDRs , die den Datenverkehr an den entsprechenden Hybridknoten weiterleiten.

In diesem Fall enthält die Routing-Tabelle des Routers einen Eintrag, der angibt, dass das `10.85.1.0/24`-Subnetz über den Hybridknoten mit der IP `10.80.0.2` erreichbar ist:

```
Destination     Next Hop
10.85.1.0/24    10.80.0.2
```

6. Zustellung an den Knoten

Basierend auf dem Eintrag in der Routing-Tabelle leitet der Router das Paket an den Hybridknoten (`10.80.0.2`) weiter. Wenn das Paket am Knoten ankommt, sieht es genauso aus wie beim Versand durch den EKS-Kubernetes-API-Server, wobei die Ziel-IP weiterhin die IP des Pods ist.

7. CNI-Verarbeitung

Der Netzwerk-Stack des Knotens empfängt das Paket und leitet es zur Verarbeitung an das CNI weiter, da er erkennt, dass die Ziel-IP nicht die eigene IP des Knotens ist. Das CNI erkennt, dass die Ziel-IP zu einem Pod gehört, der lokal auf diesem Knoten ausgeführt wird, und leitet das Paket über die entsprechenden virtuellen Schnittstellen an den richtigen Pod weiter:

```
Original packet -> node routing -> CNI -> Pod's network namespace
```

Der Webhook-Server im Pod empfängt die Anfrage und verarbeitet sie.

Antwort

Nachdem der Webhook-Pod die Anfrage verarbeitet hat, sendet er eine Antwort zurück, die demselben Pfad in umgekehrter Reihenfolge folgt:

7. Pod sendet Antwort

Der Webhook-Pod erstellt ein Antwortpaket mit seiner eigenen IP-Adresse als Quelle und dem ursprünglichen Anforderer (der EKS-Steuerebenen-ENI) als Ziel:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.85.1.23 | Src: 8443           |                 |
| Dst: 10.0.0.132 | Dst: 41892          |                 |
+-----------------+---------------------+-----------------+
```

Das CNI erkennt, dass dieses Paket an ein externes Netzwerk (nicht an einen lokalen Pod) gesendet wird, und leitet das Paket unter Beibehaltung der ursprünglichen Quell-IP-Adresse an den Netzwerk-Stack des Knotens weiter.

6. Knoten-Netzwerkverarbeitung

Der Knoten stellt fest, dass sich die Ziel-IP (`10.0.0.132`) nicht im lokalen Netzwerk befindet, und leitet das Paket an sein Standard-Gateway (den lokalen Router) weiter.

5. Lokales Router-Routing

Der lokale Router fragt seine Routing-Tabelle ab und stellt fest, dass die Ziel-IP (`10.0.0.132`) zum VPC CIDR (`10.0.0.0/16`) gehört. Er leitet das Paket an das Gateway weiter, das eine Verbindung zu AWS herstellt.

4. Grenzüberschreitender Transit

Das Paket wird über dieselbe On-Premises-VPC-Verbindung zurückgesendet und durchquert die Cloud-Grenze in umgekehrter Richtung.

3. VPC-Routing

Wenn das Paket in der VPC ankommt, erkennen die Routing-Tabellen, dass die Ziel-IP zu einem Subnetz innerhalb der VPC gehört. Das Paket wird innerhalb der VPC entsprechend weitergeleitet.

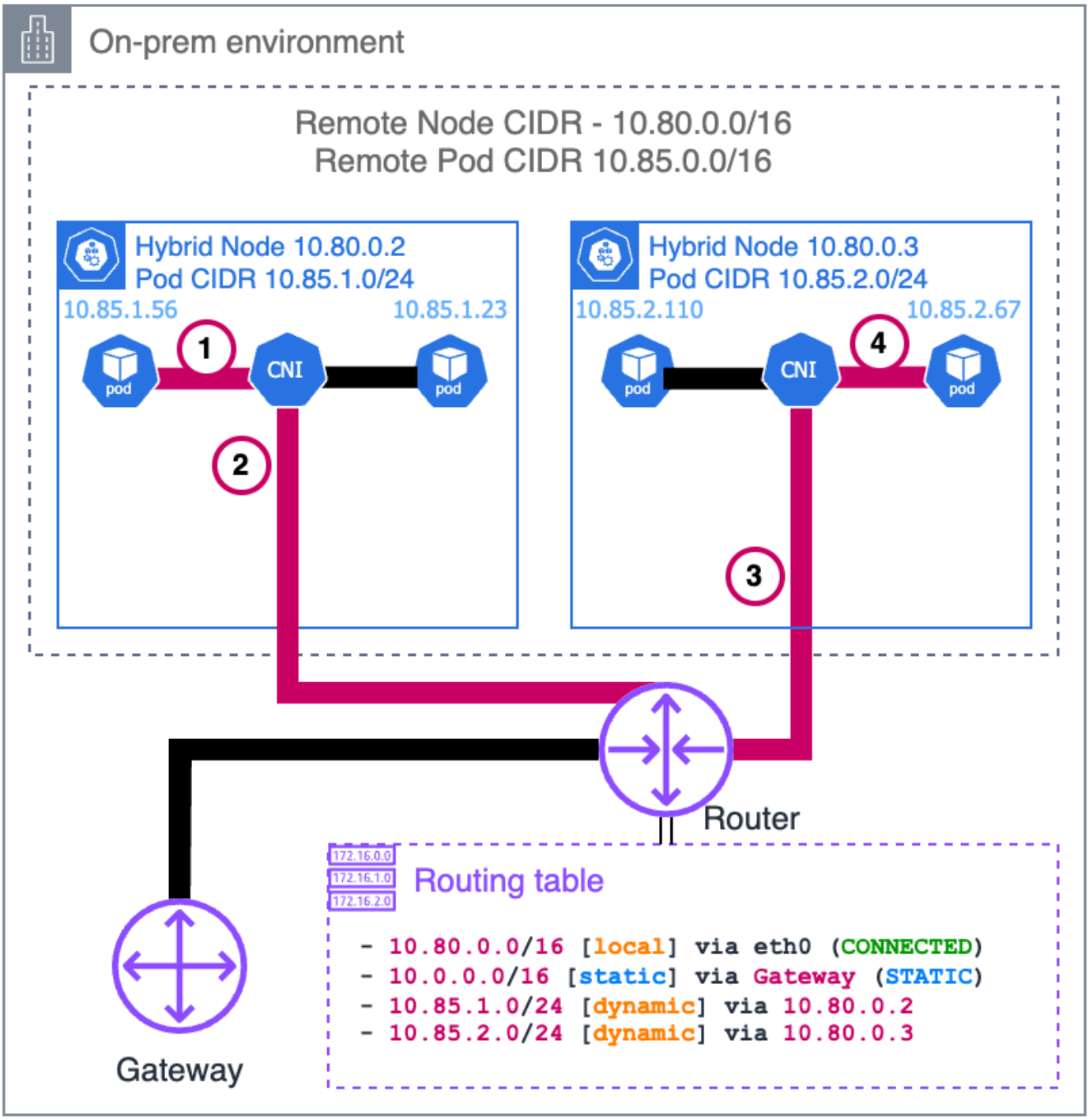
2. und 1. ENI-Empfang der EKS-Steuerebene

Das Paket erreicht das ENI, das mit dem EKS-Kubernetes-API-Server verbunden ist, und schließt damit den Rundlauf ab. Der API-Server empfängt die Webhook-Antwort und setzt die Verarbeitung der ursprünglichen API-Anforderung basierend auf dieser Antwort fort.

Dieser Verkehrsfluss zeigt, warum der Remote-Pod ordnungsgemäß konfiguriert und geroutet werden CIDRs muss:

- Die VPC muss über Routen für den Remote-Pod verfügen, die auf das lokale Gateway CIDRs verweisen
- Ihr lokales Netzwerk muss über Routen für Pods verfügen CIDRs , die den Datenverkehr zu den spezifischen Knoten weiterleiten, auf denen diese Pods gehostet werden
- Ohne diese Routing-Konfiguration wären Webhooks und andere ähnliche Services, die in Pods auf Hybridknoten ausgeführt werden, von der EKS-Steuerebene aus nicht erreichbar.

Pod-to-Pod läuft auf Hybridknoten



In diesem Abschnitt wird erläutert, wie Pods, die in verschiedenen Hybridknoten ausgeführt werden, miteinander kommunizieren. In diesem Beispiel wird davon ausgegangen, dass Ihr CNI VXLAN für

die Kapselung verwendet, was bei Cilium CNIs oder Calico üblich ist. Der Gesamtprozess ist bei anderen Kapselungsprotokollen wie Geneve oder ähnlich. IP-in-IP

Anforderung

1. Pod A initiiert die Kommunikation

Pod A (`10.85.1.56`) on Knoten 1 möchte Datenverkehr an Pod B (`10.85.2.67`) an Knoten 2 senden. Das ursprüngliche Paket sieht folgendermaßen aus:

```
+------------------+-----------------+-------------+-----------------+
| Ethernet Header  | IP Header       | TCP/UDP     | Payload         |
| Src: Pod A MAC   | Src: 10.85.1.56 | Src: 43721  |                 |
| Dst: Gateway MAC | Dst: 10.85.2.67 | Dst: 8080   |                 |
+------------------+-----------------+-------------+-----------------+
```

2. CNI fängt das Paket ab und verarbeitet es

Wenn das Paket von Pod A seinen Netzwerk-Namespace verlässt, wird es vom CNI abgefange Das CNI konsultiert seine Routing-Tabelle und stellt fest: – Die Ziel-IP (`10.85.2.67`) gehört zum Pod-CIDR. – Diese IP befindet sich nicht auf dem lokalen Knoten, sondern gehört zu Knoten 2 (`10.80.0.3`). – Das Paket muss mit VXLAN gekapselt werden.

Die Entscheidung für die Kapselung ist von entscheidender Bedeutung, da das zugrunde liegende physische Netzwerk nicht weiß, wie der Pod CIDRs direkt weitergeleitet wird — es weiß nur, wie der Verkehr zwischen den Knoten weitergeleitet wird. IPs

Das CNI kapselt das gesamte ursprüngliche Paket in einem VXLAN-Frame. Dadurch wird effektiv ein „Paket innerhalb eines Pakets“ mit neuen Headern erstellt:

```
+-----------------+----------------+-------------+-----------
+--------------------------+
| Outer Ethernet  | Outer IP       | Outer UDP    | VXLAN      | Original Pod-to-Pod
    |
| Src: Node1 MAC  | Src: 10.80.0.2 | Src: Random  | VNI: 42    | Packet (unchanged
    |
| Dst: Router MAC | Dst: 10.80.0.3 | Dst: 8472    |            | from above)
    |
+-----------------+----------------+-------------+-----------
+--------------------------+
```

Wichtige Punkte zu dieser Kapselung: – Das äußere Paket ist von Knoten 1 (`10.80.0.2`) an Knoten 2 (`10.80.0.3`) adressiert. – Der UDP-Port `8472` ist der VXLAN-Port, den Cilium standardmäßig verwendet. – Die VXLAN-Netzwerk-ID (VNI) identifiziert, zu welchem Überlagerungsnetzwerk dieses Paket gehört. – Das gesamte Originalpaket (mit der IP-Adresse von Pod A als Quelle und der IP-Adresse von Pod B als Ziel) bleibt im Inneren unverändert

Das gekapselte Paket gelangt nun in den regulären Netzwerk-Stack von Knoten 1 und wird wie jedes andere Paket verarbeitet:

1. Knoten-Netzwerkverarbeitung: Der Netzwerk-Stack von Knoten 1 leitet das Paket basierend auf seinem Ziel weiter (`10.80.0.3`)
2. Bereitstellung im lokalen Netzwerk:
   - Wenn sich beide Knoten im selben Layer-2-Netzwerk befinden, wird das Paket direkt an Knoten 2 gesendet
   - Wenn sie sich in unterschiedlichen Subnetzen befinden, wird das Paket zuerst an den lokalen Router weitergeleitet
3. Router-Verarbeitung: Der Router leitet das Paket anhand seiner Routing-Tabelle weiter und übergibt es an Knoten 2

3. Verarbeitung des empfangenden Knotens

Wenn das gekapselte Paket bei Knoten 2 (`10.80.0.3`) ankommt:

1. Der Netzwerk-Stack des Knotens empfängt es und identifiziert es als VXLAN-Paket (UDP-Port `4789`).
2. Das Paket wird zur Verarbeitung an die VXLAN-Schnittstelle des CNI weitergeleitet

4. VXLAN-Entkapselung

Das CNI auf Knoten 2 verarbeitet das VXLAN-Paket:

1. Es entfernt die äußeren Header (Ethernet, IP, UDP und VXLAN).
2. Es extrahiert das ursprüngliche innere Paket
3. Das Paket hat nun wieder seine ursprüngliche Form:

```
+------------------+----------------+-------------+-----------------+
```

```
| Ethernet Header  | IP Header       | TCP/UDP     | Payload         |
| Src: Pod A MAC   | Src: 10.85.1.56 | Src: 43721  |                 |
| Dst: Gateway MAC | Dst: 10.85.2.67 | Dst: 8080   |                 |
+------------------+-----------------+-------------+-----------------+
```

Das CNI auf Knoten 2 untersucht die Ziel-IP (`10.85.2.67`) und:

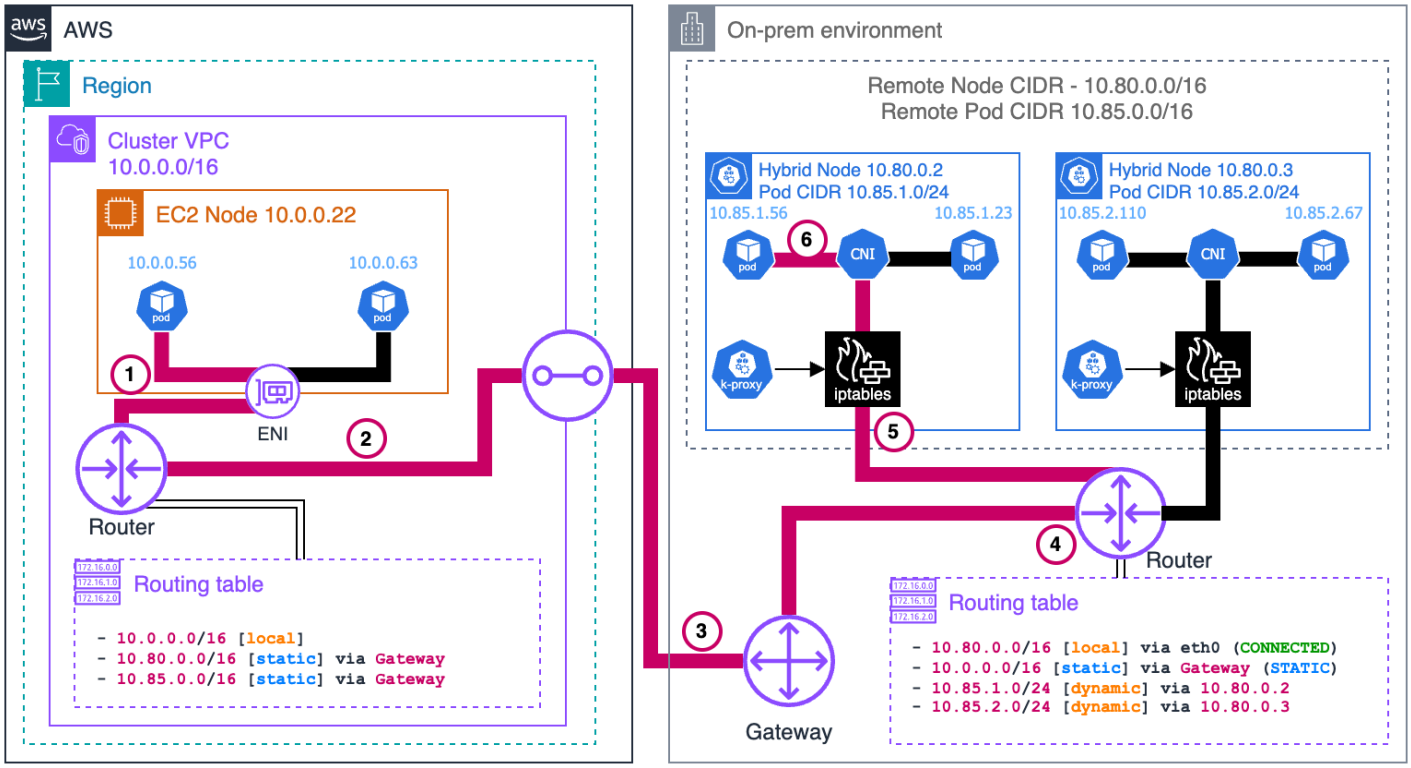
1. Identifiziert, dass diese IP zu einem lokalen Pod gehört
2. Leitet das Paket durch die entsprechenden virtuellen Schnittstellen
3. Liefert das Paket an den Netzwerk-Namespace von Pod B

Antwort

Wenn Pod B auf Pod A reagiert, verläuft der gesamte Prozess in umgekehrter Reihenfolge:

4. Pod B sendet ein Paket an Pod A (`10.85.1.56`)
5. Das CNI von Knoten 2 kapselt ihn mit VXLAN und legt das Ziel auf Knoten 1 (`10.80.0.2`) fest
6. Das gekapselte Paket wird an Knoten 1 übermittelt
7. Knoten 1 entkapselt die CNI und übermittelt die ursprüngliche Antwort an Pod A

Pods auf Cloud-Knoten zu Pods auf Hybridknoten (Ost-West–Datenverkehr)



Anforderung

1. Pod A initiiert die Kommunikation

Pod A (`10.0.0.56`) auf dem EC2 Node möchte Traffic an Pod B (`10.85.1.56`) auf dem Hybrid-Node senden. Das ursprüngliche Paket sieht folgendermaßen aus:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.0.0.56  | Src: 52390 (random) |                 |
| Dst: 10.85.1.56 | Dst: 8080           |                 |
+-----------------+---------------------+-----------------+
```

Mit der VPC CNI hat Pod A eine IP von der VPC CIDR und ist direkt mit einer ENI auf der Instance verbunden. EC2 Der Netzwerk-Namespace des Pods ist mit dem VPC-Netzwerk verbunden, sodass das Paket direkt in die VPC-Routing-Infrastruktur gelangt.

2. VPC-Routing

Die VPC-Routentabelle enthält eine spezifische Route für den Remote Pod CIDR (`10.85.0.0/16`), die diesen Verkehr an das Gateway weiterleitet: VPC-to-onprem

```
Destination     Target
10.0.0.0/16     local
10.85.0.0/16    vgw-id (VPC-to-onprem gateway)
```

Basierend auf dieser Routing-Regel wird das Paket an das Gateway weitergeleitet, das mit Ihrem On-Premises-Netzwerk verbunden ist.

3. Grenzüberschreitender Transit

Das Gateway überträgt das Paket über die Cloud-Grenze hinweg über Ihre vorhandene Verbindung (z. B. Direct Connect oder VPN) an Ihr On-Premises-Netzwerk. Das Paket behält während der gesamten Übertragung seine ursprünglichen Quell- und Ziel-IP-Adressen bei.

4. On-Premises-Netzwerkempfang

Das Paket erreicht Ihren lokalen On-Premises-Router. Der Router fragt seine Routing-Tabelle ab, um den nächsten Hop zum Erreichen der Adresse 10.85.1.56 zu ermitteln. Ihr lokaler Router muss über Routen für den Pod verfügen CIDRs , die den Datenverkehr zum entsprechenden Hybridknoten weiterleiten.

Die Tabelle des Routers enthält einen Eintrag, der angibt, dass das `10.85.1.0/24`-Subnetz über den Hybridknoten mit der IP `10.80.0.2` erreichbar ist:

```
Destination     Next Hop
10.85.1.0/24    10.80.0.2
```

5. Knoten-Netzwerkverarbeitung

Der Router leitet das Paket an den Hybridknoten weiter (`10.80.0.2`). Wenn das Paket am Knoten ankommt, enthält es weiterhin die IP-Adresse von Pod A als Quelle und die IP-Adresse von Pod B als Ziel.

6. CNI-Verarbeitung

Der Netzwerk-Stack des Knotens empfängt das Paket und leitet es, da die Ziel-IP-Adresse nicht seine eigene ist, zur Verarbeitung an das CNI weiter. Das CNI erkennt, dass die Ziel-IP zu einem Pod gehört, der lokal auf diesem Knoten ausgeführt wird, und leitet das Paket über die entsprechenden virtuellen Schnittstellen an den richtigen Pod weiter:

```
Original packet -> node routing -> CNI -> Pod B's network namespace
```

Pod B empfängt das Paket und verarbeitet es entsprechend den Anforderungen.

Antwort

6. Pod B sendet Antwort

Pod B erstellt ein Antwortpaket mit seiner eigenen IP als Quelle und der IP von Pod A als Ziel:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.85.1.56 | Src: 8080           |                 |
| Dst: 10.0.0.56  | Dst: 52390          |                 |
+-----------------+---------------------+-----------------+
```

Das CNI erkennt, dass dieses Paket für ein externes Netzwerk bestimmt ist, und leitet es an den Netz-Stack des Knotens weiter.

5. Knoten-Netzwerkverarbeitung

Der Knoten stellt fest, dass die Ziel-IP (`10.0.0.56`) nicht zum lokalen Netzwerk gehört und leitet das Paket an sein Standard-Gateway (den lokalen Router) weiter.

4. Lokales Router-Routing

Der lokale Router fragt seine Routing-Tabelle ab und stellt fest, dass die Ziel-IP (`10.0.0.56`) zum VPC CIDR (`10.0.0.0/16`) gehört. Er leitet das Paket an das Gateway weiter, das eine Verbindung zu AWS herstellt.

3. Grenzüberschreitender Transit

Das Paket wird über dieselbe On-Premises-VPC-Verbindung zurückgesendet und durchquert die Cloud-Grenze in umgekehrter Richtung.

2. VPC-Routing

Wenn das Paket in der VPC ankommt, erkennt das Routing-System, dass die Ziel-IP zu einem Subnetz innerhalb der VPC gehört. Das Paket wird durch das VPC-Netzwerk an die EC2 Instance weitergeleitet, die Pod A hostet.

1. Pod A empfängt Antwort

Das Paket kommt bei der EC2 Instance an und wird über das angehängte ENI direkt an Pod A übermittelt. Da die VPC CNI für Pods in der VPC kein Überlagerungsnetzwerk verwendet, ist keine zusätzliche Entkapselung erforderlich – das Paket kommt mit intakten Original-Headern an.

Dieser Ost-West-Verkehrsfluss zeigt, warum der Remote-Pod ordnungsgemäß konfiguriert und aus beiden Richtungen routbar sein CIDRs muss:

- Die VPC muss über Routen für den Remote-Pod verfügen, die auf das lokale Gateway CIDRs verweisen
- Ihr lokales Netzwerk muss über Routen für Pods verfügen CIDRs , die den Datenverkehr zu den spezifischen Knoten weiterleiten, auf denen diese Pods gehostet werden.

`nodeadm`-Referenz für Hybridknoten

Die Amazon EKS Hybrid Nodes CLI (`nodeadm`) vereinfacht die Installation, Konfiguration, Registrierung und Deinstallation der Hybridknoten-Komponenten. Sie können `nodeadm` in Ihre Betriebssystem-Images einbinden, um das Bootstrapping von Hybridknoten zu automatisieren. Weitere Informationen finden Sie unter the section called “Betriebssystem vorbereiten”.

Die `nodeadm` Version für Hybridknoten unterscheidet sich von der `nodeadm` Version, die für das Bootstrapping von EC2 Amazon-Instances als Knoten in Amazon EKS-Clustern verwendet wird. Befolgen Sie die Dokumentation und Referenzen für die entsprechende `nodeadm`-Version. Diese Dokumentationsseite bezieht sich auf die `nodeadm`-Version für Hybridknoten.

Der Quellcode für die Hybridknoten `nodeadm` ist im https://github.com/aws/ GitHub EKS-Hybrid-Repository veröffentlicht.

> **Important**
>
> Sie müssen `nodeadm` mit einem Benutzer arbeiten, der über Rechte verfügt root/sudo .

Herunterladen von `nodeadm`

Die Hybridknoten-Version von `nodeadm` wird in Amazon S3 gehostet, das von Amazon CloudFront bereitgestellt wird. Um die Installation auf jedem On-Premises-Host `nodeadm` durchzuführen, können Sie den folgenden Befehl von Ihren On-Premises-Hosts aus ausführen.

Für x86_64-Hosts

```
curl -OL 'https://hybrid-assets.eks.amazonaws.com/releases/latest/bin/linux/amd64/
nodeadm'
```

Für ARM-Hosts

```
curl -OL 'https://hybrid-assets.eks.amazonaws.com/releases/latest/bin/linux/arm64/
nodeadm'
```

Fügen Sie der heruntergeladenen Binärdatei auf jedem Host die Berechtigung zum Ausführen von Dateien hinzu.

```
chmod +x nodeadm
```

`nodeadm install`

Der `nodeadm install`-Befehl wird verwendet, um die Artefakte und Abhängigkeiten zu installieren, die zum Ausführen und Verbinden von Hybridknoten mit einem Amazon-EKS-Cluster erforderlich sind. Der `nodeadm install`-Befehl kann einzeln auf jedem Hybridknoten oder während Image-Entwicklungs-Pipelines ausgeführt werden, um die Abhängigkeiten der Hybridknoten in Betriebssystem-Images vorzuinstallieren.

Usage

```
nodeadm install [KUBERNETES_VERSION] [flags]
```

Positionelle Argumente

(Erforderlich) `KUBERNETES_VERSION` Die zu installierende Haupt- und Nebenversion von EKS Kubernetes, zum Beispiel `1.32`

Flags

Name	Erforderlich	Description
`-p`, `--credential-provider`	TRUE	Zu installierender Anmeldeinformationsanbieter. Unterstützte Werte sind `iam-ra` und `ssm`. Weitere Informationen

Name	Erforderlich	Description
		finden Sie unter the section called “Anmeldeinformationen vorbereiten”.

Name	Erforderlich	Description
`-s`, `--containerd-source`	FALSE	Quelle für `containerd` . `nodeadm` unterstützt die Installation von `containerd` aus der Betriebssystemvertreilung, Docker-Pakete und das Überspringen der `containerd` -Installation. Werte `distro`- Dies ist der Standardwert. `nodeadm`installiert das neueste vom Node-Betriebssystem verteilte `containerd` Paket, das mit der EKS Kubernetes-Version kompatibel ist. `distro`ist kein unterstützter Wert für Red Hat Enterprise Linux (RHEL) -Betriebssysteme. `docker`- installiert `nodeadm` das neueste von Docker erstellte und vertriebene `containerd` Paket, das mit der EKS Kubernetes-Version kompatibel ist. `docker`ist kein unterstützter Wert für Amazon Linux 2023. `none` – `nodeadm` installiert das `containerd` -Paket nicht. Sie müssen `containerd` manuell installieren, bevor

Name	Erforderlich	Description
		Sie `nodeadm init` ausführen .
`-r`, `--region`	FALSE	Gibt die AWS Region für das Herunterladen von Artefakte n wie dem SSM-Agenten an. Standardeinstellung: `us-west-2` .
`-t`, `--timeout`	FALSE	Maximale Dauer des Installat ionsbefehls. Die Eingabe erfolgt im Zeitdauerformat. Zum Beispiel `1h23m`. Die Standard-Zeitüberschreitung für den Download beim Installationsbefehl beträgt 20 Minuten.
`-h`, `--help`	FALSE	Zeigt eine Hilfemeldung mit verfügbaren Flag-, Unterbefe hls- und Positionswertparam etern an.

Beispiele

Installieren Sie die Kubernetes-Version `1.32` mit AWS Systems Manager (SSM) als Anmeldeinformationsanbieter

```
nodeadm install 1.32 --credential-provider ssm
```

Installieren Sie die Kubernetes-Version `1.32` mit AWS Systems Manager (SSM) als Anmeldeinformationsanbieter und Docker als Container-Quelle mit einem Download-Timeout von 20 Minuten.

```
nodeadm install 1.32 --credential-provider ssm --containerd-source docker --timeout 20m
```

Installieren Sie die Kubernetes-Version mit IAM Roles Anywhere als Anmeldeinformationsanbieter `1.32` AWS

```
nodeadm install 1.32 --credential-provider iam-ra
```

`nodeadm config check`

Der `nodeadm config check`-Befehl überprüft die angegebene Knotenkonfiguration auf Fehler. Mit diesem Befehl kann die Richtigkeit einer Hybridknoten-Konfigurationsdatei überprüft und validiert werden.

Usage

```
nodeadm config check [flags]
```

Flags

Name	Erforderlich	Description
`-c`, `--config-source`	TRUE	Quelle der nodeadm-K onfiguration. Bei Hybridknoten sollte die Eingabe einer URI mit Dateischema folgen.
`-h`, `--help`	FALSE	Zeigt eine Hilfemeldung mit verfügbaren Flag-, Unterbefe hls- und Positionswertparam etern an.

Beispiele

```
nodeadm config check -c file://nodeConfig.yaml
```

`nodeadm init`

Der `nodeadm init`-Befehl startet und verbindet den Hybridknoten mit dem konfigurierten Amazon-EKS-Cluster. Einzelheiten zur Konfiguration der `nodeConfig.yaml`-Datei finden Sie unter

the section called “Knotenkonfiguration für SSM-Hybridaktivierungen” oder the section called “Knotenkonfiguration für IAM Roles Anywhere”.

Usage

```
nodeadm init [flags]
```

Flags

Name	Erforderlich	Description
`-c`, `--config-source`	TRUE	Quelle der `nodeadm`-Konfiguration. Bei Hybridknoten sollte die Eingabe einer URI mit Dateischema folgen.
`-s`, `--skip`	FALSE	Zu überspringende Phasen von `init`. Es wird nicht empfohlen, eine der Phasen zu überspringen, es sei denn, dies trägt zur Behebung eines Problems bei. Werte `install-validation` überspringt die Überprüfung, ob der vorherige Installationsbefehl erfolgreich ausgeführt wurde. `cni-validation` überspringt die Überprüfung, ob die VXLAN-Ports von Cilium oder Calico CNI geöffnet sind, wenn die Firewall auf dem Knoten aktiviert ist.

Name	Erforderlich	Description
		`node-ip-validation` überspringt die Überprüfung, ob die Knoten-IP innerhalb eines CIDR in den Netzwerken der Fern-Knoten liegt.
`-h`, `--help`	FALSE	Zeigt eine Hilfemeldung mit verfügbaren Flag-, Unterbefehls- und Positionswertparametern an.

Beispiele

```
nodeadm init -c file://nodeConfig.yaml
```

`nodeadm upgrade`

Der `nodeadm upgrade`-Befehl aktualisiert alle installierten Artefakte auf die neueste Version und führt einen Bootstrapping des Knotens durch, um die aktualisierten Artefakte zu konfigurieren und dem EKS-Cluster in AWS beizutreten. Ein Upgrade ist ein Befehl zur Unterbrechung der auf dem Knoten ausgeführten Workloads. Verschieben Sie Ihre Workloads vor dem Upgrade auf einen anderen Knoten.

Usage

```
nodeadm upgrade [KUBERNETES_VERSION] [flags]
```

Positionelle Argumente

(Erforderlich) `KUBERNETES_VERSION` Die zu installierende Haupt- und Nebenversion von EKS Kubernetes, zum Beispiel `1.32`

Flags

Name	Erforderlich	Description
`-c`, `--config-source`	TRUE	Quelle der `nodeadm`-Konfiguration. Bei Hybridknoten sollte die Eingabe einer URI mit Dateischema folgen.
`-t`, `--timeout`	FALSE	Zeitüberschreitung beim Herunterladen von Artefakten. Die Eingabe erfolgt im Zeitdauerformat. Beispielsweise 1 Stunde und 23 Minuten Die Standard-Zeitüberschreitung für den Download des Upgrade-Befehls beträgt 10 Minuten.
`-s`, `--skip`	FALSE	Zu überspringende Phasen des Upgrades. Es wird nicht empfohlen, eine der Phasen zu überspringen, es sei denn, dies trägt zur Behebung eines Problems bei Werte `pod-validation` überspringt die Überprüfung, ob alle Pods auf dem Knoten ausgeführt werden, mit Ausnahme von Daemon-Sets und statischen Pods. `node-validation` überspringt die Überprüfung, ob der Knoten abgesperrt wurde.

Name	Erforderlich	Description
		`init-validation` überspringt die Überprüfung, ob der Knoten erfolgreich initialisiert wurde, bevor das Upgrade ausgeführt wird. `containerd-major-version-upgrade` verhindert Container-Hauptversions-Upgrades während des Knoten-Upgrades.
`-h, --help`	FALSE	Zeigt eine Hilfemeldung mit verfügbaren Flag-, Unterbefehls- und Positionswertparametern an.

Beispiele

```
nodeadm upgrade 1.32 -c file://nodeConfig.yaml
```

```
nodeadm upgrade 1.32 -c file://nodeConfig.yaml --timeout 20m
```

`nodeadm uninstall`

Der `nodeadm uninstall`-Befehl stoppt und entfernt die während `nodeadm install` installierten Artefakte `nodeadm`, einschließlich kubelet und containerd. Beachten Sie, dass der Deinstallations-Befehl Ihre Hybridknoten nicht aus Ihrem Cluster entfernt oder löscht. Sie müssen die Entleerungs- und Löschvorgänge separat ausführen. Weitere Informationen finden Sie unter the section called “Hybridknoten löschen”. Standardmäßig wird `nodeadm uninstall` nicht fortgesetzt, wenn auf dem Knoten noch Pods vorhanden sind. Ebenso entfernt `nodeadm uninstall` keine CNI-Abhängigkeiten oder Abhängigkeiten anderer Kubernetes-Add-Ons, die Sie in Ihrem Cluster ausführen. Informationen zum vollständigen Entfernen der CNI-Installation von Ihrem Host finden Sie in den Anweisungen unter the section called “CNI konfigurieren”. Wenn Sie AWS SSM-Hybrid-

Aktivierungen als Anbieter für lokale Anmeldeinformationen verwenden, werden Ihre Hosts mit dem `nodeadm uninstall` Befehl als SSM-verwaltete Instanzen deregistriert. AWS

Usage

```
nodeadm uninstall [flags]
```

Flags

Name	Erforderlich	Description
`-s`, `--skip`	FALSE	Zu überspringende Phasen der Deinstallation. Es wird nicht empfohlen, eine der Phasen zu überspringen, es sei denn, dies trägt zur Behebung eines Problems bei. Werte `pod-validation` überspringt die Überprüfung, ob alle Pods auf dem Knoten ausgeführt werden, mit Ausnahme von Daemon-Sets und statischen Pods. `node-validation` überspringt die Überprüfung, ob der Knoten abgesperrt wurde. `init-validation` überspringt die Überprüfung, ob der Knoten erfolgreich initialisiert wurde, bevor die Deinstallation ausgeführt wird.

Name	Erforderlich	Description
`-h`, `--help`	FALSE	Zeigt eine Hilfemeldung mit verfügbaren Flag-, Unterbefehls- und Positionswertparametern an.

Name	Erforderlich	Description
`-f`, `--force`	FALSE	Erzwingen Sie das Löschen zusätzlicher Verzeichnisse, die möglicherweise verbleibende Dateien von Kubernetes- und CNI-Komponenten enthalten. WARNUNG Dadurch werden alle Inhalte in den Standardverzeichnissen von Kubernetes und CNI (`/var/lib/cni` , `/etc/cni/net.d` usw.) gelöscht. Verwenden Sie dieses Flag nicht, wenn Sie Ihre eigenen Daten an diesen Speicherorten speichern. Ab nodeadm `v1.0.9` löscht der `./nodeadm uninstall --skip node-validation,pod-validation --force` -Befehl das `/var/lib/kubelet` -Verzeichnis nicht mehr. Dies liegt daran, dass es Pod-Volumes und Volume-Unterpfad-Verzeichnisse enthalten kann, die manchmal das eingebundene Knoten-Dateisystem enthalten. Tipps zur sicheren Handhabung

Name	Erforderlich	Description
		- Das Löschen von eingebundenen Pfaden kann zum versehentlichen Löschen des tatsächlich eingebundenen Dateisystems des Knotens führen. Bevor Sie das `/var/lib/kubelet` -Verzeichnis manuell löschen, überprüfen Sie sorgfältig alle aktiven Einbindungen und binden Sie alle Volumes sicher aus, um Datenverlust zu vermeiden.

Beispiele

```
nodeadm uninstall
```

```
nodeadm uninstall --skip node-validation,pod-validation
```

`nodeadm debug`

Der `nodeadm debug`-Befehl kann zur Fehlerbehebung bei fehlerhaften oder falsch konfigurierten Hybridknoten verwendet werden. Es bestätigt, dass die folgenden Anforderungen erfüllt sind.

- Der Knoten hat Netzwerkzugriff auf die zum Abrufen der Anmeldeinformationen erforderlichen AWS APIs
- Der Knoten ist in der Lage, AWS Anmeldeinformationen für die konfigurierte IAM-Rolle für Hybridknoten abzurufen.
- Der Knoten verfügt über Netzwerkzugriff auf den EKS-Kubernetes-API-Endpunkt und die Gültigkeit des EKS-Kubernetes-API-Endpunktzertifikats,
- Der Knoten kann sich beim EKS-Cluster authentifizieren, seine Identität im Cluster ist gültig und der Knoten hat Zugriff auf den EKS-Cluster über die für den EKS-Cluster konfigurierte VPC.

Wenn Fehler gefunden werden, werden in der Ausgabe des Befehls Schritte zur Fehlerbehebung vorgeschlagen. Bestimmte Validierungsschritte zeigen untergeordnete Prozesse an. Wenn diese fehlschlagen, wird die Ausgabe in einem stderr-Abschnitt unter dem Validierungsfehler angezeigt.

Usage

```
nodeadm debug [flags]
```

Flags

Name	Erforderlich	Description
`-c`, `--config-source`	TRUE	Quelle der `nodeadm`-Konfiguration. Bei Hybridknoten sollte die Eingabe einer URI mit Dateischema folgen.
`--no-color`	FALSE	Deaktiviert die Farbausgabe. Nützlich für die Automatisierung.
`-h`, `--help`	FALSE	Zeigt eine Hilfemeldung mit verfügbaren Flag-, Unterbefehls- und Positionswertparametern an.

Beispiele

```
nodeadm debug -c file://nodeConfig.yaml
```

Speicherorte für Nodeadm-Dateien

nodeadm-Installation

Bei der Ausführung von `nodeadm install` werden die folgenden Dateien und Dateispeicherorte konfiguriert.

Artefakt	Pfad
CLI für IAM Roles Anywhere	/_signing_helper usr/local/bin/aws
Kubelet-Binärdateien	/usr/bin/kubelet
Kubectl-Binärdatei	usr/local/bin/kubectl
ECR-Anmeldeinformationsanbieter	/etc/eks/image-credential-provider/ecr-Anmeldeinformationsanbieter
AWS IAM-Authentifikator	/-iam-Authentifikator usr/local/bin/aws
CLI für SSM-Einrichtung	/opt/ssm/ssm-setup-cli
SSM-Agent	Auf Ubuntu -/-ssm-agent snap/amazon-ssm-agent/current/amazon Auf RHEL & AL2 023 -/-ssm-agent usr/bin/amazon
Containerd	Auf Ubuntu & 023 -/ AL2usr/bin/containerd Auf RHEL – /bin/containerd
Iptables	Auf Ubuntu & AL2 023 -/usr/sbin/iptables Auf RHEL – /sbin/iptables
CNI-Plugins	/opt/cni/bin
Nachverfolgung installierter Artefakte	/opt/nodeadm/tracker

nodeadm init

Bei der Ausführung von `nodeadm init` werden die folgenden Dateien und Dateispeicherorte konfiguriert.

Name	Pfad
Kubelet kubeconfig	/var/lib/kubelet/kubeconfig
Kubelet-Konfiguration	/.json etc/kubernetes/kubelet/config
Kubelet-systemd-Einheit	/.dienst etc/systemd/system/kubelet
Konfiguration des Image-Anmeldeinformationsanbieters	/.json etc/eks/image-credential-provider/config
Kubelet-env-Datei	/etc/eks/kubelet/environment
Kubelet Certs	/.crt etc/kubernetes/pki/ca
Containerd-Konfiguration	/.toml etc/containerd/config
Konfiguration der Containerd-Kernelmodule	/.conf etc/modules-load.d/contianerd
AWS Konfigurationsdatei	/etc/aws/hybrid/config
AWS Anmeldeinformationsdatei (wenn Anmeldeinformationsdatei aktiviert ist)	/eks-hybrid/.aws/credentials
AWS Helper-Systemeinheit signieren	/etc/systemd/system/aws_signing_helper_update.service
Sysctl-Konfigurationsdatei	/etc/sysctl.d/99-nodeadm.conf
Ca-certificates	/etc/ssl/certs/ca-zertifikate.crt
GPG-Schlüsseldatei	/etc/apt/keyrings/docker.asc
Docker-Repo-Quelldatei	/.liste etc/apt/sources.list.d/docker

Knotenkonfiguration für SSM-Hybridaktivierungen

Im Folgenden finden Sie ein Beispiel für die `nodeConfig.yaml` Verwendung von AWS SSM-Hybrid-Aktivierungen für Anmeldeinformationen für Hybridknoten.

```
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name:             # Name of the EKS cluster
    region:           # AWS Region where the EKS cluster resides
  hybrid:
    ssm:
      activationCode: # SSM hybrid activation code
      activationId:   # SSM hybrid activation id
```

Knotenkonfiguration für IAM Roles Anywhere

Im Folgenden finden Sie ein Beispiel `nodeConfig.yaml` für AWS IAM Roles Anywhere für Anmeldeinformationen für Hybridknoten.

Wenn `nodeName` Sie AWS IAM Roles Anywhere als Ihren lokalen Anbieter für Anmeldeinformationen verwenden, müssen die in Ihrer `nodeadm` Konfiguration verwendeten Berechtigungen mit den Berechtigungen übereinstimmen, die Sie für Ihre IAM-Rolle für Hybrid Nodes festgelegt haben. Wenn Ihre Berechtigungen für die IAM-Rolle Hybrid Nodes beispielsweise nur zulassen, dass AWS IAM Roles Anywhere die Rolle übernimmt, wenn der Name der Rollensitzung dem CN des Host-Zertifikats entspricht, muss der `nodeName` in Ihrer `nodeadm` Konfiguration dem CN Ihrer Zertifikate entsprechen. Der von Ihnen verwendete `nodeName` kann nicht länger als 64 Zeichen sein. Weitere Informationen finden Sie unter the section called “Anmeldeinformationen vorbereiten”.

```
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name:               # Name of the EKS cluster
    region:             # AWS Region where the EKS cluster resides
  hybrid:
    iamRolesAnywhere:
      nodeName:         # Name of the node
      trustAnchorArn:   # ARN of the IAM Roles Anywhere trust anchor
      profileArn:       # ARN of the IAM Roles Anywhere profile
      roleArn:          # ARN of the Hybrid Nodes IAM role
      certificatePath: # Path to the certificate file to authenticate with the IAM
 Roles Anywhere trust anchor
      privateKeyPath:   # Path to the private key file for the certificate
```

Knotenkonfiguration zur Anpassung von kubelet (optional)

Sie können die kubelet-Konfiguration und Flags in Ihrer `nodeadm`-Konfiguration übergeben. Im folgenden Beispiel erfahren Sie, wie Sie eine zusätzliche Knotenbezeichnung `abc.amazonaws.com/test-label` und Konfiguration hinzufügen, um `shutdownGracePeriod` auf 30 Sekunden festzulegen.

```
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name:             # Name of the EKS cluster
    region:           # AWS Region where the EKS cluster resides
  kubelet:
    config:           # Map of kubelet config and values
       shutdownGracePeriod: 30s
    flags:            # List of kubelet flags
       - --node-labels=abc.company.com/test-label=true
  hybrid:
    ssm:
      activationCode: # SSM hybrid activation code
      activationId:   # SSM hybrid activation id
```

Knotenkonfiguration zur Anpassung von containerd (optional)

Sie können eine benutzerdefinierte containerd-Konfiguration in Ihrer `nodeadm`-Konfiguration übergeben. Die containerd-Konfiguration für `nodeadm` akzeptiert Inline-TOML. Im folgenden Beispiel erfahren Sie, wie Sie containerd konfigurieren, um das Löschen entpackter Image-Ebenen im containerd-Inhaltsspeicher zu deaktivieren.

```
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name:             # Name of the EKS cluster
    region:           # AWS Region where the EKS cluster resides
  containerd:
    config: |         # Inline TOML containerd additional configuration
       [plugins."io.containerd.grpc.v1.cri".containerd]
       discard_unpacked_layers = false
  hybrid:
    ssm:
```

```
    activationCode: # SSM hybrid activation code
    activationId:   # SSM hybrid activation id
```

> **Note**
>
> Die Containerversionen 1.x und 2.x verwenden unterschiedliche Konfigurationsformate. containerd 1.x verwendet die Konfigurationsversion 2, während Containerd 2.x die Konfigurationsversion 3 verwendet. Obwohl containerd 2.x weiterhin abwärtskompatibel mit Config Version 2 ist, wird Config Version 3 für eine optimale Leistung empfohlen. Überprüfen Sie Ihre Container-Version mit `containerd --version` oder überprüfen `nodeadm` Sie die Installationsprotokolle. Weitere Informationen zur Versionierung von Konfigurationen finden Sie unter https://containerd.io/releases/

Sie können auch die Containerd-Konfiguration verwenden, um die SELinux Unterstützung zu aktivieren. SELinux Wenn diese Option auf containerd aktiviert ist, stellen Sie sicher, dass Pods, die auf dem Knoten geplant sind, den richtigen SecurityContext haben und seLinuxOptions aktiviert sind. Weitere Informationen für die Konfiguration eines Sicherheitskontexts finden Sie in der Kubernetes-Dokumentation.

> **Note**
>
> Red Hat Enterprise Linux (RHEL) 8 und RHEL 9 sind standardmäßig SELinux aktiviert und auf dem Host auf Strict gesetzt. Amazon Linux 2023 ist standardmäßig SELinux aktiviert und auf den permissiven Modus eingestellt. Wenn auf dem Host auf den permissiven Modus gesetzt SELinux ist, blockiert die Aktivierung auf containerd keine Anfragen, sondern protokolliert sie entsprechend der SELinux Konfiguration auf dem Host.

```
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name:             # Name of the EKS cluster
    region:           # AWS Region where the EKS cluster resides
  containerd:
    config: |         # Inline TOML containerd additional configuration
       [plugins."io.containerd.grpc.v1.cri"]
       enable_selinux = true
```

```
  hybrid:
    ssm:
      activationCode: # SSM hybrid activation code
      activationId:   # SSM hybrid activation id
```

Fehlerbehebung bei Hybridknoten

In diesem Thema werden einige gängige Fehler behandelt, die bei der Verwendung von Amazon EKS Hybrid Nodes auftreten können, sowie deren Behebung. Weitere Informationen zur Fehlerbehebung finden Sie unter Fehlersuche und dem Knowledge Center-Tag für Amazon EKS auf AWS re:POST. Wenn Sie das Problem nicht lösen können, wenden Sie sich an den AWS Support.

Knoten-Fehlerbehebung mit **`nodeadm debug`** Sie können den `nodeadm debug`-Befehl von Ihren Hybridknoten aus ausführen, um zu überprüfen, ob die Netzwerk- und Anforderungen an den Anmeldeinformationen erfüllt sind. Weitere Informationen zum `nodeadm debug`-Befehl finden Sie unter the section called “nodeadm für Hybridknoten”.

Probleme mit Ihren Hybridknoten mithilfe von Cluster-Erkenntnissen erkennen Amazon EKS Cluster Insights umfasst Erkenntnisprüfungen, die gängige Probleme mit der Konfiguration von EKS-Hybridknoten in Ihrem Cluster erkennen. Sie können die Ergebnisse aller Insight-Checks über die AWS-Managementkonsole, AWS CLI und die anzeigen AWS SDKs. Weitere Informationen über Cluster-Erkenntnisse finden Sie unter the section called “Cluster-Einblicke”.

Fehlerbehebung bei der Installation von Hybridknoten

Die folgenden Themen zur Fehlerbehebung beziehen sich auf die Installation der Abhängigkeiten der Hybridknoten auf Hosts mit dem `nodeadm install`-Befehl.

`nodeadm`-Befehl fehlgeschlagen `must run as root`

Der `nodeadm install`-Befehl muss mit einem Benutzer ausgeführt werden, der über Root- oder `sudo`-Berechtigungen auf Ihrem Host verfügt. Wenn Sie `nodeadm install` mit einem Benutzer ausführen, der nicht über Root- oder `sudo`-Berechtigungen verfügt, wird in der `nodeadm`-Ausgabe der folgende Fehler angezeigt.

```
"msg":"Command failed","error":"must run as root"
```

Verbindung zu Abhängigkeiten nicht möglich

Der Befehl `nodeadm install` installiert die für Hybridknoten erforderlichen Abhängigkeiten. Zu den Abhängigkeiten der Hybridknoten gehören`containerd`, `kubelet``kubectl`, und AWS SSM- oder

AWS IAM Roles Anywhere-Komponenten. Sie müssen von Ihrem Ausführungsstandort von `nodeadm install` aus Zugriff haben, um diese Abhängigkeiten herunterzuladen. Weitere Informationen zur Liste der Speicherorte, auf die Sie zugreifen können müssen, finden Sie unter the section called "Netzwerk vorbereiten". Wenn Sie keinen Zugriff haben, werden in der `nodeadm install`-Ausgabe Fehlermeldungen ähnlich den folgenden angezeigt.

```
"msg":"Command failed","error":"failed reading file from url: ...: max retries achieved
 for http request"
```

Paket-Manager konnte nicht aktualisiert werden

Der Befehl `nodeadm install` führt `apt update` oder `yum update` oder `dnf update` aus, bevor die Abhängigkeiten der Hybridknoten installiert werden. Wenn dieser Schritt nicht erfolgreich ist, werden möglicherweise Fehler ähnlich den folgenden angezeigt. Zur Behebung können Sie `apt update` oder `yum update` oder `dnf update` ausführen, bevor Sie `nodeadm install` ausführen, oder Sie können versuchen, `nodeadm install` erneut auszuführen.

```
failed to run update using package manager
```

Zeitüberschreitung oder Kontext-Frist überschritten

Wenn beim Ausführen von `nodeadm install` in verschiedenen Phasen des Installationsprozesses Probleme mit Fehlern auftreten, die auf eine Zeitüberschreitung oder eine überschrittene Kontextfrist hinweisen, liegt dies möglicherweise an einer langsamen Verbindung, welche die Installation der Abhängigkeiten der Hybridknoten vor Ablauf der Zeitüberschreitungen verhindert. Um diese Probleme zu umgehen, können Sie versuchen, das `--timeout`-Flag in `nodeadm` zu verwenden, um die Dauer der Zeitüberschreitungen für das Herunterladen der Abhängigkeiten zu verlängern.

```
nodeadm install K8S_VERSION --credential-provider CREDS_PROVIDER --timeout 20m0s
```

Fehlerbehebung beim Verbinden von Hybridknoten

Die Themen zur Fehlerbehebung in diesem Abschnitt beziehen sich auf den Prozess der Verbindung von Hybridknoten mit EKS-Clustern mithilfe des `nodeadm init`-Befehls.

Betriebsfehler oder nicht unterstütztes Schema

Wenn Sie bei der Ausführung von `nodeadm init` Fehler im Zusammenhang mit `operation error` oder `unsupported scheme` entdecken, überprüfen Sie Ihr `nodeConfig.yaml`, um

sicherzustellen, dass es richtig formatiert und an `nodeadm` übergeben wird. Weitere Informationen zum Format und den Optionen für `nodeConfig.yaml` finden Sie unter the section called “nodeadm für Hybridknoten”.

```
"msg":"Command failed","error":"operation error ec2imds: GetRegion, request canceled,
 context deadline exceeded"
```

Für die IAM-Rolle für Hybridknoten fehlen Berechtigungen für die **`eks:DescribeCluster`**-Aktion

`nodeadm`Versucht bei der Ausführung`nodeadm init`, Informationen über Ihren EKS-Cluster zu sammeln, indem die `DescribeCluster` EKS-Aktion aufgerufen wird. Wenn Ihre IAM-Rolle Hybrid Nodes nicht über die Berechtigung für die `eks:DescribeCluster` Aktion verfügt, müssen Sie Ihren Kubernetes-API-Endpunkt, das Cluster-CA-Bundle und den IPv4 Service-CIDR in der Knotenkonfiguration übergeben, an die Sie bei der Ausführung übergeben`nodeadm`. `nodeadm init` Weitere Informationen zu den erforderlichen Berechtigungen für die IAM-Rolle für Hybridknoten finden Sie unter the section called “Anmeldeinformationen vorbereiten”.

```
"msg":"Command failed","error":"operation error EKS: DescribeCluster, https response
 error StatusCode: 403 ... AccessDeniedException"
```

Für die IAM-Rolle für Hybridknoten fehlen Berechtigungen für die **`eks:ListAccessEntries`**-Aktion

`nodeadm`Versucht bei der Ausführung zu überprüfen`nodeadm init`, ob Ihr EKS-Cluster über einen Zugriffseintrag des Typs verfügt, der mit der IAM-Rolle Hybrid Nodes `HYBRID_LINUX` verknüpft ist, indem die EKS-Aktion aufgerufen wird. `ListAccessEntries` Wenn Ihre Hybrid Nodes IAM-Rolle nicht über die Berechtigung für die `eks:ListAccessEntries` Aktion verfügt, müssen Sie das `--skip cluster-access-validation` Flag übergeben, wenn Sie den Befehl ausführen. `nodeadm init` Weitere Informationen zu den erforderlichen Berechtigungen für die IAM-Rolle für Hybridknoten finden Sie unter the section called “Anmeldeinformationen vorbereiten”.

```
"msg":"Command failed","error":"operation error EKS: ListAccessEntries, https response
 error StatusCode: 403 ... AccessDeniedException"
```

Knoten-IP nicht im CIDR des Fern-Knotennetzwerks

Bei der Ausführung kann ein Fehler auftreten`nodeadm init`, wenn sich die IP-Adresse des Knotens nicht im angegebenen Remote-Knoten-Netzwerk CIDRs befindet. Der Fehler sieht in etwa wie im folgenden Beispiel aus:

```
node IP 10.18.0.1 is not in any of the remote network CIDR blocks [10.0.0.0/16
 192.168.0.0/16]
```

Dieses Beispiel zeigt einen Knoten mit IP 10.18.0.1, der versucht, einem Cluster mit dem Remote-Netzwerk CIDRs 10.0.0.0/16 und 192.168.0.0/16 beizutreten. Der Fehler tritt auf, weil 10.18.0.1 nicht in einem der Bereiche liegt.

Bestätigen Sie, dass Sie Ihr `RemoteNodeNetworks` richtig konfiguriert haben, um alle Knoten-IP-Adressen einzuschließen. Weitere Informationen zur Netzwerkkonfiguration finden Sie unter the section called “Netzwerk vorbereiten”.

- Führen Sie den folgenden Befehl in der Region aus, in der sich Ihr Cluster befindet, um Ihre `RemoteNodeNetwork`-Konfigurationen zu überprüfen. Überprüfen Sie, ob die in der Ausgabe aufgeführten CIDR-Blöcke den IP-Bereich Ihres Knotens enthalten und mit den in der Fehlermeldung aufgeführten CIDR-Blöcken übereinstimmen. Wenn sie nicht übereinstimmen, stellen Sie sicher, dass der Cluster-Name und die Region in Ihrem `nodeConfig.yaml` mit dem gewünschten Cluster übereinstimmen.

```
aws eks describe-cluster --name CLUSTER_NAME --region REGION_NAME --query
 cluster.remoteNetworkConfig.remoteNodeNetworks
```

- Überprüfen Sie, ob Sie mit dem gewünschten Knoten arbeiten:
  - Bestätigen Sie, dass Sie sich auf dem richtigen Knoten befinden, indem Sie prüfen, ob dessen Hostname und IP-Adresse mit denen übereinstimmen, die Sie beim Cluster registrieren möchten.
  - Bestätigen Sie, dass sich dieser Knoten im korrekten On-Premises-Netzwerk befindet (das Netzwerk, dessen CIDR-Bereich während der Cluster-Einrichtung als `RemoteNodeNetwork` registriert wurde).

Wenn Ihre Knoten-IP immer noch nicht Ihren Erwartungen entspricht, überprüfen Sie Folgendes:

- Wenn Sie IAM Roles Anywhere verwenden, führt `kubelet` eine DNS-Abfrage für IAM Roles Anywhere `nodeName` durch und verwenden Sie eine mit dem Knotennamen verknüpfte IP-Adresse, sofern verfügbar. Wenn Sie DNS-Einträge für Ihre Knoten verwalten, stellen Sie sicher, dass diese Einträge auf Ihr Remote-Knotennetzwerk verweisen. IPs CIDRs
- Wenn Ihr Knoten über mehrere Netzwerkschnittstellen verfügt, wählen Sie `kubelet` möglicherweise standardmäßig eine Schnittstelle mit einer IP-Adresse außerhalb Ihres Remote-

Knoten-Netzwerks CIDRs aus. Um eine andere Schnittstelle zu verwenden, geben Sie deren IP-Adresse mithilfe des Flags `--node-ip` `kubelet` in Ihrem `nodeConfig.yaml` an. Weitere Informationen finden Sie unter the section called “nodeadm für Hybridknoten”. Sie können die Netzwerkschnittstellen und IP-Adressen Ihres Knotens anzeigen, indem Sie den folgenden Befehl auf Ihrem Knoten ausführen:

```
ip addr show
```

Hybridknoten werden im EKS-Cluster nicht angezeigt

Wenn Sie `nodeadm init` ausgeführt haben und es abgeschlossen wurde, Ihre Hybridknoten jedoch nicht in Ihrem Cluster angezeigt werden, liegen möglicherweise Probleme mit der Netzwerkverbindung zwischen Ihren Hybridknoten und der EKS-Steuerebene vor. Möglicherweise haben Sie die erforderlichen Sicherheitsgruppenberechtigungen nicht konfiguriert oder Sie verfügen möglicherweise nicht über die erforderliche Zuordnung Ihrer IAM-Rolle für Hybridknoten zur rollenbasierten Zugriffskontrolle (RBAC) von Kubernetes. Sie können den Debugging-Prozess starten, indem Sie den Status von `kubelet` und die `kubelet`-Protokolle mit den folgenden Befehlen überprüfen. Führen Sie die folgenden Befehle von den Hybridknoten aus, die Ihrem Cluster nicht beitreten konnten.

```
systemctl status kubelet
```

```
journalctl -u kubelet -f
```

Kommunikation mit Cluster nicht möglich

Wenn Ihr Hybridknoten nicht mit der Cluster-Steuerebene kommunizieren konnte, werden möglicherweise Protokolle ähnlich den folgenden angezeigt.

```
"Failed to ensure lease exists, will retry" err="Get ..."
```

```
"Unable to register node with API server" err="Post ..."
```

```
Failed to contact API server when waiting for CSINode publishing ... dial tcp <ip
 address>: i/o timeout
```

Wenn diese Meldungen angezeigt werden, überprüfen Sie Folgendes, um sicherzustellen, dass die in the section called “Netzwerk vorbereiten” aufgeführten Anforderungen für Hybridknoten erfüllt werden.

- Vergewissern Sie sich, dass die an den EKS-Cluster übergebene VPC Routen zu Ihrem Transit Gateway (TGW) oder Virtual Private Gateway (VGW) für Ihren lokalen Knoten und optional Ihren Pod hat. CIDRs
- Vergewissern Sie sich, dass Sie über eine zusätzliche Sicherheitsgruppe für Ihren EKS-Cluster verfügen, die Regeln für eingehenden Datenverkehr für Ihren lokalen Knoten und optional für den Pod enthält. CIDRs CIDRs
- Bestätigen Sie, dass Ihr On-Premises-Router so konfiguriert ist, dass er den Datenverkehr zur und von der EKS-Steuerebene zulässt.

Nicht autorisiert

Wenn Ihr Hybridknoten mit der EKS-Steuersebene kommunizieren konnte, sich jedoch nicht registrieren konnte, werden möglicherweise Protokolle angezeigt, die den folgenden ähneln. Beachten Sie, dass der wesentliche Unterschied in den folgenden Protokollmeldungen der `Unauthorized`-Fehler ist. Dies weist darauf hin, dass der Knoten seine Aufgaben nicht ausführen konnte, da er nicht über die erforderlichen Berechtigungen verfügt.

```
"Failed to ensure lease exists, will retry" err="Unauthorized"
```

```
"Unable to register node with API server" err="Unauthorized"
```

```
Failed to contact API server when waiting for CSINode publishing: Unauthorized
```

Wenn Sie diese Meldungen sehen, überprüfen Sie Folgendes, um sicherzustellen, dass die Anforderungen für Hybridknoten in the section called “Anmeldeinformationen vorbereiten” und the section called “Cluster-Zugriff vorbereiten” erfüllt werden.

- Bestätigen Sie, dass die Identität der Hybridknoten mit Ihrer erwarteten IAM-Rolle für Hybridknoten übereinstimmt. Dies kann durch Ausführen von `sudo aws sts get-caller-identity` von Ihren Hybridknoten aus erfolgen.
- Bestätigen Sie, dass Ihre IAM-Rolle für Hybridknoten über die erforderlichen Berechtigungen verfügt.

- Vergewissern Sie sich, dass Sie in Ihrem Cluster einen EKS-Zugriffseintrag für Ihre Hybrid Nodes IAM-Rolle haben, oder vergewissern Sie sich, dass Sie `aws-auth` ConfigMap einen Eintrag für Ihre Hybrid Nodes IAM-Rolle haben. Wenn Sie EKS-Zugriffseinträge verwenden, bestätigen Sie, dass Ihr `HYBRID_LINUX`-Zugriffseintrag für Ihre IAM-Rolle für Hybridknoten den Zugriffstyp hat. Wenn Sie die verwenden, stellen Sie sicher `aws-auth` ConfigMap, dass Ihr Eintrag für die IAM-Rolle Hybrid Nodes die unter beschriebenen Anforderungen und Formatierungen erfüllt. the section called “Cluster-Zugriff vorbereiten”

Hybridknoten sind beim EKS-Cluster registriert, zeigen aber den Status **`Not Ready`** an

Wenn Ihre Hybridknoten erfolgreich bei Ihrem EKS-Cluster registriert wurden, die Hybridknoten jedoch den Status `Not Ready` anzeigen, müssen Sie zunächst den Status Ihrer Container Networking Interface (CNI) überprüfen. Wenn Sie kein CNI installiert haben, wird erwartet, dass Ihre Hybridknoten den Status `Not Ready` haben. Sobald ein CNI installiert ist und erfolgreich ausgeführt wird, werden die Knoten auf den Status `Ready` aktualisiert. Wenn Sie versucht haben, ein CNI zu installieren, es aber nicht erfolgreich ausgeführt wird, finden Sie weitere Informationen unter the section called “CNI-Fehlerbehebung bei Hybridknoten” auf dieser Seite.

Anfragen zum Signieren von Zertifikaten (CSRs) bleiben hängen. Ausstehend

Wenn Sie nach dem Verbinden der Hybridknoten mit Ihrem EKS-Cluster feststellen, dass noch ausstehende Knoten CSRs für Ihre Hybridknoten vorliegen, erfüllen Ihre Hybridknoten nicht die Anforderungen für die automatische Genehmigung. CSRs für Hybridknoten werden automatisch genehmigt und ausgestellt, wenn die CSRs vier Hybridknoten von einem Knoten mit `eks.amazonaws.com/compute-type: hybrid` Bezeichnung erstellt wurden und der CSR die folgenden alternativen Subject Names (SANs) hat: mindestens ein DNS-SAN, das dem Knotennamen und der IP entspricht, SANs gehört zum Netzwerk CIDRs des Remote-Knotens.

Hybridprofil ist bereits vorhanden

Wenn Sie Ihre `nodeadm`-Konfiguration geändert haben und versuchen, den Knoten mit der neuen Konfiguration erneut zu registrieren, wird möglicherweise ein Fehler angezeigt, der besagt, dass das Hybrid-Profil bereits vorhanden ist, sich sein Inhalt jedoch geändert hat. Anstatt zwischen Konfigurationsänderungen `nodeadm init` auszuführen, führen Sie `nodeadm uninstall` gefolgt von `nodeadm install` und `nodeadm init` aus. Dadurch wird eine ordnungsgemäße Bereinigung der Konfigurationsänderungen sichergestellt.

```
"msg":"Command failed","error":"hybrid profile already exists at /etc/aws/hybrid/config
 but its contents do not align with the expected configuration"
```

Hybridknoten konnte die private API nicht auflösen

Wenn nach dem Ausführen von `nodeadm init` in den `kubelet`-Protokollen ein Fehler angezeigt wird, der auf Fehler beim Kontaktieren des EKS Kubernetes-API-Servers aufgrund von `no such host` hinweist, müssen Sie möglicherweise Ihren DNS-Eintrag für den EKS Kubernetes-API-Endpunkt in Ihrem On-Premises-Netzwerk oder auf Host-Ebene ändern. Weitere Informationen finden Sie in der AWS Route53-Dokumentation unter Weiterleiten eingehender DNS-Abfragen an Ihre VPC.

```
Failed to contact API server when waiting for CSINode publishing: Get ... no such host
```

Hybridknoten können in der EKS-Konsole nicht angezeigt werden

Wenn Sie Ihre Hybridknoten registriert haben, diese aber nicht in der EKS-Konsole anzeigen können, überprüfen Sie die Berechtigungen des IAM-Prinzipals, das Sie zum Anzeigen der Konsole verwenden. Der von Ihnen verwendete IAM-Prinzipal muss über bestimmte Mindestberechtigungen für IAM und Kubernetes verfügen, um Ressourcen in der Konsole anzuzeigen. Weitere Informationen finden Sie unter the section called “Zugriff auf Cluster-Ressourcen”.

Fehlerbehebung beim Ausführen von Hybridknoten

Wenn Ihre Hybridknoten bei Ihrem EKS-Cluster registriert waren, den Status `Ready` hatten und dann in den Status `Not Ready` übergingen, gibt es eine Vielzahl von Problemen, die zu diesem fehlerhaften Status geführt haben könnten, z. B. der Mangel an ausreichenden Ressourcen für CPU, Arbeitsspeicher oder verfügbaren Speicherplatz auf dem Knoten oder die Trennung des Knotens von der EKS-Steuerebene. Sie können die folgenden Schritte verwenden, um Probleme mit Ihren Knoten zu beheben. Wenn Sie das Problem nicht lösen können, wenden Sie sich an den AWS Support.

Führen Sie `nodeadm debug` von Ihren Hybridknoten aus, um zu überprüfen, ob die Netzwerk- und Anmeldeinformationen erfüllt sind. Weitere Informationen zum `nodeadm debug`-Befehl finden Sie unter the section called “nodeadm für Hybridknoten”.

Knotenstatus abrufen

```
kubectl get nodes -o wide
```

Knoten-Bedingungen und -ereignisse prüfen

```
kubectl describe node NODE_NAME
```

Pod-Status abrufen

```
kubectl get pods -A -o wide
```

Pod-Bedingungen und -ereignisse prüfen

```
kubectl describe pod POD_NAME
```

Pod-Protokolle prüfen

```
kubectl logs POD_NAME
```

`kubectl`-Protokolle überprüfen

```
systemctl status kubelet
```

```
journalctl -u kubelet -f
```

Pod-Aktivitätsprüfungen schlagen fehl oder Webhooks funktionieren nicht

Wenn Anwendungen, Add-Ons oder Webhooks, die in Ihren Hybridknoten ausgeführt werden, nicht ordnungsgemäß starten, liegen möglicherweise Netzwerkprobleme vor, welche die Kommunikation mit den Pods blockieren. Damit die EKS-Steuerebene auf Hybridknoten ausgeführten Webhooks kontaktieren kann, müssen Sie Ihren EKS-Cluster mit einem Fern-Pod-Netzwerk konfigurieren und Routen für Ihr On-Premises-Pod-CIDR in Ihrer VPC-Routing-Tabelle haben, mit dem Ziel als Ihr Transit Gateway (TGW), Virtual Private Gateway (VPW) oder anderes Gateway, das Sie zum Verbinden Ihres VPC mit Ihrem On-Premises-Netzwerk verwenden. Weitere Informationen zu den Netzwerkanforderungen für Hybridknoten finden Sie unter the section called “Netzwerk vorbereiten”. Sie müssen diesen Datenverkehr zusätzlich in Ihrer On-Premises-Firewall zulassen und sicherstellen, dass Ihr Router ordnungsgemäß an Ihre Pods weiterleiten kann. Weitere Informationen zu den

Anforderungen für die Ausführung von Webhooks auf Hybridknoten finden Sie unter the section called “Webhooks konfigurieren”.

Eine allgemeine Pod-Protokollnachricht für dieses Szenario wird unten angezeigt, wobei „ip-address“ die Cluster-IP für den Kubernetes-Service ist.

```
dial tcp <ip-address>:443: connect: no route to host
```

`kubectl logs`oder **`kubectl exec`** Befehle, die nicht funktionieren (**`kubelet`**API-Befehle)

Wenn bei Befehlen `kubectl attach` `kubectl cp`kubectl exec,`kubectl logs`, und ein `kubectl port-forward` Timeout auftritt, während andere `kubectl` Befehle erfolgreich sind, hängt das Problem wahrscheinlich mit der Remote-Netzwerkkonfiguration zusammen. Diese Befehle stellen über den Cluster eine Verbindung zum `kubelet`-Endpunkt auf dem Knoten her. Weitere Informationen finden Sie unter the section called “`kubelet`-Endpunkt”.

Stellen Sie sicher, dass Ihr Node IPs und Ihr Pod in das für Ihren Cluster CIDRs konfigurierte Remote-Knoten-Netzwerk und das Remote-Pod-Netzwerk IPs fallen. Verwenden Sie die folgenden Befehle, um die IP-Zuweisungen zu überprüfen.

```
kubectl get nodes -o wide
```

```
kubectl get pods -A -o wide
```

Vergleichen Sie diese IPs mit Ihrem konfigurierten Remote-Netzwerk CIDRs , um sicherzustellen, dass das Routing korrekt ist. Die Anforderungen für die Netzwerkkonfiguration finden Sie unter the section called “Netzwerk vorbereiten”.

CNI-Fehlerbehebung bei Hybridknoten

Wenn beim ersten Starten von Cilium oder Calico mit Hybridknoten Probleme auftreten, liegt dies meistens an Netzwerkproblemen zwischen Hybridknoten oder den auf Hybridknoten ausgeführten CNI-Pods und der EKS-Steuerebene. Stellen Sie sicher, dass Ihre Umgebung die Anforderungen unter „Netzwerk für Hybridknoten vorbereiten“ erfüllt. Es ist sinnvoll, das Problem in Teilbereiche zu unterteilen.

EKS-Cluster-Konfiguration

- Sind die RemotePodNetwork Konfigurationen RemoteNodeNetwork und korrekt?

VPC-Konfiguration

Gibt es Routen für das RemoteNodeNetwork und RemotePodNetwork in der VPC-Routingtabelle mit dem Ziel des Transit Gateway oder Virtual Private Gateways?

Sicherheitsgruppenkonfiguration

Gibt es Regeln für eingehenden und ausgehenden Datenverkehr? RemoteNodeNetwork RemotePodNetwork

On-Premises-Netzwerk

Gibt es Routen und Zugriff von und zur EKS-Steuerebene sowie von und zu den Hybridknoten und den in Hybridknoten ausgeführten Pods?

CNI-Konfiguration

Wenn Sie ein Overlay-Netzwerk verwenden, stimmt die IP-Pool-Konfiguration für das CNI mit der für den EKS-Cluster RemotePodNetwork konfigurierten überein, wenn Webhooks verwendet werden?

Der Hybridknoten hat den Status **`Ready`** ohne installiertes CNI

Wenn Ihre Hybridknoten den Status `Ready` anzeigen, Sie aber kein CNI auf Ihrem Cluster installiert haben, ist es möglich, dass sich auf Ihren Hybridknoten alte CNI-Artefakte befinden. Wenn Sie Cilium und Calico mit Tools wie Helm deinstallieren, werden die Ressourcen auf der Festplatte standardmäßig nicht von Ihren physischen oder virtuellen Rechnern entfernt. Darüber hinaus sind die benutzerdefinierten Ressourcendefinitionen (CRDs) für diese CNIs möglicherweise noch aus einer alten Installation auf Ihrem Cluster vorhanden. Weitere Informationen finden Sie in den Abschnitten „Cilium löschen“ und „Calico löschen“ unter the section called “CNI konfigurieren”.

Fehlerbehebung bei Cilium

Wenn Sie Probleme bei der Ausführung von Cilium in Hybridknoten haben, lesen Sie die Schritte zur Fehlerbehebung in der Cilium-Dokumentation. Die folgenden Abschnitte behandeln Probleme, die speziell bei der Bereitstellung von Cilium in Hybridknoten auftreten können.

Cilium startet nicht

Wenn die auf jedem Hybridknoten ausgeführten Cilium-Agenten nicht starten, überprüfen Sie die Protokolle der Cilium-Agent-Pods auf Fehler. Der Cilium-Agent benötigt eine Verbindung zum EKS-Kubernetes-API-Endpunkt, um zu starten. Der Startup des Cilium-Agenten schlägt fehl, wenn diese

Konnektivität nicht richtig konfiguriert ist. In diesem Fall werden in den Pod-Protokollen des Cilium-Agenten Protokollmeldungen ähnlich den folgenden angezeigt.

```
msg="Unable to contact k8s api-server"
level=fatal msg="failed to start: Get \"https://<k8s-cluster-ip>:443/api/v1/namespaces/
kube-system\": dial tcp <k8s-cluster-ip>:443: i/o timeout"
```

Der Cilium-Agent wird im Host-Netzwerk ausgeführt. Ihr EKS-Cluster muss für die Cilium-Konnektivität mit `RemoteNodeNetwork` konfiguriert sein. Bestätigen Sie, dass Sie über eine zusätzliche Sicherheitsgruppe für Ihren EKS-Cluster mit einer eingehenden Regel für Ihr `RemoteNodeNetwork` verfügen, dass Sie Routen in Ihrem VPC für Ihr `RemoteNodeNetwork` haben und dass Ihr On-Premises-Netzwerk richtig konfiguriert ist, um die Verbindung zur EKS-Steuerebene zu ermöglichen.

Wenn der Cilium-Operator läuft und einige Ihrer Cilium-Agenten laufen, aber nicht alle, stellen Sie sicher, dass Sie über einen verfügbaren Pod verfügen, IPs den Sie allen Knoten in Ihrem Cluster zuweisen können. Sie konfigurieren die Größe Ihres zuweisbaren Pods, CIDRs wenn Sie den Clusterpool IPAM mit in Ihrer Cilium-Konfiguration verwenden. `clusterPoolIPv4PodCIDRList` Die CIDR-Größe pro Knoten wird mit der Einstellung `clusterPoolIPv4MaskSize` in Ihrer Cilium-Konfiguration konfiguriert. Weitere Informationen finden Sie unter Erweiterung des Cluster-Pools in der Cilium-Dokumentation.

Cilium BGP funktioniert nicht

Wenn Sie die Cilium BGP-Steuerebene verwenden, um Ihre Pod- oder Service-Adressen in Ihrem On-Premises-Netzwerk bekannt zu geben, können Sie die folgenden Cilium-CLI-Befehle verwenden, um zu überprüfen, ob BGP die Routen zu Ihren Ressourcen bekannt gibt. Die Schritte zur Installation der Cilium-CLI finden Sie unter Installation der Cilium-CLI in der Cilium-Dokumentation.

Wenn BGP ordnungsgemäß funktioniert, sollten Ihre Hybridknoten den Sitzungsstatus `established` in der Ausgabe haben. Möglicherweise müssen Sie mit Ihrem Netzwerk-Team zusammenarbeiten, um die korrekten Werte für die lokalen AS, Peer-AS und Peer-Adressen Ihrer Umgebung zu ermitteln.

```
cilium bgp peers
```

```
cilium bgp routes
```

Wenn Sie Cilium BGP verwenden, um die IPs Dienste mit Typ zu bewerben`LoadBalancer`, müssen Sie sowohl auf Ihrem Dienst als auch auf Ihrem Dienst dieselbe Bezeichnung haben, die in Ihrem

`CiliumLoadBalancerIPPool` Selektor verwendet werden sollte. `CiliumBGPAdvertisement` Es folgt ein Beispiel. Beachten Sie, dass, wenn Sie Cilium BGP verwenden, um die Dienste mit Typ zu bewerben LoadBalancer, die BGP-Routen während IPs des Neustarts des Cilium-Agenten unterbrochen werden können. Weitere Informationen finden Sie unter Fehlerszenarien in der Cilium-Dokumentation.

Service

```
kind: Service
apiVersion: v1
metadata:
  name: guestbook
  labels:
    app: guestbook
spec:
  ports:
  - port: 3000
    targetPort: http-server
  selector:
    app: guestbook
  type: LoadBalancer
```

CiliumLoadBalancerIPPool

```
apiVersion: cilium.io/v2alpha1
kind: CiliumLoadBalancerIPPool
metadata:
  name: guestbook-pool
  labels:
    app: guestbook
spec:
  blocks:
  - cidr: <CIDR to advertise>
  serviceSelector:
    matchExpressions:
      - { key: app, operator: In, values: [ guestbook ] }
```

CiliumBGPAdvertisement

```
apiVersion: cilium.io/v2alpha1
kind: CiliumBGPAdvertisement
```

```
metadata:
  name: bgp-advertisements-guestbook
  labels:
    advertise: bgp
spec:
  advertisements:
    - advertisementType: "Service"
      service:
        addresses:
          - ExternalIP
          - LoadBalancerIP
      selector:
        matchExpressions:
          - { key: app, operator: In, values: [ guestbook ] }
```

Calico-Fehlerbehebung

Wenn Sie Probleme bei der Ausführung von Calico in Hybridknoten haben, lesen Sie die Schritte zur Fehlerbehebung in der Calico-Dokumentation. Die nachfolgenden Abschnitte behandeln Probleme, die speziell bei der Bereitstellung von Calico in Hybridknoten auftreten können.

Die nachfolgende Tabelle fasst die Calico-Komponenten zusammen und gibt an, ob sie standardmäßig im Knoten- oder Pod-Netzwerk ausgeführt werden. Wenn Sie Calico für die Verwendung von NAT für ausgehenden Pod-Datenverkehr konfiguriert haben, muss Ihr On-Premises-Netzwerk so konfiguriert sein, dass der Datenverkehr an die CIDR Ihres On-Premises-Knotens weitergeleitet wird. Außerdem müssen Ihre VPC-Routing-Tabellen mit einer Route für die CIDR Ihres On-Premises-Knotens konfiguriert sein, wobei Ihr Transit-Gateway (TGW) oder Ihr Virtual Private Gateway (VGW) als Ziel angegeben ist. Wenn Sie Calico nicht für die Verwendung von NAT für ausgehenden Pod-Datenverkehr konfigurieren, muss Ihr On-Premises-Netzwerk so konfiguriert sein, dass der Datenverkehr an Ihre On-Premises-Pod-CIDR weitergeleitet wird. Außerdem müssen Ihre VPC-Routing-Tabellen mit einer Route für Ihre On-Premises-Pod-CIDR konfiguriert sein, wobei Ihr Transit-Gateway (TGW) oder Ihr Virtual Private Gateway (VGW) als Ziel angegeben ist.

Komponente	Netzwerk
Calico-API-Server	Knoten
Calico-Controller für Kubernetes	Pod
Calico-Knoten-Agent	Knoten

Komponente	Netzwerk
Calico `typha`	Knoten
Calico-CSI-Knoten-Treiber	Schote
Calico-Operator	Knoten

Calico-Ressourcen sind geplant oder werden in abgesperrten Knoten ausgeführt

Die Calico-Ressourcen, die nicht als A ausgeführt werden, DaemonSet verfügen standardmäßig über flexible Toleranzen, sodass sie auf gesperrten Knoten geplant werden können, die noch nicht für die Planung oder Ausführung von Pods bereit sind. Sie können die Toleranzen für Ressourcen, die nicht zu DaemonSet Calico gehören, verschärfen, indem Sie Ihre Betreiberinstallation dahingehend ändern, dass sie Folgendes beinhaltet.

```
installation:
  ...
  controlPlaneTolerations:
  - effect: NoExecute
    key: node.kubernetes.io/unreachable
    operator: Exists
    tolerationSeconds: 300
  - effect: NoExecute
    key: node.kubernetes.io/not-ready
    operator: Exists
    tolerationSeconds: 300
  calicoKubeControllersDeployment:
    spec:
      template:
        spec:
          tolerations:
          - effect: NoExecute
            key: node.kubernetes.io/unreachable
            operator: Exists
            tolerationSeconds: 300
          - effect: NoExecute
            key: node.kubernetes.io/not-ready
            operator: Exists
            tolerationSeconds: 300
  typhaDeployment:
```

```
    spec:
      template:
        spec:
          tolerations:
          - effect: NoExecute
            key: node.kubernetes.io/unreachable
            operator: Exists
            tolerationSeconds: 300
          - effect: NoExecute
            key: node.kubernetes.io/not-ready
            operator: Exists
            tolerationSeconds: 300
```

Fehlerbehebung bei Anmeldeinformationen

Sowohl bei AWS SSM-Hybrid-Aktivierungen als auch bei AWS IAM Roles Anywhere können Sie überprüfen, ob die Anmeldeinformationen für die IAM-Rolle Hybrid Nodes auf Ihren Hybridknoten korrekt konfiguriert sind, indem Sie den folgenden Befehl von Ihren Hybridknoten aus ausführen. Überprüfen Sie, ob der Knotenname und der Name der IAM-Rolle für Hybridknoten Ihren Erwartungen entsprechen.

```
sudo aws sts get-caller-identity
```

```
{
    "UserId": "ABCDEFGHIJKLM12345678910:<node-name>",
    "Account": "<aws-account-id>",
    "Arn": "arn:aws: sts::<aws-account-id>:assumed-role/<hybrid-nodes-iam-role/<node-
name>"
}
```

Fehlerbehebung für AWS Systems Manager (SSM)

Wenn Sie AWS SSM-Hybrid-Aktivierungen für Ihre Anmeldeinformationen für Hybridknoten verwenden, beachten Sie die folgenden SSM-Verzeichnisse und -Artefakte, die von auf Ihren Hybridknoten installiert werden. `nodeadm` Weitere Informationen zum SSM-Agenten finden Sie unter Arbeiten mit dem SSM-Agenten im AWS Systems Manager Manager-Benutzerhandbuch.

Description	Speicherort
SSM-Agent	Ubuntu — `/snap/amazon-ssm-agent/current/amazon-ssm-agent` RHEL & 023 — AL2 `/usr/bin/amazon-ssm-agent`
SSM-Agent-Protokolle	`/var/log/amazon/ssm`
AWS Referenzen	`/root/.aws/credentials`
CLI für SSM-Einrichtung	`/opt/ssm/ssm-setup-cli`

Neustart des SSM-Agenten

Einige Probleme können durch einen Neustart des SSM-Agenten behoben werden. Verwenden Sie die folgenden Befehle, um das System neu zu starten.

AL2023 und andere Betriebssysteme

```
systemctl restart amazon-ssm-agent
```

Ubuntu

```
systemctl restart snap.amazon-ssm-agent.amazon-ssm-agent
```

Konnektivität zu SSM-Endpunkten überprüfen

Bestätigen Sie, dass Sie von Ihren Hybridknoten aus eine Verbindung zu den SSM-Endpunkten herstellen können. Eine Liste der SSM-Endpunkte finden Sie unter AWS -Systems-Manager-Endpunkte und -Kontingente. Ersetzen Sie `us-west-2` den folgenden Befehl durch die AWS Region für Ihre AWS SSM-Hybrid-Aktivierung.

```
ping ssm.us-west-2.amazonaws.com
```

Verbindungsstatus registrierter SSM-Instances anzeigen

Sie können den Verbindungsstatus der Instances, die mit SSM-Hybrid-Aktivierungen registriert sind, mit dem folgenden AWS CLI-Befehl überprüfen. Ersetzen Sie die Rechner-ID durch die Rechner-ID Ihrer Instance.

```
aws ssm get-connection-status --target mi-012345678abcdefgh
```

CLI-Prüfsummen-Fehlanpassung bei SSM-Einrichtung

Wenn Sie beim Ausführen von `nodeadm install` ein Problem mit einer nicht übereinstimmenden `ssm-setup-cli`-Prüfsumme feststellen, sollten Sie sicherstellen, dass auf Ihrem Host keine älteren SSM-Installationen vorhanden sind. Wenn auf Ihrem Host ältere SSM-Installationen vorhanden sind, entfernen Sie diese und führen Sie `nodeadm install` erneut aus, um das Problem zu beheben.

```
Failed to perform agent-installation/on-prem registration: error while verifying
 installed ssm-setup-cli checksum: checksum mismatch with latest ssm-setup-cli.
```

SSM **`InvalidActivation`**

Wenn bei der Registrierung Ihrer Instance bei AWS SSM ein Fehler auftritt, überprüfen Sie, ob die Werte `regionactivationCode`, und `activationId` in Ihrem `nodeConfig.yaml` richtig sind. Die AWS Region für Ihren EKS-Cluster muss mit der Region Ihrer SSM-Hybrid-Aktivierung übereinstimmen. Wenn diese Werte falsch konfiguriert sind, wird möglicherweise ein Fehler ähnlich dem folgenden angezeigt.

```
ERROR Registration failed due to error registering the instance with AWS SSM.
 InvalidActivation
```

SSM **`ExpiredTokenException`**: Das in der Anfrage enthaltene Sicherheits-Token ist abgelaufen

Wenn der SSM-Agent die Anmeldeinformationen nicht aktualisieren kann, wird möglicherweise ein `ExpiredTokenException` angezeigt. Wenn Sie in diesem Szenario von Ihren Hybridknoten aus eine Verbindung zu den SSM-Endpunkten herstellen können, müssen Sie den SSM-Agenten möglicherweise neu starten, um eine Aktualisierung der Anmeldeinformationen zu erzwingen.

```
"msg":"Command failed","error":"operation error SSM: DescribeInstanceInformation, https
 response error StatusCode: 400, RequestID: eee03a9e-f7cc-470a-9647-73d47e4cf0be, api
 error ExpiredTokenException: The security token included in the request is expired"
```

SSM-Fehler beim Ausführen des Befehls „Rechner registrieren“

Wenn beim Registrieren des Rechners bei SSM ein Fehler auftritt, müssen Sie `nodeadm install` möglicherweise erneut ausführen, um sicherzustellen, dass alle SSM-Abhängigkeiten ordnungsgemäß installiert sind.

```
"error":"running register machine command: , error: fork/exec /opt/aws/ssm-setup-cli:
 no such file or directory"
```

SSM **`ActivationExpired`**

Wenn bei der Ausführung von `nodeadm init` ein Fehler bei der Registrierung der Instance bei SSM aufgrund einer abgelaufenen Aktivierung auftritt, müssen Sie eine neue SSM-Hybridaktivierung erstellen, `nodeConfig.yaml` mit `activationCode` und `activationId` Ihrer neuen SSM-Hybridaktivierung aktualisieren und `nodeadm init` erneut ausführen.

```
"msg":"Command failed","error":"SSM activation expired. Please use a valid activation"
```

```
ERROR Registration failed due to error registering the instance with AWS SSM.
 ActivationExpired
```

SSM konnte zwischengespeicherte Anmeldeinformationen nicht aktualisieren

Wenn beim Aktualisieren der zwischengespeicherten Anmeldeinformationen ein Fehler auftritt, wurde die `/root/.aws/credentials`-Datei möglicherweise auf Ihrem Host gelöscht. Überprüfen Sie zunächst Ihre SSM-Hybridaktivierung und stellen Sie sicher, dass sie aktiv ist und Ihre Hybridknoten für die Verwendung der Aktivierung korrekt konfiguriert sind. Überprüfen Sie die SSM-Agent-Protokolle unter `/var/log/amazon/ssm` und führen Sie den `nodeadm init`-Befehl erneut aus, sobald Sie das Problem auf der SSM-Seite behoben haben.

```
"Command failed","error":"operation error SSM: DescribeInstanceInformation, get
 identity: get credentials: failed to refresh cached credentials"
```

SSM bereinigen

Um den SSM-Agenten von Ihrem Host zu entfernen, können Sie die folgenden Befehle ausführen.

```
dnf remove -y amazon-ssm-agent
sudo apt remove --purge amazon-ssm-agent
```

```
snap remove amazon-ssm-agent
rm -rf /var/lib/amazon/ssm/Vault/Store/RegistrationKey
```

Fehlerbehebung für AWS IAM Roles Anywhere

Wenn Sie AWS IAM Roles Anywhere für Ihre Anmeldeinformationen für Hybridknoten verwenden, beachten Sie die folgenden Verzeichnisse und Artefakte, die auf Ihren Hybridknoten von installiert werden. `nodeadm` Weitere Informationen zur Problembehandlung bei IAM Roles Anywhere finden Sie unter Problembehandlung bei Identität und Zugriff auf AWS IAM Roles Anywhere im AWS IAM Roles Anywhere-Benutzerhandbuch.

Description	Speicherort
CLI für IAM Roles Anywhere	`/usr/local/bin/aws_signing_helper`
Standard-Speicherort und -Name des Zertifikats	`/etc/iam/pki/server.pem`
Standard-Speicherort und -Name der Schlüssel	`/etc/iam/pki/server.key`

IAM Roles Anywhere konnte die zwischengespeicherten Anmeldeinformationen nicht aktualisieren

Wenn beim Aktualisieren der zwischengespeicherten Anmeldeinformationen ein Fehler auftritt, überprüfen Sie den Inhalt von `/etc/aws/hybrid/config` und bestätigen Sie, dass IAM Roles Anywhere in Ihrer `nodeadm`-Konfiguration richtig konfiguriert wurde. Bestätigen Sie, dass `/etc/iam/pki` vorhanden ist. Jeder Knoten muss über ein eindeutiges Zertifikat und einen eindeutigen Schlüssel verfügen. Wenn Sie IAM Roles Anywhere als Anmeldeinformationsanbieter verwenden, verwendet `nodeadm` standardmäßig `/etc/iam/pki/server.pem` für den Speicherort und Namen des Zertifikats und `/etc/iam/pki/server.key` für den privaten Schlüssel. Möglicherweise müssen Sie die Verzeichnisse erstellen, bevor Sie die Zertifikate und Schlüssel in den Verzeichnissen mit `sudo mkdir -p /etc/iam/pki` platzieren. Sie können den Inhalt Ihres Zertifikats mit dem folgenden Befehl überprüfen.

```
openssl x509 -text -noout -in server.pem
```

```
open /etc/iam/pki/server.pem: no such file or directory
could not parse PEM data
```

```
Command failed {"error": "... get identity: get credentials: failed to refresh cached
 credentials, process provider error: error in credential_process: exit status 1"}
```

IAM Roles Anywhere ist nicht autorisiert, **`sts:AssumeRole`** auszuführen

Wenn Sie in den `kubelet`-Protokollen ein Problem mit einer Zugriffsverweigerung für die `sts:AssumeRole`-Operation bei Verwendung von IAM Roles Anywhere feststellen, überprüfen Sie die Vertrauensrichtlinie Ihrer IAM-Rolle für Hybridknoten, um sicherzustellen, dass der Service-Prinzipal von IAM Roles Anywhere die IAM-Rolle für Hybridknoten übernehmen darf. Bestätigen Sie außerdem, dass der Trust-Anchor-ARN in Ihrer Vertrauensrichtlinie für die IAM-Rolle für Hybridknoten ordnungsgemäß konfiguriert ist und dass Ihre IAM-Rolle für Hybridknoten zu Ihrem IAM–Roles-Anywhere-Profil hinzugefügt wurde.

```
could not get token: AccessDenied: User: ... is not authorized to perform:
 sts:AssumeRole on resource: ...
```

IAM Roles Anywhere ist nicht autorisiert, **`roleSessionName`** festzulegen

Wenn Sie in den `kubelet`-Protokollen ein Problem mit einer Zugriffsverweigerung für die Einstellung von `roleSessionName` feststellen, überprüfen Sie, ob Sie `acceptRoleSessionName` für Ihr IAM-Roles-Anywhere-Profil auf „true“ gesetzt haben.

```
AccessDeniedException: Not authorized to set roleSessionName
```

Fehlerbehebung beim Betriebssystem

RHEL

Fehler bei der Registrierung des Berechtigungs- oder Abonnement-Managers

Wenn Sie `nodeadm install` ausführen und die Installation der Hybridknoten-Abhängigkeiten aufgrund von Problemen bei der Berechtigungsregistrierung fehlschlägt, stellen Sie sicher, dass Sie Ihren Red-Hat-Benutzernamen und Ihr Kennwort auf Ihrem Host richtig eingerichtet haben.

```
This system is not registered with an entitlement server
```

Ubuntu

GLIBC nicht gefunden

Wenn Sie Ubuntu als Betriebssystem und IAM Roles Anywhere als Anmeldeinformationsanbieter mit Hybridknoten verwenden und ein Problem mit „GLIBC nicht gefunden“ feststellen, können Sie diese Abhängigkeit manuell installieren, um das Problem zu beheben.

```
GLIBC_2.32 not found (required by /usr/local/bin/aws_signing_helper)
```

Führen Sie die folgenden Befehle aus, um die Abhängigkeit zu installieren:

```
ldd --version
sudo apt update && apt install libc6
sudo apt install glibc-source
```

Bottlerocket

Wenn Sie den Bottlerocket-Admin-Container aktiviert haben, können Sie mit SSH darauf zugreifen, um erweitertes Debugging und Fehlerbehebung mit erhöhten Berechtigungen durchzuführen Die folgenden Abschnitte enthalten Befehle, die im Kontext des Bottlerocket-Hosts ausgeführt werden müssen. Sobald Sie sich im Admin-Container befinden, können Sie `sheltie` ausführen, um eine vollständige Root-Shell im Bottlerocket-Host zu erhalten.

```
sheltie
```

Sie können die Befehle in den folgenden Abschnitten auch von der Admin-Container-Shell aus ausführen, indem Sie jedem Befehl ein `sudo chroot /.bottlerocket/rootfs` voranstellen.

```
sudo chroot /.bottlerocket/rootfs <command>
```

Verwendung von logdog zur Protokoll-Erfassung

Bottlerocket bietet ein Service-Programm `logdog` zur effizienten Erfassung von Protokollen und Systeminformationen für die Fehlerbehebung.

```
logdog
```

Das Service-Programm `logdog` erfasst Protokolle von verschiedenen Speicherorten auf einem Bottlerocket-Host und fasst sie in einem Tarball zusammen. Standardmäßig wird das Tarball-Archiv unter `/var/log/support/bottlerocket-logs.tar.gz` erstellt und ist von Host-Containern unter `/.bottlerocket/support/bottlerocket-logs.tar.gz` zugänglich.

Zugriff auf Systemprotokolle mit journalctl

Sie können den Status der verschiedenen System-Services wie `kubelet`, `containerd`, usw. überprüfen und ihre Protokolle mit den folgenden Befehlen anzeigen. Das `-f`-Flag wird die Protokolle in Echtzeit verfolgen.

Um den Status des `kubelet`-Services zu überprüfen und `kubelet`-Protokolle abzurufen, können Sie folgenden Befehl ausführen:

```
systemctl status kubelet
journalctl -u kubelet -f
```

Um den Status des `containerd`-Services zu überprüfen und die Protokolle für die orchestrierte `containerd`-Instance abzurufen, können Sie folgenden Befehl ausführen:

```
systemctl status containerd
journalctl -u containerd -f
```

Um den Status des `host-containerd`-Services zu überprüfen und die Protokolle für die `containerd`-Host-Instance abzurufen, können Sie folgenden Befehl ausführen:

```
systemctl status host-containerd
journalctl -u host-containerd -f
```

Um die Protokolle für die Bootstrap-Container und Host-Container abzurufen, können Sie folgenden Befehl ausführen:

```
journalctl _COMM=host-ctr -f
```

Verwendung von Anwendungsdatenspeichern für Ihren Cluster

Sie können mit Amazon EKS eine Reihe von AWS Speicherservices für die Speicheranforderungen Ihrer Anwendungen nutzen. Durch eine Vielzahl von AWS unterstützten Container Storage Interface (CSI) -Treibern können Sie Amazon EBS, Amazon S3, Amazon EFS, Amazon FSX und Amazon File Cache problemlos für die Speicheranforderungen Ihrer auf Amazon EKS ausgeführten Anwendungen verwenden. Informationen zur Verwaltung von Backups Ihres Amazon EKS-Clusters finden Sie unter AWS Backup-Unterstützung für Amazon EKS.

In diesem Kapitel werden Speicheroptionen für Amazon EKS-Cluster behandelt.

Themen

- Kubernetes-Volume-Speicher mit Amazon EBS verwenden
- Verwendung von elastischem Dateisystemspeicher mit Amazon EFS
- Verwendung von leistungsstarkem App-Speicher mit Amazon FSx für Lustre
- Verwendung von leistungsstarkem App-Speicher mit FSx für NetApp ONTAP
- Datenspeicherung mit Amazon FSx für OpenZFS verwenden
- Minimierung der Latenz mit Amazon File Cache
- Zugriff auf Amazon-S3-Objekte mit dem CSI-Treiber für Mountpoint für Amazon S3
- Snapshot-Funktion für CSI-Volumes aktivieren

Kubernetes-Volume-Speicher mit Amazon EBS verwenden

> **Note**
>
> Neu: Amazon EKS Auto Mode automatisiert Routineaufgaben für Blockspeicher. Weitere Informationen erhalten Sie unter the section called “Workload mit Statusverwaltung bereitstellen”.

Der Amazon Elastic Block Store (Amazon EBS) Container-Storage-Interface (CSI)-Treiber verwaltet den Lebenszyklus von Amazon-EBS-Volumes als Speicher für von Ihnen erstellte Kubernetes-

Volumes. Der Amazon-EBS-CSI-Treiber erstellt Amazon-EBS-Volumes für diese Arten von Kubernetes-Volumes: generische flüchtige Volumes und persistente Volumes.

Überlegungen

- Es ist nicht erforderlich, den Amazon-EBS-CSI-Controller in Clustern von EKS Auto Mode zu installieren.
- Sie können Amazon-EBS-Volumes nicht in Fargate-Pods mounten.
- Sie können den Amazon EBS CSI-Controller auf Fargate-Knoten ausführen, aber der Amazon EBS CSI-Knoten `DaemonSet` kann nur auf Amazon-Instances ausgeführt werden. EC2
- Amazon-EBS-Volumes und der Amazon-EBS-CSI-Treiber sind nicht mit Amazon EKS Hybrid Nodes kompatibel.
- Support wird für die neueste Add-On-Version und eine vorherige Version bereitgestellt. Fehler oder Schwachstellen, die in der neuesten Version gefunden werden, werden in einer neuen Nebenversion auf die vorherige Version zurückportiert.
- EKS Auto Mode erfordert, dass Speicherklassen `ebs.csi.eks.amazonaws.com` als Bereitsteller verwenden. Der Standard-Amazon-EBS-CSI-Treiber (`ebs.csi.aws.com`) verwaltet seine eigenen Volumes separat. Um vorhandene Volumes mit EKS Auto Mode zu verwenden, migrieren Sie diese mithilfe von Volume-Snapshots in eine Speicherklasse, die den Bereisteller im Automatikmodus verwendet.

> **Important**
>
> Um die Snapshot-Funktionalität des Amazon-EBS-CSI-Treibers nutzen zu können, ist es erforderlich, zunächst den CSI-Snapshot-Controller zu installieren. Weitere Informationen finden Sie unter the section called “CSI-Snapshot-Controller”.

Voraussetzungen

- Einen vorhandenen -Cluster. Führen Sie den folgenden Befehl aus, um die erforderliche Plattformversion anzuzeigen.

```
aws eks describe-addon-versions --addon-name aws-ebs-csi-driver
```

- Der EBS CSI-Treiber benötigt IAM-Berechtigungen. AWS

- AWS schlägt vor, EKS Pod Identities zu verwenden. Weitere Informationen finden Sie unter the section called “Übersicht über die Einrichtung von EKS-Pod-Identitäten”.
- Informationen zu IAM-Rollen für Servicekonten finden Sie unter the section called “IAM-OIDC-Anbieter”.

Schritt 1: Erstellen einer IAM-Rolle

Das Amazon EBS CSI-Plugin benötigt IAM-Berechtigungen, um in Ihrem Namen Anrufe tätigen AWS APIs zu können. Wenn Sie diese Schritte nicht ausführen, wird beim Versuch, das Add-On zu installieren und `kubectl describe pvc` auszuführen, `failed to provision volume with StorageClass` zusammen mit einem `could not create volume in EC2: UnauthorizedOperation`-Fehler angezeigt. Weitere Informationen finden Sie unter Treiberberechtigung einrichten auf GitHub.

> **Note**
>
> Pods haben Zugriff auf die Berechtigungen, die der IAM-Rolle zugewiesen sind, es sei denn, Sie blockieren den Zugriff auf IMDS. Weitere Informationen finden Sie unter the section called “Bewährte Methoden”.

Das folgende Verfahren zeigt Ihnen, wie Sie eine IAM-Rolle erstellen und ihr die AWS verwaltete Richtlinie zuordnen. Um dieses Verfahren umzusetzen, können Sie eines dieser Tools verwenden:

- the section called “`eksctl`”
- the section called “AWS-Managementkonsole”
- the section called “AWS CLI”

> **Note**
>
> Sie können eine selbstverwaltete Richtlinie mit eingeschränkten Berechtigungen erstellen. Überprüfen Sie `AmazonEBSCSIDriverPolicy` und erstellen Sie eine benutzerdefinierte IAM-Richtlinie mit eingeschränkten Berechtigungen.

> **Note**
>
> Die spezifischen Schritte in diesem Verfahren sind für die Verwendung des Treibers als Amazon-EKS-Add-on geschrieben. Es sind verschiedene Schritte erforderlich, um den Treiber als selbstverwaltetes Add-on zu verwenden. Weitere Informationen finden Sie unter Fahrerberechtigungen einrichten für. GitHub

`eksctl`

1. Erstellen Sie eine IAM-Rolle und fügen Sie eine Richtlinie hinzu. AWS verwaltet eine AWS verwaltete Richtlinie oder Sie können Ihre eigene benutzerdefinierte Richtlinie erstellen. Sie können eine IAM-Rolle erstellen und die AWS verwaltete Richtlinie mit dem folgenden Befehl anhängen. Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters. Der Befehl stellt einen AWS CloudFormation Stack bereit, der eine IAM-Rolle erstellt und ihr die IAM-Richtlinie anhängt.

   ```
   eksctl create iamserviceaccount \
           --name ebs-csi-controller-sa \
           --namespace kube-system \
           --cluster my-cluster \
           --role-name AmazonEKS_EBS_CSI_DriverRole \
           --role-only \
           --attach-policy-arn arn:aws: iam::aws:policy/service-role/
   AmazonEBSCSIDriverPolicy \
           --approve
   ```

2. Sie können diesen Schritt überspringen, wenn Sie keinen benutzerdefinierten KMS-Schlüssel verwenden. Wenn Sie einen zur Verschlüsselung Ihrer Amazon-EBS-Volumes verwenden, passen Sie die IAM-Rolle nach Bedarf an. Führen Sie beispielsweise folgende Schritte aus:

   a. Kopieren Sie den folgenden Code und fügen Sie diesen in eine neue `kms-key-for-encryption-on-ebs.json`-Datei ein. Ersetzen Sie *`custom-key-arn`* durch den benutzerdefinierten KMS-Schlüssel-ARN.

   ```
   {
         "Version":"2012-10-17",
         "Statement": [
           {
             "Effect": "Allow",
             "Action": [
               "kms:CreateGrant",
   ```

```
            "kms:ListGrants",
            "kms:RevokeGrant"
          ],
          "Resource": ["arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"],
          "Condition": {
            "Bool": {
              "kms:GrantIsForAWSResource": "true"
            }
          }
        },
        {
          "Effect": "Allow",
          "Action": [
            "kms:Encrypt",
            "kms:Decrypt",
            "kms:ReEncrypt*",
            "kms:GenerateDataKey*",
            "kms:DescribeKey"
          ],
          "Resource": ["arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"]
        }
      ]
    }
```

b. Erstellen Sie die Richtlinie. Sie können *KMS_Key_For_Encryption_On_EBS_Policy* in einen anderen Namen ändern. In diesem Fall ändern Sie sie auch in den späteren Schritten.

```
aws iam create-policy \
      --policy-name KMS_Key_For_Encryption_On_EBS_Policy \
      --policy-document file://kms-key-for-encryption-on-ebs.json
```

c. Hängen Sie die IAM-Richtlinie mit dem folgenden Befehl an die Rolle an. Ersetzen Sie *111122223333* durch Ihre Konto-ID.

```
aws iam attach-role-policy \
      --policy-arn arn:aws: iam::111122223333:policy/
KMS_Key_For_Encryption_On_EBS_Policy \
      --role-name AmazonEKS_EBS_CSI_DriverRole
```

AWS-Managementkonsole

1. Öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im linken Navigationsbereich Roles aus.
3. Klicken Sie auf der Seite Roles (Rollen) auf Create role (Rolle erstellen).
4. Gehen Sie auf der Seite Select trusted entity (Vertrauenswürdige Entität auswählen) wie folgt vor:
   a. Wählen Sie im Abschnitt Trusted entity type (Typ der vertrauenswürdigen Entität) die Option Web identity (Web-Identität) aus.
   b. Wählen Sie für Identity provider (Identitätsanbieter) die Option OpenID Connect provider URL (OpenID-Connect-Anbieter-URL) für Ihren Cluster aus (wie unter Overview (Übersicht) in Amazon EKS gezeigt).
   c. Wählen Sie für Audience (Zielgruppe) `sts.amazonaws.com`.
   d. Wählen Sie Weiter aus.
5. Gehen Sie auf der Seite Add permissions (Berechtigungen hinzufügen) wie folgt vor:
   a. Geben Sie im Feld Filter policies (Filterrichtlinien) `AmazonEBSCSIDriverPolicy` ein.
   b. Aktivieren Sie das Kontrollkästchen links neben der `AmazonEBSCSIDriverPolicy`, die bei der Suche zurückgegeben wurde.
   c. Wählen Sie Weiter aus.
6. Gehen Sie auf der Seite Name, review, and create (Benennen, überprüfen und erstellen) wie folgt vor:
   a. Geben Sie unter Role name (Rollenname) einen eindeutigen Namen für die Rolle ein, z. B. *`AmazonEKS_EBS_CSI_DriverRole`*.
   b. Fügen Sie der Rolle unter Tags hinzufügen (optional) Metadaten hinzu, indem Sie Tags als Schlüssel-Wert-Paare anfügen. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen im IAM-Benutzerhandbuch.
   c. Wählen Sie Create role (Rolle erstellen) aus.
7. Nachdem die Rolle erstellt wurde, wählen Sie die Rolle in der Konsole aus, um sie zur Bearbeitung zu öffnen.
8. Wählen Sie die Registerkarte Trust Relationships (Vertrauensstellungen) und dann Edit trust policy (Vertrauensrichtlinie bearbeiten) aus.
9. Suchen Sie die Zeile, die der folgenden Zeile ähnelt:

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud":
 "sts.amazonaws.com"
```

Fügen Sie am Ende der vorherigen Zeile ein Komma hinzu und fügen Sie anschließend die folgende Zeile nach der vorherigen Zeile hinzu. *region-code*Ersetzen Sie durch die AWS Region, in der sich Ihr Cluster befindet. Ersetzen Sie *EXAMPLED539D4633E53DE1B71EXAMPLE* mit dem OIDC-Anbieter-ID Ihres Clusters.

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub":
 "system:serviceaccount:kube-system:ebs-csi-controller-sa"
```

10. Wählen Sie Update Policy (Richtlinie aktualisieren) aus, um den Vorgang abzuschließen.
11. Wenn Sie einen benutzerdefinierten KMS-Schlüssel für die Verschlüsselung auf Ihren Amazon-EBS-Volumes verwenden, passen Sie die IAM-Rolle nach Bedarf an. Führen Sie beispielsweise folgende Schritte aus:
    a. Wählen Sie im linken Navigationsbereich die Option Policies aus.
    b. Wählen Sie auf der Seite Policies (Richtlinien) die Option Create a policy (Richtlinie erstellen).
    c. Wählen Sie auf der Seite Create policy (Richtlinie erstellen) die Registerkarte JSON aus.
    d. Kopieren Sie den folgenden Code in den Editor und ersetzen Sie *custom-key-arn* durch den benutzerdefinierten KMS-Schlüssel-ARN:

```
{
      "Version":"2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": [
            "kms:CreateGrant",
            "kms:ListGrants",
            "kms:RevokeGrant"
          ],
          "Resource": ["arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"],
          "Condition": {
            "Bool": {
              "kms:GrantIsForAWSResource": "true"
            }
          }
```

```
        },
        {
          "Effect": "Allow",
          "Action": [
            "kms:Encrypt",
            "kms:Decrypt",
            "kms:ReEncrypt*",
            "kms:GenerateDataKey*",
            "kms:DescribeKey"
          ],
          "Resource": ["arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"]
        }
      ]
    }
```

e. Wählen Sie Next: Markierungen (Weiter: Markierungen).

f. Wählen Sie auf der Seite Add Tags (optional) (Tags hinzufügen (optional)) die Option Next: Review (Weiter: Prüfen).

g. Geben Sie unter Name einen eindeutigen Namen für Ihre Richtlinie ein (z. B. *KMS_Key_For_Encryption_On_EBS_Policy*).

h. Wählen Sie Create Policy (Richtlinie erstellen) aus.

i. Wählen Sie im linken Navigationsbereich Roles aus.

j. Wählen Sie ***AmazonEKS_EBS_CSI_DriverRole***in der Konsole das aus, um es zur Bearbeitung zu öffnen.

k. Wählen Sie in der Dropdown-Liste Add permissions (Berechtigungen hinzufügen) Attach policies (Richtlinien hinzufügen) aus.

l. Geben Sie im Feld Filter policies (Filterrichtlinien) *KMS_Key_For_Encryption_On_EBS_Policy* ein.

m. Aktivieren Sie das Kontrollkästchen links neben der *KMS_Key_For_Encryption_On_EBS_Policy*, die bei der Suche zurückgegeben wurde.

n. Wählen Sie Attach Policies (Richtlinien hinzufügen).

AWS CLI

1. Zeigen Sie die OIDC-Anbieter-URL Ihres Clusters an. Ersetzen Sie *my-cluster* durch Ihren Clusternamen. Wenn die Ausgabe des Befehls `None` ist, überprüfen Sie die Voraussetzungen.

```
aws eks describe-cluster --name my-cluster --query "cluster.identity.oidc.issuer" --
output text
```

Eine Beispielausgabe sieht wie folgt aus.

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

2. Erstellen Sie die IAM-Rolle und gewähren Sie ihr die `AssumeRoleWithWebIdentity`-Aktion.

   a. Kopieren Sie den folgenden Inhalt in eine Datei mit dem Namen `aws-ebs-csi-driver-trust-policy.json`. Ersetzen Sie *`111122223333`* durch Ihre Konto-ID. Ersetzen Sie *`EXAMPLED539D4633E53DE1B71EXAMPLE`* und *`region-code`* mit den im vorherigen Schritt zurückgegebenen Werten.

   ```
   {
         "Version":"2012-10-17",
         "Statement": [
           {
             "Effect": "Allow",
             "Principal": {
               "Federated": "arn:aws:iam::111122223333:oidc-provider/oidc.eks.us-
   east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
             },
             "Action": "sts:AssumeRoleWithWebIdentity",
             "Condition": {
               "StringEquals": {
                 "oidc.eks.us-east-1.amazonaws.com/id/
   EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
                 "oidc.eks.us-east-1.amazonaws.com/id/
   EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-system:ebs-csi-
   controller-sa"
               }
             }
           }
         ]
       }
   ```

   b. Erstellen Sie die Rolle. Sie können *`AmazonEKS_EBS_CSI_DriverRole`* in einen anderen Namen ändern. Wenn Sie es ändern, ändern Sie es in späteren Schritten.

   ```
   aws iam create-role \
   ```

```
      --role-name AmazonEKS_EBS_CSI_DriverRole \
      --assume-role-policy-document file://"aws-ebs-csi-driver-trust-policy.json"
```

3. Hängen Sie eine Richtlinie an. AWS verwaltet eine AWS verwaltete Richtlinie oder Sie können Ihre eigene benutzerdefinierte Richtlinie erstellen. Hängen Sie die AWS verwaltete Richtlinie mit dem folgenden Befehl an die Rolle an.

```
aws iam attach-role-policy \
      --policy-arn arn:aws: iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy \
      --role-name AmazonEKS_EBS_CSI_DriverRole
```

4. Wenn Sie einen benutzerdefinierten KMS-Schlüssel für die Verschlüsselung auf Ihren Amazon-EBS-Volumes verwenden, passen Sie die IAM-Rolle nach Bedarf an. Führen Sie beispielsweise folgende Schritte aus:

   a. Kopieren Sie den folgenden Code und fügen Sie diesen in eine neue `kms-key-for-encryption-on-ebs.json`-Datei ein. Ersetzen Sie *`custom-key-arn`* durch den benutzerdefinierten KMS-Schlüssel-ARN.

```
{
      "Version":"2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": [
            "kms:CreateGrant",
            "kms:ListGrants",
            "kms:RevokeGrant"
          ],
          "Resource": ["arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"],
          "Condition": {
            "Bool": {
              "kms:GrantIsForAWSResource": "true"
            }
          }
        },
        {
          "Effect": "Allow",
          "Action": [
            "kms:Encrypt",
            "kms:Decrypt",
            "kms:ReEncrypt*",
```

```
            "kms:GenerateDataKey*",
            "kms:DescribeKey"
          ],
          "Resource": ["arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"]
        }
      ]
    }
```

b. Erstellen Sie die Richtlinie. Sie können *KMS_Key_For_Encryption_On_EBS_Policy* in einen anderen Namen ändern. In diesem Fall ändern Sie sie auch in den späteren Schritten.

```
aws iam create-policy \
      --policy-name KMS_Key_For_Encryption_On_EBS_Policy \
      --policy-document file://kms-key-for-encryption-on-ebs.json
```

c. Hängen Sie die IAM-Richtlinie mit dem folgenden Befehl an die Rolle an. Ersetzen Sie *111122223333* durch Ihre Konto-ID.

```
aws iam attach-role-policy \
      --policy-arn arn:aws: iam::111122223333:policy/
KMS_Key_For_Encryption_On_EBS_Policy \
      --role-name AmazonEKS_EBS_CSI_DriverRole
```

Nachdem Sie die IAM-Rolle des Amazon-EBS-CSI-Treibers erstellt haben, können Sie mit dem nächsten Abschnitt fortfahren. Wenn Sie das Add-on mit dieser IAM-Rolle bereitstellen, wird ein Servicekonto mit dem Namen `ebs-csi-controller-sa` erstellt und für die Verwendung konfiguriert. Das Servicekonto ist an ein Kubernetes `clusterrole` gebunden, dem die erforderlichen Kubernetes-Berechtigungen zugewiesen sind.

Schritt 2: Amazon-EBS-CSI-Treiber herunterladen

Wir empfehlen Ihnen, den Amazon-EBS-CSI-Treiber über das Amazon-EKS-Add-On zu installieren, um die Sicherheit zu erhöhen und den Arbeitsaufwand zu reduzieren. Wenn Sie ein Amazon-EKS-Add-on zu Ihrem Cluster hinzufügen möchten, lesen Sie the section called “Add-On erstellen”. Weitere Informationen zu Add-ons finden Sie unter the section called “Amazon-EKS-Add-ons”.

> **Important**
>
> Bevor Sie den Amazon-EBS-Treiber als Amazon-EKS-Add-On hinzufügen, stellen Sie sicher, dass Sie keine selbstverwaltete Version des Treibers auf Ihrem Cluster installiert haben. Falls ja, siehe Deinstallation eines selbstverwalteten Amazon-EBS-CSI-Treibers in GitHub.

> **Note**
>
> Standardmäßig ist die vom EBS-CSI verwendete RBAC-Rolle berechtigt, Knoten zu mutieren, um deren Funktion zum Entfernen von Verunreinigungen zu unterstützen. Aufgrund der Einschränkungen von Kubernetes RBAC kann Kubernetes dadurch auch jeden anderen Knoten im Cluster mutieren. Das Helm-Diagramm hat einen Parameter (`node.serviceAccount.disableMutation`), der die Änderung der Node-RBAC-Berechtigungen für das Dienstkonto deaktiviert. ebs-csi-node Wenn diese Option aktiviert ist, funktionieren Treiberfunktionen wie das Entfernen von Flecken nicht.

Wenn Sie alternativ eine selbstverwaltete Installation des Amazon EBS CSI-Treibers wünschen, finden Sie weitere Informationen unter Installation auf. GitHub

Schritt 3: Beispielanwendung bereitstellen

Sie können eine Vielzahl von Beispiel-Apps bereitstellen und diese nach Bedarf ändern. Weitere Informationen finden Sie unter Kubernetes-Beispiele unter. GitHub

Verwendung von elastischem Dateisystemspeicher mit Amazon EFS

Amazon Elastic File System (Amazon EFS) bietet vollständig elastischen Serverless-Dateispeicher, sodass Sie Dateidaten gemeinsam nutzen können, ohne Speicherkapazität und Leistung bereitstellen oder verwalten zu müssen. Der Amazon EFS Container Storage Interface (CSI) -Treiber bietet eine CSI-Schnittstelle, über die Kubernetes-Cluster ausgeführt werden, AWS um den Lebenszyklus von Amazon EFS-Dateisystemen zu verwalten. In diesem Thema erfahren Sie, wie Sie den Amazon-EFS-CSI-Treiber für Ihren Amazon-EKS-Cluster bereitstellen.

Überlegungen

- Der Amazon-EFS-CSI-Treiber ist nicht mit Windows-basierten Container-Images kompatibel.
- Sie können die dynamische Bereitstellung für persistente Volumes mit Fargate-Knoten nicht verwenden, aber Sie können die statische Bereitstellung verwenden.
- Dynamische Bereitstellung erfordert Version 1.2 oder höher des Treibers. Sie können die statische Bereitstellung für persistente Volumes mithilfe der `1.1`-Version des Treibers in jeder unterstützten Amazon-EKS-Cluster-Version verwenden (siehe Unterstützte Amazon-EKS-Versionen).
- Version 1.3.2 oder höher dieses Treibers unterstützt die Arm64-Architektur, einschließlich Amazon EC2 Graviton-basierter Instances.
- Version 1.4.2 oder höher dieses Treibers unterstützt die Verwendung von FIPS zum Einbinden von Dateisystemen.
- Beachten Sie die Ressourcenkontingente für Amazon EFS. Beispielsweise gibt es ein Kontingent von 1 000 Zugriffspunkten, die für jedes Amazon-EFS-Dateisystem erstellt werden können. Weitere Informationen finden Sie unter Amazon-EFS-Ressourcenkontingente, die Sie nicht ändern können.
- Ab Version 2.0.0 wurde bei diesem Treiber für TLS-Verbindungen von der Verwendung von `stunnel` auf `efs-proxy` umgestellt. Bei Verwendung von `efs-proxy` wird eine Anzahl von Threads geöffnet, die eins plus der Anzahl der Kerne des Knotens entspricht, auf dem er ausgeführt wird.
- Der Amazon-EFS-CSI-Treiber ist mit Amazon EKS Hybrid Nodes nicht kompatibel.

Voraussetzungen

- Der Amazon EFS CSI-Treiber benötigt AWS Identity and Access Management (IAM) -Berechtigungen.
  - AWS schlägt vor, EKS Pod Identities zu verwenden. Weitere Informationen finden Sie unter the section called “Übersicht über die Einrichtung von EKS-Pod-Identitäten”.
  - Informationen zu IAM-Rollen für Servicekonten und zum Einrichten eines IAM OpenID Connect (OIDC)-Anbieters für Ihren Cluster finden Sie unter the section called “IAM-OIDC-Anbieter”.
- Version `2.12.3` oder höher oder Version `1.27.160` oder höher der auf Ihrem Gerät installierten und konfigurierten AWS Befehlszeilenschnittstelle (AWS CLI) oder AWS CloudShell. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Paketmanager wie `yum` `apt-get`, oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der AWS CLI. Informationen zur Installation

der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Die AWS CLI-Version, in der installiert ist, AWS CloudShell kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zur Aktualisierung finden Sie im AWS CloudShell Benutzerhandbuch unter AWS CLI in Ihrem Home-Verzeichnis installieren.

- Das `kubectl`-Befehlszeilen-Tool ist auf Ihrem Gerät oder in der AWS CloudShell installiert. Die Version kann mit der Kubernetes-Version Ihres Clusters identisch sein oder bis zu einer Nebenversion älter oder neuer sein. Wenn Ihre Clusterversion beispielsweise `1.29` ist, können Sie `kubectl`-Version `1.28`, `1.29`, oder `1.30` damit verwenden. Informationen zum Installieren oder Aktualisieren von `kubectl` finden Sie unter the section called “kubectl und eksctl einrichten”.

> **Note**
>
> Ein in Fargate ausgeführter Pod bindet automatisch ein Amazon-EFS-Dateisystem ein, ohne dass manuelle Schritte zur Treiberinstallation erforderlich sind.

Schritt 1: Erstellen einer IAM-Rolle

Der Amazon-EFS-CSI-Treiber benötigt IAM-Berechtigungen, um mit Ihrem Dateisystem zu interagieren. Erstellen Sie eine IAM-Rolle und fügen Sie ihr die erforderliche AWS verwaltete Richtlinie hinzu. Um dieses Verfahren umzusetzen, können Sie eines dieser Tools verwenden:

- the section called “`eksctl`”
- the section called “AWS-Managementkonsole”
- the section called “AWS CLI”

> **Note**
>
> Die spezifischen Schritte in diesem Verfahren sind für die Verwendung des Treibers als Amazon-EKS-Add-on geschrieben. Ausführliche Informationen zu selbstverwalteten Installationen finden Sie auf GitHub unter Set up driver permission.

`eksctl`

Bei Verwendung von Pod Identities

Führen Sie die folgenden Befehle aus, um eine IAM-Rolle und eine Pod-Identity-Zuordnung mit `eksctl` zu erstellen. Ersetzen Sie `my-cluster` mit Ihrem Clusternamen. Sie können `AmazonEKS_EFS_CSI_DriverRole` durch einen anderen Namen ersetzen.

```
export cluster_name=my-cluster
export role_name=AmazonEKS_EFS_CSI_DriverRole
eksctl create podidentityassociation \
    --service-account-name efs-csi-controller-sa \
    --namespace kube-system \
    --cluster $cluster_name \
    --role-name $role_name \
    --permission-policy-arns arn:aws: iam::aws:policy/service-role/
AmazonEFSCSIDriverPolicy
```

Bei Verwendung von IAM-Rollen für Servicekonten

Führen Sie die folgenden Befehle aus, um eine IAM-Rolle mit `eksctl` zu erstellen. Ersetzen Sie `my-cluster` mit Ihrem Clusternamen. Sie können `AmazonEKS_EFS_CSI_DriverRole` durch einen anderen Namen ersetzen.

```
export cluster_name=my-cluster
export role_name=AmazonEKS_EFS_CSI_DriverRole
eksctl create iamserviceaccount \
    --name efs-csi-controller-sa \
    --namespace kube-system \
    --cluster $cluster_name \
    --role-name $role_name \
    --role-only \
    --attach-policy-arn arn:aws: iam::aws:policy/service-role/AmazonEFSCSIDriverPolicy
 \
    --approve
TRUST_POLICY=$(aws iam get-role --output json --role-name $role_name --query
 'Role.AssumeRolePolicyDocument' | \
    sed -e 's/efs-csi-controller-sa/efs-csi-*/' -e 's/StringEquals/StringLike/')
aws iam update-assume-role-policy --role-name $role_name --policy-document
 "$TRUST_POLICY"
```

AWS-Managementkonsole

Führen Sie den folgenden Befehl aus, um eine IAM-Rolle mit zu erstellen. AWS-Managementkonsole

1. Öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im linken Navigationsbereich Roles aus.
3. Klicken Sie auf der Seite Roles (Rollen) auf Create role (Rolle erstellen).
4. Gehen Sie auf der Seite Select trusted entity (Vertrauenswürdige Entität auswählen) wie folgt vor:
   a. Bei Verwendung von EKS Pod Identities:
      i. Wählen Sie im Abschnitt Vertrauenswürdiger Entitätstyp die Option AWS Service aus.
      ii. Wählen Sie im Dropdown-Menü Service oder Anwendungsfall die Option EKS aus.
      iii. Wählen Sie im Abschnitt Anwendungsfall die Option EKS – Pod Identity aus.
      iv. Wählen Sie Weiter aus.
   b. Bei Verwendung von IAM-Rollen für Servicekonten:
      i. Wählen Sie im Abschnitt Trusted entity type (Typ der vertrauenswürdigen Entität) die Option Web identity (Web-Identität) aus.
      ii. Wählen Sie für Identity provider (Identitätsanbieter) die Option OpenID Connect provider URL (OpenID-Connect-Anbieter-URL) für Ihren Cluster aus (wie unter Overview (Übersicht) in Amazon EKS gezeigt).
      iii. Wählen Sie für Audience (Zielgruppe) `sts.amazonaws.com`.
      iv. Wählen Sie Weiter aus.
5. Gehen Sie auf der Seite Add permissions (Berechtigungen hinzufügen) wie folgt vor:
   a. Geben Sie im Feld Filter policies (Filterrichtlinien) `AmazonEFSCSIDriverPolicy` ein.
   b. Aktivieren Sie das Kontrollkästchen links neben der `AmazonEFSCSIDriverPolicy`, die bei der Suche zurückgegeben wurde.
   c. Wählen Sie Weiter aus.
6. Gehen Sie auf der Seite Name, review, and create (Benennen, überprüfen und erstellen) wie folgt vor:
   a. Geben Sie unter Role name (Rollenname) einen eindeutigen Namen für die Rolle ein, z. B. `AmazonEKS_EFS_CSI_DriverRole`.
   b. Fügen Sie der Rolle unter Tags hinzufügen (optional) Metadaten hinzu, indem Sie Tags als Schlüssel-Wert-Paare anfügen. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen im IAM-Benutzerhandbuch.

c. Wählen Sie Rolle erstellen aus.

7. Nachdem die Rolle erstellt wurde:

a. Bei Verwendung von EKS Pod Identities:

i. Öffnen Sie die Amazon-EKS-Konsole.

ii. Wählen Sie im linken Navigationsbereich Cluster aus und anschließend den Namen des Clusters, für den Sie die EKS-Pod-Identity-Zuordnung konfigurieren möchten.

iii. Wählen Sie die Registerkarte Zugriff aus.

iv. Wählen Sie unter Pod-Identity-Zuordnungen die Option Erstellen aus.

v. Wählen Sie das Dropdown-Menü IAM-Rolle und wählen Sie Ihre neu erstellte Rolle aus.

vi. Wählen Sie das Feld Kubernetes-Namespace und geben Sie `kube-system` ein.

vii. Wählen Sie das Feld Kubernetes-Servicekonto und geben Sie `efs-csi-controller-sa` ein.

viii. Wählen Sie Erstellen aus.

ix. Weitere Informationen zum Erstellen von Pod-Identity-Zuordnungen finden Sie unter the section called “Erstellen Sie eine Pod Identity-Zuordnung (AWS Konsole)”.

b. Bei Verwendung von IAM-Rollen für Servicekonten:

i. Wählen Sie die Rolle aus, um sie zur Bearbeitung zu öffnen.

ii. Wählen Sie die Registerkarte Trust Relationships (Vertrauensstellungen) und dann Edit trust policy (Vertrauensrichtlinie bearbeiten) aus.

iii. Suchen Sie die Zeile, die der folgenden Zeile ähnelt:

```
"oidc.eks.region-code.amazonaws.com/id/<EXAMPLED539D4633E53DE1B71EXAMPLE>:aud":
 "sts.amazonaws.com"
```

Fügen Sie die folgende Zeile über der vorherigen Zeile hinzu. `<region-code>`Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet. Ersetzen Sie `<EXAMPLED539D4633E53DE1B71EXAMPLE>` mit dem OIDC-Anbieter-ID Ihres Clusters.

```
"oidc.eks.<region-code>.amazonaws.com/id/
<EXAMPLED539D4633E53DE1B71EXAMPLE>:sub": "system:serviceaccount:kube-system:efs-
csi-*",
```

iv. Ändern Sie den `Condition`-Operator von `"StringEquals"` zu `"StringLike"`.

v. Wählen Sie Update Policy (Richtlinie aktualisieren) aus, um den Vorgang abzuschließen.

AWS CLI

Führen Sie die folgenden Befehle aus, um eine IAM-Rolle mit AWS CLI zu erstellen.

Bei Verwendung von Pod Identities

1. Erstellen Sie die IAM-Rolle, die dem `pods.eks.amazonaws.com`-Service die Aktionen `AssumeRole` und `TagSession` gewährt.
   a. Kopieren Sie den folgenden Inhalt in eine Datei namens `aws-efs-csi-driver-trust-policy-pod-identity.json`.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEksAuthToAssumeRoleForPodIdentity",
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
```

   b. Erstellen Sie die Rolle. Ersetzen Sie `my-cluster` mit Ihrem Clusternamen. Sie können `AmazonEKS_EFS_CSI_DriverRole` durch einen anderen Namen ersetzen.

```
export cluster_name=my-cluster
export role_name=AmazonEKS_EFS_CSI_DriverRole
aws iam create-role \
  --role-name $role_name \
  --assume-role-policy-document file://"aws-efs-csi-driver-trust-policy-pod-identity.json"
```

2. Fügen Sie der Rolle mit dem folgenden Befehl die erforderliche AWS verwaltete Richtlinie hinzu.

```
aws iam attach-role-policy \
  --policy-arn arn:aws: iam::aws:policy/service-role/AmazonEFSCSIDriverPolicy \
```

```
  --role-name $role_name
```

3. Führen Sie den folgenden Befehl aus, um die Pod-Identity-Zuordnung zu erstellen. Ersetzen Sie `arn:aws: iam::<111122223333>:role/my-role` durch die in den vorherigen Schritten erstellte Rolle.

```
aws eks create-pod-identity-association --cluster-name $cluster_name --role-arn {arn-aws}iam::<111122223333>:role/my-role --namespace kube-system --service-account efs-csi-controller-sa
```

4. Weitere Informationen zum Erstellen von Pod-Identity-Zuordnungen finden Sie unter the section called “Erstellen Sie eine Pod Identity-Zuordnung (AWS Konsole)”.

Bei Verwendung von IAM-Rollen für Servicekonten

1. Zeigen Sie die OIDC-Anbieter-URL Ihres Clusters an. Ersetzen Sie `my-cluster` mit Ihrem Clusternamen. Sie können `AmazonEKS_EFS_CSI_DriverRole` durch einen anderen Namen ersetzen.

```
export cluster_name=my-cluster
export role_name=AmazonEKS_EFS_CSI_DriverRole
aws eks describe-cluster --name $cluster_name --query "cluster.identity.oidc.issuer"
 --output text
```

   Eine Beispielausgabe sieht wie folgt aus.

```
https://oidc.eks.<region-code>.amazonaws.com/id/<EXAMPLED539D4633E53DE1B71EXAMPLE>
```

   Wenn die Ausgabe des Befehls `None` ist, überprüfen Sie die Voraussetzungen.

2. Erstellen Sie die IAM-Rolle, die die Aktion `AssumeRoleWithWebIdentity` erlaubt.
   a. Kopieren Sie den folgenden Inhalt in eine Datei namens `aws-efs-csi-driver-trust-policy.json`. Ersetzen Sie <111122223333> durch Ihre Konto-ID. Ersetzen Sie <EXAMPLED539D4633E53DE1B71EXAMPLE> und <region-code> mit den im vorherigen Schritt zurückgegebenen Werten.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-
east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringLike": {
          "oidc.eks.us-east-1.amazonaws.com/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-system:efs-csi-
*",
          "oidc.eks.us-east-1.amazonaws.com/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
        }
      }
    }
  ]
}
```

b. Erstellen Sie die Rolle.

```
aws iam create-role \
  --role-name $role_name \
  --assume-role-policy-document file://"aws-efs-csi-driver-trust-policy.json"
```

3. Fügen Sie der Rolle mit dem folgenden Befehl die erforderliche AWS verwaltete Richtlinie hinzu.

```
aws iam attach-role-policy \
  --policy-arn arn:aws: iam::aws:policy/service-role/AmazonEFSCSIDriverPolicy \
  --role-name $role_name
```

Schritt 2: Amazon-EFS-CSI-Treiber herunterladen

Wir empfehlen, den Amazon-EFS-CSI-Treiber über das Amazon-EKS-Add-on zu installieren. Wenn Sie ein Amazon-EKS-Add-on zu Ihrem Cluster hinzufügen möchten, lesen Sie the section called “Add-On erstellen”. Weitere Informationen zu Add-ons finden Sie unter the section called “Amazon-EKS-Add-ons”. Wenn Sie das Amazon EKS-Add-on nicht verwenden können, empfehlen wir Ihnen, ein Problem mit der Begründung, warum Sie das nicht können, an das GitHub Container-Roadmap-Repository zu senden.

> **Important**
>
> Bevor Sie den Amazon-EFS-Treiber als Amazon-EKS-Add-On hinzufügen, vergewissern Sie sich, dass auf Ihrem Cluster keine selbstverwaltete Version des Treibers installiert ist. Falls ja, finden Sie weitere Informationen unter Deinstallieren des Amazon EFS CSI-Treibers auf GitHub.

Wenn Sie alternativ eine selbstverwaltete Installation des Amazon-EFS-CSI-Treibers wünschen, finden Sie Informationen unter Installation in GitHub.

Schritt 3: Erstellen eines Amazon EFS-Dateisystems

Informationen zum Erstellen eines Amazon-EFS-Dateisystems finden Sie unter Erstellen eines Amazon-EFS-Dateisystems für Amazon EKS in GitHub.

Schritt 4: Beispielanwendung bereitstellen

Sie können eine Vielzahl von Beispiel-Apps bereitstellen und diese nach Bedarf ändern. Weitere Informationen finden Sie unter Beispiele unter GitHub.

Verwendung von leistungsstarkem App-Speicher mit Amazon FSx für Lustre

Der Treiber für Amazon FSx für Lustre Container Storage Interface (CSI) bietet eine CSI-Schnittstelle, mit der Amazon-EKS-Cluster den Lebenszyklus von FSx-für-Lustre-Dateisystemen verwalten können. Weitere Informationen finden Sie im Benutzerhandbuch zu Amazon FSx für Lustre.

Einzelheiten zum Bereitstellen des CSI-Treibers für Amazon FSx für Lustre in Ihrem Amazon-EKS-Cluster und zum Überprüfen seiner Funktion finden Sie unter the section called “Treiber bereitstellen”.

Stellen Sie den FSx for Lustre-Treiber bereit

In diesem Thema erfahren Sie, wie Sie den FSx for Lustre CSI-Treiber in Ihrem Amazon EKS-Cluster bereitstellen und überprüfen, ob er funktioniert. Wir empfehlen die neueste Version des Treibers zu verwenden. Die verfügbaren Versionen finden Sie in der Kompatibilitätsmatrix der CSI-Spezifikation unter GitHub.

> **Note**
>
> Der Treiber wird in Fargate oder Amazon EKS Hybrid Nodes nicht unterstützt.

Eine ausführliche Beschreibung der verfügbaren Parameter und vollständige Beispiele zur Veranschaulichung der Treiberfunktionen finden Sie im Treiberprojekt FSx for Lustre Container Storage Interface (CSI) unter GitHub.

Voraussetzungen

- Einen vorhandenen -Cluster.
- Das Amazon FSx CSI Driver EKS-Add-On benötigt den EKS Pod Identity-Agenten für die Authentifizierung. Ohne diese Komponente schlägt das Add-On mit dem Fehler `Amazon EKS Pod Identity agent is not installed in the cluster` fehl, wodurch Volume-Operationen verhindert werden. Installieren Sie den Pod Identity-Agenten vor oder nach der Bereitstellung des FSx CSI-Treiber-Add-ons. Weitere Informationen finden Sie unter the section called “Agent einrichten”.
- Version `2.12.3` oder höher oder Version `1.27.160` oder höher der auf Ihrem Gerät installierten und konfigurierten AWS Befehlszeilenschnittstelle (AWS CLI) oder AWS CloudShell. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Paketmanager wie `yum`apt-get, oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der AWS CLI. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Die AWS CLI-Version, in der installiert ist, AWS CloudShell kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zur Aktualisierung finden Sie im AWS CloudShell Benutzerhandbuch unter AWS CLI in Ihrem Home-Verzeichnis installieren.
- Version `0.215.0` oder höher des `eksctl`-Befehlszeilen-Tools, das auf Ihrem Computer oder in der AWS CloudShell installiert ist. Informationen zum Installieren und Aktualisieren von `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.
- Das `kubectl`-Befehlszeilen-Tool ist auf Ihrem Gerät oder in der AWS CloudShell installiert. Die Version kann mit der Kubernetes-Version Ihres Clusters identisch sein oder bis zu einer Nebenversion älter oder neuer sein. Wenn Ihre Clusterversion beispielsweise `1.29` ist, können Sie `kubectl`-Version `1.28`, `1.29`, oder `1.30` damit verwenden. Informationen zum Installieren oder Aktualisieren von `kubectl` finden Sie unter the section called “kubectl und eksctl einrichten”.

Schritt 1: Erstellen einer IAM-Rolle

Das Amazon FSx CSI-Plugin benötigt IAM-Berechtigungen, um in Ihrem Namen Anrufe tätigen AWS APIs zu können.

> **Note**
>
> Pods haben Zugriff auf die Berechtigungen, die der IAM-Rolle zugewiesen sind, es sei denn, Sie blockieren den Zugriff auf IMDS. Weitere Informationen finden Sie unter the section called “Bewährte Methoden”.

Das folgende Verfahren zeigt Ihnen, wie Sie eine IAM-Rolle erstellen und ihr die AWS verwaltete Richtlinie zuordnen.

1. Erstellen Sie eine IAM-Rolle und fügen Sie die AWS verwaltete Richtlinie mit dem folgenden Befehl an. Ersetzen Sie `my-cluster` durch den Namen des Clusters, den Sie verwenden möchten. Der Befehl stellt einen AWS CloudFormation Stack bereit, der eine IAM-Rolle erstellt und ihr die IAM-Richtlinie anhängt.

   ```
   eksctl create iamserviceaccount \
       --name fsx-csi-controller-sa \
       --namespace kube-system \
       --cluster my-cluster \
       --role-name AmazonEKS_FSx_CSI_DriverRole \
       --role-only \
       --attach-policy-arn arn:aws: iam::aws:policy/AmazonFSxFullAccess \
       --approve
   ```

   Beim Erstellen des Servicekontos werden Ihnen mehrere Ausgabezeilen angezeigt. Die letzten Ausgabezeilen ähneln den folgenden.

   ```
   [#]  1 task: {
       2 sequential sub-tasks: {
           create IAM role for serviceaccount "kube-system/fsx-csi-controller-sa",
           create serviceaccount "kube-system/fsx-csi-controller-sa",
       } }
   [#]  building iamserviceaccount stack "eksctl-my-cluster-addon-iamserviceaccount-
   kube-system-fsx-csi-controller-sa"
   ```

```
[#]  deploying stack "eksctl-my-cluster-addon-iamserviceaccount-kube-system-fsx-csi-
controller-sa"
[#]  waiting for CloudFormation stack "eksctl-my-cluster-addon-iamserviceaccount-
kube-system-fsx-csi-controller-sa"
[#]  created serviceaccount "kube-system/fsx-csi-controller-sa"
```

Notieren Sie sich den Namen des AWS CloudFormation Stacks, der bereitgestellt wurde. In der vorigen Beispielausgabe lautet der Name des Stacks `eksctl-my-cluster-addon-iamserviceaccount-kube-system-fsx-csi-controller-sa`.

Nachdem Sie die IAM-Rolle des Amazon FSx CSI-Treibers erstellt haben, können Sie mit dem nächsten Abschnitt fortfahren. Wenn Sie das Add-on mit dieser IAM-Rolle bereitstellen, wird ein Servicekonto mit dem Namen `fsx-csi-controller-sa` erstellt und für die Verwendung konfiguriert. Das Servicekonto ist an ein Kubernetes `clusterrole` gebunden, dem die erforderlichen Kubernetes-Berechtigungen zugewiesen sind.

Schritt 2: Installieren Sie den Amazon FSx CSI-Treiber

Wir empfehlen, den Amazon FSx CSI-Treiber über das Amazon EKS-Add-on zu installieren, um die Sicherheit zu verbessern und den Arbeitsaufwand zu reduzieren. Wenn Sie ein Amazon-EKS-Add-on zu Ihrem Cluster hinzufügen möchten, lesen Sie the section called “Add-On erstellen”. Weitere Informationen zu Add-ons finden Sie unter the section called “Amazon-EKS-Add-ons”.

Important

Bereits vorhandene Amazon FSx CSI-Treiberinstallationen im Cluster können zu Fehlern bei der Installation von Add-Ons führen. Wenn Sie versuchen, die Amazon EKS-Add-On-Version zu installieren, während ein FSx Nicht-EKS-CSI-Treiber vorhanden ist, schlägt die Installation aufgrund von Ressourcenkonflikten fehl. Verwenden Sie während der Installation das `OVERWRITE`-Flag, um dieses Problem zu beheben.

```
aws eks create-addon --addon-name aws-fsx-csi-driver --cluster-name my-cluster
 --resolve-conflicts OVERWRITE
```

Wenn Sie alternativ eine selbstverwaltete Installation des Amazon FSx CSI-Treibers wünschen, finden Sie weitere Informationen unter Installation auf GitHub.

Schritt 3: Speicherklasse, persistenter Volume-Anspruchs und Beispiel-App bereitstellen

Bei diesem Verfahren wird das GitHub Treiber-Repository FSx für Lustre Container Storage Interface (CSI) verwendet, um ein dynamisch FSx bereitgestelltes Volume für Lustre zu nutzen.

1. Beachten Sie die Sicherheitsgruppe für Ihren Cluster. Sie können es im AWS-Managementkonsole Abschnitt Netzwerk oder mit dem folgenden AWS CLI-Befehl sehen. Ersetzen Sie `my-cluster` durch den Namen des Clusters, den Sie verwenden möchten.

```
aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId
```

2. Erstellen Sie eine Sicherheitsgruppe für Ihr FSx Amazon-Dateisystem gemäß den Kriterien, die im Amazon FSx for Lustre-Benutzerhandbuch unter Amazon VPC-Sicherheitsgruppen aufgeführt sind. Als VPC wählen Sie die VPC Ihres Clusters aus, die im Abschnitt Networking (Netzwerk) gezeigt wird. Unter „the security groups associated with your Lustre clients“ (die mit Ihren Lustre-Clients verknüpften Sicherheitsgruppen) wählen Sie Ihre Cluster-Sicherheitsgruppe aus. Wenn Sie keine Regeln für ausgehenden Datenverkehr festlegen, können Sie den All traffic (gesamten Datenverkehr) erlauben.
3. Laden Sie das Speicherklassen-Manifest mit dem folgenden Befehl herunter.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/master/
examples/kubernetes/dynamic_provisioning/specs/storageclass.yaml
```

4. Bearbeiten Sie den Parameterabschnitt in der Datei `storageclass.yaml`. Ersetzen Sie jeden Beispielwert durch Ihre eigenen Werte.

```
parameters:
  subnetId: subnet-0eabfaa81fb22bcaf
  securityGroupIds: sg-068000ccf82dfba88
  deploymentType: PERSISTENT_1
  automaticBackupRetentionDays: "1"
  dailyAutomaticBackupStartTime: "00:00"
  copyTagsToBackups: "true"
  perUnitStorageThroughput: "200"
  dataCompressionType: "NONE"
  weeklyMaintenanceStartTime: "7:09:00"
  fileSystemTypeVersion: "2.12"
```

- **`subnetId`**— Die Subnetz-ID, in der das Amazon FSx for Lustre-Dateisystem erstellt werden soll. Amazon FSx for Lustre wird nicht in allen Availability Zones unterstützt. Öffnen Sie die Amazon FSx for Lustre-Konsole unter, https://console.aws.amazon.com/fsx/ um zu bestätigen, dass sich das Subnetz, das Sie verwenden möchten, in einer unterstützten Availability Zone befindet. Das Subnetz kann Ihre Knoten enthalten oder ein anderes Subnetz oder eine andere VPC sein:
  - Sie können nach den Knoten-Subnetzen in der suchen, AWS-Managementkonsole indem Sie die Knotengruppe im Abschnitt Compute auswählen.
  - Wenn das von Ihnen angegebene Subnetz nicht dasselbe Subnetz ist, in dem Sie Knoten haben, VPCs müssen Sie verbunden sein und sicherstellen, dass die erforderlichen Ports in Ihren Sicherheitsgruppen geöffnet sind.
- **`securityGroupIds`** – Die ID der Sicherheitsgruppe, die Sie für das Dateisystem erstellt haben.
- **`deploymentType`** (optional) – Der Bereitstellungstyp des Dateisystems. Gültige Werte sind `SCRATCH_1`, `SCRATCH_2`, `PERSISTENT_1` und `PERSISTENT_2`. Weitere Informationen zu Bereitstellungstypen finden Sie unter Erstellen Sie Ihr Amazon FSx for Lustre-Dateisystem.
- andere Parameter (optional) — Informationen zu den anderen Parametern finden Sie unter Bearbeiten StorageClass am GitHub.

5. Erstellen Sie das Speicherklassen-Manifest.

```
kubectl apply -f storageclass.yaml
```

   Eine Beispielausgabe sieht wie folgt aus.

```
storageclass.storage.k8s.io/fsx-sc created
```

6. Laden Sie das Manifest für den dauerhaften Volume-Anspruch herunter.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/master/
examples/kubernetes/dynamic_provisioning/specs/claim.yaml
```

7. (Optional) Bearbeiten Sie die `claim.yaml`-Datei. Ändern Sie `1200Gi` in einen der folgenden Inkrementwerte, basierend auf Ihren Speicheranforderungen und dem `deploymentType`, den Sie in einem vorherigen Schritt ausgewählt haben.

```
storage: 1200Gi
```

- SCRATCH_2 und PERSISTENT – 1.2 TiB, 2.4 TiB oder Schritte von 2,4 TiB über 2,4 TiB.
- SCRATCH_1 – 1.2 TiB, 2.4 TiB, 3.6 TiB oder Schritte von 3,6 TiB über 3,6 TiB.

8. Erstellen Sie den dauerhaften Volume-Anspruch.

```
kubectl apply -f claim.yaml
```

Eine Beispielausgabe sieht wie folgt aus.

```
persistentvolumeclaim/fsx-claim created
```

9. Vergewissern Sie sich, dass das Dateisystem bereitgestellt wurde.

```
kubectl describe pvc
```

Eine Beispielausgabe sieht wie folgt aus.

```
Name:          fsx-claim
Namespace:     default
StorageClass:  fsx-sc
Status:        Bound
[...]
```

> **Note**
>
> Der Status kann für 5-10 Minuten als Pending angezeigt werden, bevor er zu Bound wechselt. Fahren Sie nicht mit dem nächsten Schritt fort, bis der Status Bound ist. Wenn der Status länger als 10 Minuten Pending anzeigt, verwenden Sie die Warnmeldungen in den Events als Referenz zur Behebung von Problemen.

10. Stellen Sie die Beispielanwendung bereit.

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/master/examples/kubernetes/dynamic_provisioning/specs/pod.yaml
```

11. Stellen Sie sicher, dass die Beispielanwendung ausgeführt wird.

```
kubectl get pods
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME      READY   STATUS              RESTARTS   AGE
fsx-app   1/1     Running             0          8s
```

12. Überprüfen Sie, ob das Dateisystem ordnungsgemäß von der Anwendung aufgespielt wurde.

```
kubectl exec -ti fsx-app -- df -h
```

Eine Beispielausgabe sieht wie folgt aus.

```
Filesystem                  Size  Used Avail Use% Mounted on
overlay                      80G  4.0G   77G   5% /
tmpfs                        64M     0   64M   0% /dev
tmpfs                       3.8G     0  3.8G   0% /sys/fs/cgroup
192.0.2.0@tcp:/abcdef01     1.1T  7.8M  1.1T   1% /data
/dev/nvme0n1p1               80G  4.0G   77G   5% /etc/hosts
shm                          64M     0   64M   0% /dev/shm
tmpfs                       6.9G   12K  6.9G   1% /run/secrets/kubernetes.io/
serviceaccount
tmpfs                       3.8G     0  3.8G   0% /proc/acpi
tmpfs                       3.8G     0  3.8G   0% /sys/firmware
```

13. Stellen Sie sicher, dass Daten von der Beispiel-App in das FSx for Lustre-Dateisystem geschrieben wurden.

```
kubectl exec -it fsx-app -- ls /data
```

Eine Beispielausgabe sieht wie folgt aus.

```
out.txt
```

Diese Beispielausgabe zeigt, dass die Beispiel-App erfolgreich die `out.txt`-Datei in das Dateisystem geschrieben hat.

> **Note**
>
> Stellen Sie vor dem Löschen des Clusters sicher, dass Sie das FSx for Lustre-Dateisystem löschen. Weitere Informationen finden Sie unter Ressourcen bereinigen im FSx for Lustre-Benutzerhandbuch.

Leistungssteigerung für FSx für Lustre

Wenn Sie FSx for Lustre mit Amazon EKS verwenden, können Sie die Leistung optimieren, indem Sie Lustre-Tunings während der Knoteninitialisierung anwenden. Der empfohlene Ansatz besteht darin, Startvorlagen-Benutzerdaten zu verwenden, um eine konsistente Konfiguration auf allen Knoten sicherzustellen.

Diese Optimierungen umfassen:

- Netzwerk- und RPC-Optimierungen
- Lustre-Modulverwaltung
- LRU-Abstimmungen (Lock Resource Unit)
- Einstellungen für die Client-Cache-Steuerung
- RPC-Steuerungen für OST und MDC

Für detaillierte Anweisungen zur Implementierung dieser Leistungsoptimierungen:

- Zur Optimierung der Leistung für Standardknoten (nicht EFA) finden Sie unter the section called “Optimieren (nicht-EFA)” ein vollständiges Skript, das Sie zu Ihren Benutzerdaten der Startvorlage hinzufügen können.
- Informationen zur Optimierung der Leistung für EFA-fähige Knoten finden Sie unter the section called “Optimieren (EFA)”.

Optimieren Sie die Leistung von Amazon FSx for Lustre auf Knoten (EFA)

In diesem Thema wird beschrieben, wie Sie das Elastic Fabric Adapter (EFA) -Tuning mit Amazon EKS und Amazon FSx for Lustre einrichten.

> **Note**
>
> - Informationen zur Erstellung und Bereitstellung des CSI-Treibers FSx für Lustre finden Sie unter. the section called “Treiber bereitstellen”
> - Informationen zur Optimierung von Standardknoten ohne EFA finden Sie unter the section called “Optimieren (nicht-EFA)”.

Schritt 1. EKS-Cluster erstellen

Erstellen Sie einen Cluster unter Verwendung der bereitgestellten Konfigurationsdatei:

```
# Create cluster using efa-cluster.yaml
eksctl create cluster -f efa-cluster.yaml
```

Beispiel: `efa-cluster.yaml`

```
#efa-cluster.yaml

apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: csi-fsx
  region: us-east-1
  version: "1.30"

iam:
  withOIDC: true

availabilityZones: ["us-east-1a", "us-east-1d"]

managedNodeGroups:
  - name: my-efa-ng
    instanceType: c6gn.16xlarge
    minSize: 1
    desiredCapacity: 1
    maxSize: 1
    availabilityZones: ["us-east-1b"]
    volumeSize: 300
    privateNetworking: true
```

```
    amiFamily: Ubuntu2204
    efaEnabled: true
    preBootstrapCommands:
      - |
        #!/bin/bash
        eth_intf="$(/sbin/ip -br -4 a sh | grep $(hostname -i)/ | awk '{print $1}')"
        efa_version=$(modinfo efa | awk '/^version:/ {print $2}' | sed 's/[^0-9.]//g')
        min_efa_version="2.12.1"

        if [[ "$(printf '%s\n' "$min_efa_version" "$efa_version" | sort -V | head -
n1)" != "$min_efa_version" ]]; then
            sudo curl -O https://efa-installer.amazonaws.com/aws-efa-
installer-1.37.0.tar.gz
            tar -xf aws-efa-installer-1.37.0.tar.gz && cd aws-efa-installer
            echo "Installing EFA driver"
            sudo apt-get update && apt-get upgrade -y
            sudo apt install -y pciutils environment-modules libnl-3-dev libnl-
route-3-200 libnl-route-3-dev dkms
            sudo ./efa_installer.sh -y
            modinfo efa
        else
            echo "Using EFA driver version $efa_version"
        fi

        echo "Installing Lustre client"
        sudo wget -O - https://fsx-lustre-client-repo-public-keys.s3.amazonaws.com/fsx-
ubuntu-public-key.asc | gpg --dearmor | sudo tee /usr/share/keyrings/fsx-ubuntu-public-
key.gpg > /dev/null
        sudo echo "deb [signed-by=/usr/share/keyrings/fsx-ubuntu-public-key.gpg]
 https://fsx-lustre-client-repo.s3.amazonaws.com/ubuntu jammy main" > /etc/apt/
sources.list.d/fsxlustreclientrepo.list
        sudo apt update | tail
        sudo apt install -y lustre-client-modules-$(uname -r) amazon-ec2-utils | tail
        modinfo lustre

        echo "Loading Lustre/EFA modules..."
        sudo /sbin/modprobe lnet
        sudo /sbin/modprobe kefalnd ipif_name="$eth_intf"
        sudo /sbin/modprobe ksocklnd
        sudo lnetctl lnet configure

        echo "Configuring TCP interface..."
        sudo lnetctl net del --net tcp 2> /dev/null
        sudo lnetctl net add --net tcp --if $eth_intf
```

```
        # For P5 instance type which supports 32 network cards,
        # by default add 8 EFA interfaces selecting every 4th device (1 per PCI bus)
        echo "Configuring EFA interface(s)..."
        instance_type="$(ec2-metadata --instance-type | awk '{ print $2 }')"
        num_efa_devices="$(ls -1 /sys/class/infiniband | wc -l)"
        echo "Found $num_efa_devices available EFA device(s)"

        if [[ "$instance_type" == "p5.48xlarge" || "$instance_type" ==
 "p5e.48xlarge" ]]; then
           for intf in $(ls -1 /sys/class/infiniband | awk 'NR % 4 == 1'); do
               sudo lnetctl net add --net efa --if $intf --peer-credits 32
          done
        else
        # Other instances: Configure 2 EFA interfaces by default if the instance
 supports multiple network cards,
        # or 1 interface for single network card instances
        # Can be modified to add more interfaces if instance type supports it
            sudo lnetctl net add --net efa --if $(ls -1 /sys/class/infiniband | head -
n1) --peer-credits 32
            if [[ $num_efa_devices -gt 1 ]]; then
               sudo lnetctl net add --net efa --if $(ls -1 /sys/class/infiniband | tail
 -n1) --peer-credits 32
            fi
        fi

        echo "Setting discovery and UDSP rule"
        sudo lnetctl set discovery 1
        sudo lnetctl udsp add --src efa --priority 0
        sudo /sbin/modprobe lustre

        sudo lnetctl net show
        echo "Added $(sudo lnetctl net show | grep -c '@efa') EFA interface(s)"
```

Schritt 2. Knotengruppen erstellen

EFA-fähige Knotengruppe erstellen:

```
# Create node group using efa-ng.yaml
eksctl create nodegroup -f efa-ng.yaml
```

> **Important**
>
> === Passen Sie diese Werte in Abschnitt `# 5. Mount FSx filesystem` an Ihre Umgebung an.

```
FSX_DNS="<your-fsx-filesystem-dns>" # Needs to be adjusted.
MOUNT_NAME="<your-mount-name>" # Needs to be adjusted.
MOUNT_POINT="</your/mount/point>" # Needs to be adjusted.
```

===

Beispiel`efa-ng.yaml`:

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: final-efa
  region: us-east-1

managedNodeGroups:
  - name: ng-1
    instanceType: c6gn.16xlarge
    minSize: 1
    desiredCapacity: 1
    maxSize: 1
    availabilityZones: ["us-east-1a"]
    volumeSize: 300
    privateNetworking: true
    amiFamily: Ubuntu2204
    efaEnabled: true
    preBootstrapCommands:
      - |
        #!/bin/bash
        exec 1> >(logger -s -t $(basename $0)) 2>&1


 ###############################################################################################
        #                                      Configuration Section
         #
```

```
 ##############################################################################################

        # File System Configuration
        FSX_DNS="<your-fsx-filesystem-dns>" # Needs to be adjusted.
        MOUNT_NAME="<your-mount-name>" # Needs to be adjusted.
        MOUNT_POINT="</your/mount/point>" # Needs to be adjusted.

        # Lustre Tuning Parameters
        LUSTRE_LRU_MAX_AGE=600000
        LUSTRE_MAX_CACHED_MB=64
        LUSTRE_OST_MAX_RPC=32
        LUSTRE_MDC_MAX_RPC=64
        LUSTRE_MDC_MOD_RPC=50

        # File paths
        FUNCTIONS_SCRIPT="/usr/local/bin/lustre_functions.sh"
        TUNINGS_SCRIPT="/usr/local/bin/apply_lustre_tunings.sh"
        SERVICE_FILE="/etc/systemd/system/lustre-tunings.service"

        #EFA
        MIN_EFA_VERSION="2.12.1"

        # Function to check if a command was successful
        check_success() {
            if [ $? -eq 0 ]; then
                echo "SUCCESS: $1"
            else
                echo "FAILED: $1"
                return 1
            fi
        }

        echo "********Starting FSx for Lustre configuration********"

        # 1. Install Lustre client
        if grep -q '^ID=ubuntu' /etc/os-release; then
            echo "Detected Ubuntu, proceeding with Lustre setup..."
            # Add Lustre repository
            sudo wget -O - https://fsx-lustre-client-repo-public-keys.s3.amazonaws.com/
fsx-ubuntu-public-key.asc | sudo gpg --dearmor | sudo tee /usr/share/keyrings/fsx-
ubuntu-public-key.gpg > /dev/null
```

```
            echo "deb [signed-by=/usr/share/keyrings/fsx-ubuntu-public-key.gpg]
 https://fsx-lustre-client-repo.s3.amazonaws.com/ubuntu jammy main" | sudo tee /etc/
apt/sources.list.d/fsxlustreclientrepo.list

            sudo apt-get update
            sudo apt-get install -y lustre-client-modules-$(uname -r)
            sudo apt-get install -y lustre-client
        else
            echo "Not Ubuntu, exiting"
            exit 1
        fi

        check_success "Install Lustre client"

        # Ensure Lustre tools are in the PATH
        export PATH=$PATH:/usr/sbin

        # 2. Apply network and RPC tunings
        echo "********Applying network and RPC tunings********"
        if ! grep -q "options ptlrpc ptlrpcd_per_cpt_max" /etc/modprobe.d/
modprobe.conf; then
            echo "options ptlrpc ptlrpcd_per_cpt_max=64" | sudo tee -a /etc/modprobe.d/
modprobe.conf
            check_success "Set ptlrpcd_per_cpt_max"
        else
            echo "ptlrpcd_per_cpt_max already set in modprobe.conf"
        fi

        if ! grep -q "options ksocklnd credits" /etc/modprobe.d/modprobe.conf; then
            echo "options ksocklnd credits=2560" | sudo tee -a /etc/modprobe.d/
modprobe.conf
            check_success "Set ksocklnd credits"
        else
            echo "ksocklnd credits already set in modprobe.conf"
        fi

        # 3. Load Lustre modules
        manage_lustre_modules() {
            echo "Checking for existing Lustre modules..."
            if lsmod | grep -q lustre; then
                echo "Existing Lustre modules found."

                # Check for mounted Lustre filesystems
                echo "Checking for mounted Lustre filesystems..."
```

```
                if mount | grep -q "type lustre"; then
                    echo "Found mounted Lustre filesystems. Attempting to unmount..."
                    mounted_fs=$(mount | grep "type lustre" | awk '{print $3}')
                    for fs in $mounted_fs; do
                        echo "Unmounting $fs"
                        sudo umount $fs
                        check_success "Unmount filesystem $fs"
                    done
                else
                    echo "No Lustre filesystems mounted."
                fi

                # After unmounting, try to remove modules
                echo "Attempting to remove Lustre modules..."
                sudo lustre_rmmod
                if [ $? -eq 0 ]; then
                    echo "SUCCESS: Removed existing Lustre modules"
                else
                    echo "WARNING: Could not remove Lustre modules. They may still be
in use."
                    echo "Please check for any remaining Lustre processes or mounts."
                    return 1
                fi
            else
                echo "No existing Lustre modules found."
            fi

            echo "Loading Lustre modules..."
            sudo modprobe lustre
            check_success "Load Lustre modules" || exit 1

            echo "Checking loaded Lustre modules:"
            lsmod | grep lustre
        }

        # Managing Lustre kernel modules
        echo "********Managing Lustre kernel modules********"
        manage_lustre_modules

        # 4. Initializing Lustre networking
        echo "********Initializing Lustre networking********"
        sudo lctl network up
        check_success "Initialize Lustre networking" || exit 1
```

```
        # 4.5 EFA Setup and Configuration
        setup_efa() {
            echo "********Starting EFA Setup********"

            # Get interface and version information
            eth_intf="$(/sbin/ip -br -4 a sh | grep $(hostname -i)/ | awk '{print
 $1}')"
            efa_version=$(modinfo efa | awk '/^version:/ {print $2}' | sed 's/[^0-9.]//
g')
            min_efa_version=$MIN_EFA_VERSION

            # Install or verify EFA driver
            if [[ "$(printf '%s\n' "$min_efa_version" "$efa_version" | sort -V | head -
n1)" != "$min_efa_version" ]]; then
                echo "Installing EFA driver..."
                sudo curl -O https://efa-installer.amazonaws.com/aws-efa-
installer-1.37.0.tar.gz
                tar -xf aws-efa-installer-1.37.0.tar.gz && cd aws-efa-installer

                # Install dependencies
                sudo apt-get update && apt-get upgrade -y
                sudo apt install -y pciutils environment-modules libnl-3-dev libnl-
route-3-200 libnl-route-3-dev dkms

                # Install EFA
                sudo ./efa_installer.sh -y
                modinfo efa
            else
                echo "Using existing EFA driver version $efa_version"
            fi
        }

        configure_efa_network() {
            echo "********Configuring EFA Network********"

            # Load required modules
            echo "Loading network modules..."
            sudo /sbin/modprobe lnet
            sudo /sbin/modprobe kefalnd ipif_name="$eth_intf"
            sudo /sbin/modprobe ksocklnd

            # Initialize LNet
            echo "Initializing LNet..."
            sudo lnetctl lnet configure
```

```
            # Configure TCP interface
            echo "Configuring TCP interface..."
            sudo lnetctl net del --net tcp 2> /dev/null
            sudo lnetctl net add --net tcp --if $eth_intf

            # For P5 instance type which supports 32 network cards,
            # by default add 8 EFA interfaces selecting every 4th device (1 per PCI
bus)
            echo "Configuring EFA interface(s)..."
            instance_type="$(ec2-metadata --instance-type | awk '{ print $2 }')"
            num_efa_devices="$(ls -1 /sys/class/infiniband | wc -l)"
            echo "Found $num_efa_devices available EFA device(s)"

            if [[ "$instance_type" == "p5.48xlarge" || "$instance_type" ==
"p5e.48xlarge" ]]; then
                # P5 instance configuration
                for intf in $(ls -1 /sys/class/infiniband | awk 'NR % 4 == 1'); do
                    sudo lnetctl net add --net efa --if $intf --peer-credits 32
                done
            else
                # Standard configuration
                # Other instances: Configure 2 EFA interfaces by default if the
instance supports multiple network cards,
                # or 1 interface for single network card instances
                # Can be modified to add more interfaces if instance type supports it
                sudo lnetctl net add --net efa --if $(ls -1 /sys/class/infiniband |
head -n1) --peer-credits 32
                if [[ $num_efa_devices -gt 1 ]]; then
                    sudo lnetctl net add --net efa --if $(ls -1 /sys/class/infiniband |
tail -n1) --peer-credits 32
                fi
            fi

            # Configure discovery and UDSP
            echo "Setting up discovery and UDSP rules..."
            sudo lnetctl set discovery 1
            sudo lnetctl udsp add --src efa --priority 0
            sudo /sbin/modprobe lustre

            # Verify configuration
            echo "Verifying EFA network configuration..."
            sudo lnetctl net show
            echo "Added $(sudo lnetctl net show | grep -c '@efa') EFA interface(s)"
```

```
        }

        # Main execution
        setup_efa
        configure_efa_network

        # 5. Mount FSx filesystem
        if [ ! -z "$FSX_DNS" ] && [ ! -z "$MOUNT_NAME" ]; then
            echo "********Creating mount point********"
            sudo mkdir -p $MOUNT_POINT
            check_success "Create mount point"

            echo "Mounting FSx filesystem..."
            sudo mount -t lustre ${FSX_DNS}@tcp:/${MOUNT_NAME} ${MOUNT_POINT}
            check_success "Mount FSx filesystem"
        else
            echo "Skipping FSx mount as DNS or mount name is not provided"
        fi

        # 6. Applying Lustre performance tunings
        echo "********Applying Lustre performance tunings********"

        # Get number of CPUs
        NUM_CPUS=$(nproc)

        # Calculate LRU size (100 * number of CPUs)
        LRU_SIZE=$((100 * NUM_CPUS))

        #Apply LRU tunings
        echo "Apply LRU tunings"
        sudo lctl set_param ldlm.namespaces.*.lru_max_age=${LUSTRE_LRU_MAX_AGE}
        check_success "Set lru_max_age"
        sudo lctl set_param ldlm.namespaces.*.lru_size=$LRU_SIZE
        check_success "Set lru_size"

        # Client Cache Control
        sudo lctl set_param llite.*.max_cached_mb=${LUSTRE_MAX_CACHED_MB}
        check_success "Set max_cached_mb"

        # RPC Controls
        sudo lctl set_param osc.*OST*.max_rpcs_in_flight=${LUSTRE_OST_MAX_RPC}
        check_success "Set OST max_rpcs_in_flight"

        sudo lctl set_param mdc.*.max_rpcs_in_flight=${LUSTRE_MDC_MAX_RPC}
```

```
        check_success "Set MDC max_rpcs_in_flight"

        sudo lctl set_param mdc.*.max_mod_rpcs_in_flight=${LUSTRE_MDC_MOD_RPC}
        check_success "Set MDC max_mod_rpcs_in_flight"

        # 7. Verify all tunings
        echo "********Verifying all tunings********"

        # Function to verify parameter value
        verify_param() {
            local param=$1
            local expected=$2
            local actual=$3

            if [ "$actual" == "$expected" ]; then
                echo "SUCCESS: $param is correctly set to $expected"
            else
                echo "WARNING: $param is set to $actual (expected $expected)"
            fi
        }

        echo "Verifying all parameters:"

        # LRU tunings
        actual_lru_max_age=$(lctl get_param -n ldlm.namespaces.*.lru_max_age | head -1)
        verify_param "lru_max_age" "600000" "$actual_lru_max_age"

        actual_lru_size=$(lctl get_param -n ldlm.namespaces.*.lru_size | head -1)
        verify_param "lru_size" "$LRU_SIZE" "$actual_lru_size"

        # Client Cache
        actual_max_cached_mb=$(lctl get_param -n llite.*.max_cached_mb | grep
"max_cached_mb:" | awk '{print $2}')
        verify_param "max_cached_mb" "64" "$actual_max_cached_mb"

        # RPC Controls
        actual_ost_rpcs=$(lctl get_param -n osc.*OST*.max_rpcs_in_flight | head -1)
        verify_param "OST max_rpcs_in_flight" "32" "$actual_ost_rpcs"

        actual_mdc_rpcs=$(lctl get_param -n mdc.*.max_rpcs_in_flight | head -1)
        verify_param "MDC max_rpcs_in_flight" "64" "$actual_mdc_rpcs"

        actual_mdc_mod_rpcs=$(lctl get_param -n mdc.*.max_mod_rpcs_in_flight | head -1)
        verify_param "MDC max_mod_rpcs_in_flight" "50" "$actual_mdc_mod_rpcs"
```

```
        # Network and RPC configurations from modprobe.conf
        actual_ptlrpc=$(grep "ptlrpc ptlrpcd_per_cpt_max" /etc/modprobe.d/modprobe.conf
| awk '{print $3}')
        verify_param "ptlrpcd_per_cpt_max" "ptlrpcd_per_cpt_max=64" "$actual_ptlrpc"

        actual_ksocklnd=$(grep "ksocklnd credits" /etc/modprobe.d/modprobe.conf | awk
'{print $3}')
        verify_param "ksocklnd credits" "credits=2560" "$actual_ksocklnd"

        # 8. Setup persistence
        setup_persistence() {
            # Create functions file
            cat << EOF > $FUNCTIONS_SCRIPT
        #!/bin/bash

        apply_lustre_tunings() {
            local NUM_CPUS=\$(nproc)
            local LRU_SIZE=\$((100 * NUM_CPUS))

            echo "Applying Lustre performance tunings..."
            lctl set_param ldlm.namespaces.*.lru_max_age=$LUSTRE_LRU_MAX_AGE
            lctl set_param ldlm.namespaces.*.lru_size=\$LRU_SIZE
            lctl set_param llite.*.max_cached_mb=$LUSTRE_MAX_CACHED_MB
            lctl set_param osc.*OST*.max_rpcs_in_flight=$LUSTRE_OST_MAX_RPC
            lctl set_param mdc.*.max_rpcs_in_flight=$LUSTRE_MDC_MAX_RPC
            lctl set_param mdc.*.max_mod_rpcs_in_flight=$LUSTRE_MDC_MOD_RPC
        }
        EOF

            # Create tuning script
            cat << EOF > $TUNINGS_SCRIPT
        #!/bin/bash
        exec 1> >(logger -s -t \$(basename \$0)) 2>&1

        source $FUNCTIONS_SCRIPT

        # Function to check if Lustre is mounted
        is_lustre_mounted() {
            mount | grep -q "type lustre"
        }

        # Function to mount Lustre
        mount_lustre() {
```

```
            echo "Mounting Lustre filesystem..."
            mkdir -p $MOUNT_POINT
            mount -t lustre ${FSX_DNS}@tcp:/${MOUNT_NAME} $MOUNT_POINT
            return \$?
        }

        # Main execution
        # Try to mount if not already mounted
        if ! is_lustre_mounted; then
            echo "Lustre filesystem not mounted, attempting to mount..."
            mount_lustre
        fi

        # Wait for successful mount (up to 5 minutes)
        for i in {1..30}; do
            if is_lustre_mounted; then
                echo "Lustre filesystem mounted, applying tunings..."
                apply_lustre_tunings
                exit 0
            fi
            echo "Waiting for Lustre filesystem to be mounted... (attempt $i/30)"
            sleep 10
        done

        echo "Timeout waiting for Lustre filesystem mount"
        exit 1
        EOF

        # Create systemd service

        # Create systemd directory if it doesn't exist
        sudo mkdir -p /etc/systemd/system/

            # Create service file directly for Ubuntu
            cat << EOF > $SERVICE_FILE
        [Unit]
        Description=Apply Lustre Performance Tunings
        After=network.target remote-fs.target

        [Service]
        Type=oneshot
        ExecStart=/bin/bash -c 'source $FUNCTIONS_SCRIPT && $TUNINGS_SCRIPT'
        RemainAfterExit=yes
```

```
        [Install]
        WantedBy=multi-user.target
        EOF


            # Make scripts executable and enable service
            sudo chmod +x $FUNCTIONS_SCRIPT
            sudo chmod +x $TUNINGS_SCRIPT
            systemctl enable lustre-tunings.service
            systemctl start lustre-tunings.service
        }

        echo "********Setting up persistent tuning********"
        setup_persistence

        echo "FSx for Lustre configuration completed."
```

(Optional) Schritt 3. EFA-Einrichtung überprüfen

SSH inherhalb Knoten:

```
# Get instance ID from EKS console or {aws} CLI
ssh -i /path/to/your-key.pem ec2-user@<node-internal-ip>
```

EFA-Konfiguration überprüfen:

```
sudo lnetctl net show
```

Einrichtungsprotokolle überprüfen:

```
sudo cat /var/log/cloud-init-output.log
```

Hier ist ein Beispiel für die erwartete Ausgabe für `lnetctl net show`:

```
net:
    - net type: tcp
      ...
    - net type: efa
      local NI(s):
        - nid: xxx.xxx.xxx.xxx@efa
          status: up
```

Beispiel-Bereitstellungen

a. claim.yaml erstellen

```
#claim.yaml

apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: fsx-claim-efa
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: ""
  resources:
    requests:
      storage: 4800Gi
  volumeName: fsx-pv
```

Anspruch anwenden:

```
kubectl apply -f claim.yaml
```

b. pv.yaml erstellen

Aktualisieren Sie das`<replaceable-placeholders>`:

```
#pv.yaml

apiVersion: v1
kind: PersistentVolume
metadata:
  name: fsx-pv
spec:
  capacity:
    storage: 4800Gi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteMany
  mountOptions:
    - flock
  persistentVolumeReclaimPolicy: Recycle
```

```
  csi:
    driver: fsx.csi.aws.com
    volumeHandle: fs-<1234567890abcdef0>
    volumeAttributes:
      dnsname: fs-<1234567890abcdef0>.fsx.us-east-1.amazonaws.com
      mountname: <abcdef01>
```

Persistentes Volumen anwenden:

```
kubectl apply -f pv.yaml
```

c. pod.yaml erstellen

```
#pod.yaml

apiVersion: v1
kind: Pod
metadata:
  name: fsx-efa-app
spec:
  containers:
  - name: app
    image: amazonlinux:2
    command: ["/bin/sh"]
    args: ["-c", "while true; do dd if=/dev/urandom bs=100M count=20 > data/test_file;
 sleep 10; done"]
    resources:
      requests:
        vpc.amazonaws.com/efa: 1
      limits:
        vpc.amazonaws.com/efa: 1
    volumeMounts:
    - name: persistent-storage
      mountPath: /data
  volumes:
  - name: persistent-storage
    persistentVolumeClaim:
      claimName: fsx-claim-efa
```

Pod anwenden:

```
kubectl apply -f pod.yaml
```

Zusätzliche Überprüfungsbefehle

Überprüfung der Einbindung des Pods und des Schreibvorgangs in das Dateisystem:

```
kubectl exec -ti fsx-efa-app -- df -h | grep data
# Expected output:
# <192.0.2.0>@tcp:/<abcdef01>  4.5T  1.2G  4.5T   1% /data

kubectl exec -ti fsx-efa-app -- ls /data
# Expected output:
# test_file
```

SSH-Verbindung zum Knoten herstellen, um zu überprüfen, ob der Datenverkehr über EFA läuft:

```
sudo lnetctl net show -v
```

Die erwartete Ausgabe zeigt EFA-Schnittstellen mit Datenverkehrsstatistiken an.

Ähnliche Informationen

- the section called “Treiber bereitstellen”
- the section called “Optimieren (nicht-EFA)”
- Amazon FSx für Lustre Performance
- Elastic Fabric Adapter

Optimieren Sie die Leistung von Amazon FSx for Lustre auf Knoten (ohne EFA)

Sie können die Leistung von Amazon FSx for Lustre optimieren, indem Sie während der Knoteninitialisierung mithilfe von Benutzerdaten der Startvorlage Optimierungsparameter anwenden.

> **Note**
>
> - Informationen zur Erstellung und Bereitstellung des CSI-Treibers FSx für Lustre finden Sie unter. the section called “Treiber bereitstellen” Informationen zur Leistungsoptimierung mit EFA-fähigen Knoten finden Sie unter the section called “Optimieren (EFA)”.

Warum Benutzerdaten von Startvorlagen verwenden?

- Wendet Optimierungen automatisch während der Knoten-Initialisierung an.
- Gewährleistet eine konsistente Konfiguration auf allen Knoten.
- Macht die manuelle Knotenkonfiguration überflüssig.

Übersicht über das Beispiel-Skript

Das in diesem Thema definierte Beispiel-Skript führt die folgenden Vorgänge aus:

`# 1. Install Lustre client`

- Erkennt automatisch Ihre Amazon Linux (AL) Betriebssystemversion.
- Installation des entsprechenden Lustre-Client-Pakets.

`# 2. Apply network and RPC tunings`

- Festlegen von `ptlrpcd_per_cpt_max=64` für die parallele RPC-Verarbeitung.
- Konfiguriert `ksocklnd credits=2560` zur Optimierung der Netzwerkpuffer.

`# 3. Load Lustre modules`

- Sichere Entfernung vorhandener Lustre-Module, falls vorhanden.
- Entfernt vorhandene eingebundene Dateisysteme.
- Lädt aktuelle Lustre-Module.

`# 4. Lustre Network Initialization`

- Initialisiert die Lustre-Netzwerkkonfiguration.
- Richtet die erforderlichen Netzwerkparameter ein.

`# 5. Mount FSx filesystem`

- In diesem Abschnitt müssen Sie die Werte für Ihre Umgebung anpassen.

 # 6. Apply tunings

- LRU-Abstimmungen (Lock Resource Unit):
  - `lru_max_age=600000`
  - `lru_size` berechnet basierend auf der CPU-Anzahl
- Client-Cache-Steuerung: `max_cached_mb=64`
- RPC-Steuerungen
  - OST `max_rpcs_in_flight=32`
  - MDC `max_rpcs_in_flight=64`
  - MDC `max_mod_rpcs_in_flight=50`

 # 7. Verify tunings

- Überprüft alle angewendeten Abstimmungen.
- Meldet Erfolg oder Warnung für jeden Parameter.

 # 8. Setup persistence

- Auch in diesem Abschnitt müssen Sie die Werte für Ihre Umgebung anpassen.
- Erkennt automatisch Ihre Betriebssystemversion (AL2023), um zu bestimmen, welcher `Systemd` Service angewendet werden soll.
- Das System startet.
- `Systemd` startet den `lustre-tunings`-Service (aufgrund von `WantedBy=multi-user.target`).
- Service führt `apply_lustre_tunings.sh` aus, das:
  - Überprüft, ob das Dateisystem eingebunden ist.
  - Bindet das Dateisystem, falls es noch nicht eingebunden ist.
  - Wartet auf erfolgreiche Einbindung (bis zu fünf Minuten).
  - Wendet nach erfolgreicher Einbindung die Abstimmungsparameter an.
- Einstellungen bleiben bis zum Neustart aktiv.
- Service wird nach Abschluss des Skripts beendet.
  - Systemd kennzeichnet den Service als „aktiv (beendet)“.
- Der Vorgang wiederholt sich beim nächsten Neustart.

Erstellen einer Startvorlage

1. Öffnen Sie die EC2 Amazon-Konsole unter https://console.aws.amazon.com/ec2/.
2. Wählen Sie Startvorlagen aus.
3. Wählen Sie Startvorlage erstellen.
4. Suchen Sie unter Erweiterte Details den Abschnitt Benutzerdaten.
5. Fügen Sie das folgende Skript ein und aktualisieren Sie es nach Bedarf.

> **Important**
>
> Passen Sie diese Werte sowohl im Abschnitt `# 5. Mount FSx filesystem` als auch in der `setup_persistence()`-Funktion von `apply_lustre_tunings.sh` im Abschnitt `# 8. Setup persistence` an Ihre Umgebung an:
>
> ```
> FSX_DNS="<your-fsx-filesystem-dns>" # Needs to be adjusted.
> MOUNT_NAME="<your-mount-name>" # Needs to be adjusted.
> MOUNT_POINT="</your/mount/point>" # Needs to be adjusted.
> ```

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="
--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"
#!/bin/bash
exec 1> >(logger -s -t $(basename $0)) 2>&1
# Function definitions
check_success() {
    if [ $? -eq 0 ]; then
        echo "SUCCESS: $1"
    else
        echo "FAILED: $1"
        return 1
    fi
}
apply_tunings() {
    local NUM_CPUS=$(nproc)
    local LRU_SIZE=$((100 * NUM_CPUS))
    local params=(
        "ldlm.namespaces.*.lru_max_age=600000"
```

```
        "ldlm.namespaces.*.lru_size=$LRU_SIZE"
        "llite.*.max_cached_mb=64"
        "osc.*OST*.max_rpcs_in_flight=32"
        "mdc.*.max_rpcs_in_flight=64"
        "mdc.*.max_mod_rpcs_in_flight=50"
    )
    for param in "${params[@]}"; do
        lctl set_param $param
        check_success "Set ${param%%=*}"
    done
}
verify_param() {
    local param=$1
    local expected=$2
    local actual=$3

    if [ "$actual" == "$expected" ]; then
        echo "SUCCESS: $param is correctly set to $expected"
    else
        echo "WARNING: $param is set to $actual (expected $expected)"
    fi
}
verify_tunings() {
    local NUM_CPUS=$(nproc)
    local LRU_SIZE=$((100 * NUM_CPUS))
    local params=(
        "ldlm.namespaces.*.lru_max_age:600000"
        "ldlm.namespaces.*.lru_size:$LRU_SIZE"
        "llite.*.max_cached_mb:64"
        "osc.*OST*.max_rpcs_in_flight:32"
        "mdc.*.max_rpcs_in_flight:64"
        "mdc.*.max_mod_rpcs_in_flight:50"
    )
    echo "Verifying all parameters:"
    for param in "${params[@]}"; do
        name="${param%%:*}"
        expected="${param#*:}"
        actual=$(lctl get_param -n $name | head -1)
        verify_param "${name##*.}" "$expected" "$actual"
    done
}
setup_persistence() {
    # Create functions file
    cat << 'EOF' > /usr/local/bin/lustre_functions.sh
```

```
#!/bin/bash
apply_lustre_tunings() {
    local NUM_CPUS=$(nproc)
    local LRU_SIZE=$((100 * NUM_CPUS))

    echo "Applying Lustre performance tunings..."
    lctl set_param ldlm.namespaces.*.lru_max_age=600000
    lctl set_param ldlm.namespaces.*.lru_size=$LRU_SIZE
    lctl set_param llite.*.max_cached_mb=64
    lctl set_param osc.*OST*.max_rpcs_in_flight=32
    lctl set_param mdc.*.max_rpcs_in_flight=64
    lctl set_param mdc.*.max_mod_rpcs_in_flight=50
}
EOF
    # Create tuning script
    cat << 'EOF' > /usr/local/bin/apply_lustre_tunings.sh
#!/bin/bash
exec 1> >(logger -s -t $(basename $0)) 2>&1
# Source the functions
source /usr/local/bin/lustre_functions.sh
# FSx details
FSX_DNS="<your-fsx-filesystem-dns>" # Needs to be adjusted.
MOUNT_NAME="<your-mount-name>" # Needs to be adjusted.
MOUNT_POINT="</your/mount/point>" # Needs to be adjusted.
# Function to check if Lustre is mounted
is_lustre_mounted() {
    mount | grep -q "type lustre"
}
# Function to mount Lustre
mount_lustre() {
    echo "Mounting Lustre filesystem..."
    mkdir -p ${MOUNT_POINT}
    mount -t lustre ${FSX_DNS}@tcp:/${MOUNT_NAME} ${MOUNT_POINT}
    return $?
}
# Main execution
# Try to mount if not already mounted
if ! is_lustre_mounted; then
    echo "Lustre filesystem not mounted, attempting to mount..."
    mount_lustre
fi
# Wait for successful mount (up to 5 minutes)
for i in {1..30}; do
    if is_lustre_mounted; then
```

```
        echo "Lustre filesystem mounted, applying tunings..."
        apply_lustre_tunings
        exit 0
    fi
    echo "Waiting for Lustre filesystem to be mounted... (attempt $i/30)"
    sleep 10
done
echo "Timeout waiting for Lustre filesystem mount"
exit 1
EOF
    # Create systemd service
    cat << 'EOF' > /etc/systemd/system/lustre-tunings.service
[Unit]
Description=Apply Lustre Performance Tunings
After=network.target remote-fs.target
StartLimitIntervalSec=0
[Service]
Type=oneshot
ExecStart=/usr/local/bin/apply_lustre_tunings.sh
RemainAfterExit=yes
Restart=on-failure
RestartSec=30
[Install]
WantedBy=multi-user.target
EOF
    chmod +x /usr/local/bin/lustre_functions.sh
    chmod +x /usr/local/bin/apply_lustre_tunings.sh
    systemctl enable lustre-tunings.service
    systemctl start lustre-tunings.service
}
echo "Starting FSx for Lustre configuration..."
# 1. Install Lustre client
if grep -q 'VERSION="2"' /etc/os-release; then
    amazon-linux-extras install -y lustre
elif grep -q 'VERSION="2023"' /etc/os-release; then
    dnf install -y lustre-client
fi
check_success "Install Lustre client"
# 2. Apply network and RPC tunings
export PATH=$PATH:/usr/sbin
echo "Applying network and RPC tunings..."
if ! grep -q "options ptlrpc ptlrpcd_per_cpt_max" /etc/modprobe.d/modprobe.conf; then
    echo "options ptlrpc ptlrpcd_per_cpt_max=64" | tee -a /etc/modprobe.d/
modprobe.conf
```

```
    echo "options ksocklnd credits=2560" | tee -a /etc/modprobe.d/modprobe.conf
fi
# 3. Load Lustre modules
modprobe lustre
check_success "Load Lustre modules" || exit 1
# 4. Lustre Network Initialization
lctl network up
check_success "Initialize Lustre networking" || exit 1
# 5. Mount FSx filesystem
FSX_DNS="<your-fsx-filesystem-dns>" # Needs to be adjusted.
MOUNT_NAME="<your-mount-name>" # Needs to be adjusted.
MOUNT_POINT="</your/mount/point>" # Needs to be adjusted.
if [ ! -z "$FSX_DNS" ] && [ ! -z "$MOUNT_NAME" ]; then
    mkdir -p $MOUNT_POINT
    mount -t lustre ${FSX_DNS}@tcp:/${MOUNT_NAME} ${MOUNT_POINT}
    check_success "Mount FSx filesystem"
fi
# 6. Apply tunings
apply_tunings
# 7. Verify tunings
verify_tunings
# 8. Setup persistence
setup_persistence
echo "FSx for Lustre configuration completed."
--==MYBOUNDARY==--
```

6. Wählen Sie beim Erstellen von Amazon-EKS-Knotengruppen diese Startvorlage aus. Weitere Informationen finden Sie unter the section called “Create”.

Ähnliche Informationen

- the section called “Treiber bereitstellen”
- the section called “Optimieren (EFA)”
- Amazon FSx für Lustre Performance

Verwendung von leistungsstarkem App-Speicher mit FSx für NetApp ONTAP

Der NetApp Trident bietet dynamische Speicher-Orchestrierung mithilfe eines Container Storage Interface (CSI)-kompatiblen Treibers. Das ermöglicht es Amazon-EKS-Clustern, den Lebenszyklus

persistenter Volumes (PVs) zu verwalten, die von Amazon-FSx-für-NetApp-ONTAP Dateisystemen gestützt werden. Beachten Sie, dass der CSI-Treiber für Amazon FSx für NetApp ONTAP nicht mit Amazon EKS Hybrid Nodes kompatibel ist. Informationen zu den ersten Schritten finden Sie unter Verwendung von Trident mit Amazon FSx für NetApp ONTAP in der NetApp-Trident-Dokumentation.

Amazon FSx für NetApp ONTAP ist ein Speicher-Service, mit dem Sie vollständig verwaltete ONTAP-Dateisysteme in der Cloud starten und ausführen können. ONTAP ist die Dateisystem-Technologie von NetApp, die eine weit verbreitete Reihe von Datenzugriffs- und Datenverwaltungs-Funktionen bietet. FSx für ONTAP bietet die Feature, Leistung und APIs von On-Premises-NetApp-Dateisysteme mit der Agilität, Skalierbarkeit und Einfachheit eines vollständig verwalteten AWS-Services. Weitere Informationen finden Sie im FSx für ONTAP-Benutzerhandbuch.

Important

Wenn Sie Amazon FSx für NetApp ONTAP zusammen mit dem Amazon-EBS-CSI-Treiber verwenden, um EBS-Volumes bereitzustellen, müssen Sie in der `multipath.conf`-Datei angeben, dass keine EBS-Geräte verwendet werden sollen. Unterstützte Methoden finden Sie unter Blacklist der Konfigurationsdatei. Ein Beispiel.

```
defaults {
       user_friendly_names yes
       find_multipaths no
     }
     blacklist {
       device {
         vendor "NVME"
         product "Amazon Elastic Block Store"
       }
     }
```

Datenspeicherung mit Amazon FSx für OpenZFS verwenden

Amazon FSx für OpenZFS ist ein vollständig verwalteter Dateispeicherservice, der das Verschieben von Daten zu AWS von On-Premises-ZFS- oder anderen Linux-basierten Dateiservern erleichtert. Sie können dies tun, ohne Ihren Anwendungscode oder die Art und Weise, wie Sie Daten verwalten, zu ändern. Es bietet einen äußerst zuverlässigen, skalierbaren, effizienten und featurereichen Dateispeicher, der auf dem Open-Source-Dateisystem OpenZFS basiert. Es kombiniert diese

Funktionen mit der Agilität, Skalierbarkeit und Einfachheit eines vollständig verwalteten AWS-Services. Weitere Informationen finden Sie im Benutzerhandbuch für Amazon FSx für OpenZFS.

Der FSx für OpenZFS Container Storage Interface (CSI)-Treiber bietet eine CSI-Schnittstelle, die es Amazon-EKS-Clustern ermöglicht, den Lebenszyklus von FSx für OpenZFS-Volumes zu verwalten. Beachten Sie, dass der Amazon FSx für OpenZFS-CSI-Treiber nicht mit Amazon EKS Hybrid Nodes kompatibel ist. Informationen zum Bereitstellen des CSI-Treibers für FSx für OpenZFS in Ihrem Amazon-EKS-Cluster finden Sie unter aws-fsx-openzfs-csi-driver auf GitHub.

Minimierung der Latenz mit Amazon File Cache

Amazon File Cache ist ein vollständig verwalteter Hochgeschwindigkeits-Cache in AWS, der zur Verarbeitung von Dateidaten verwendet wird, unabhängig davon, wo die Daten gespeichert sind. Amazon File Cache lädt Daten automatisch in den Cache, wenn sie zum ersten Mal abgerufen werden, und gibt sie frei, wenn sie nicht verwendet werden. Weitere Informationen finden Sie in Benutzerhandbuch zu Amazon File Cache.

Der Container Storage Interface (CSI)-Treiber für Amazon File Cache bietet eine CSI-Schnittstelle, die es Amazon-EKS-Clustern ermöglicht, den Lebenszyklus von Amazon-Datei-Caches zu verwalten. Beachten Sie, dass der Amazon-File-Cache-CSI-Treiber nicht mit Amazon EKS Hybrid Nodes kompatibel ist. Informationen zur Bereitstellung des Amazon-File-Cache-CSI-Treibers in Ihrem Amazon-EKS-Cluster finden Sie unter aws-file-cache-csi-driver auf GitHub.

Zugriff auf Amazon-S3-Objekte mit dem CSI-Treiber für Mountpoint für Amazon S3

Mit dem Mountpoint für Amazon S3 Container Storage Interface (CSI)-Treiber können Ihre Kubernetes-Anwendungen über eine Dateisystemschnittstelle auf Amazon-S3-Objekte zugreifen und so einen hohen Gesamtdurchsatz erzielen, ohne dass Änderungen am Anwendungscode erforderlich sind. Der CSI-Treiber basiert auf Mountpoint für Amazon S3 und stellt einen Amazon-S3-Bucket als Volume dar, auf das Container in Amazon EKS und selbstverwalteten Kubernetes-Clustern zugreifen können.

Überlegungen

- Der CSI-Treiber für Mountpoint für Amazon S3 ist derzeit nicht mit Windows-basierten Container-Images kompatibel.

- Der CSI-Treiber für Mountpoint für Amazon S3 ist derzeit nicht mit Amazon EKS Hybrid Nodes kompatibel.
- AWS Fargate wird vom CSI-Treiber für Mountpoint für Amazon S3 nicht unterstützt. Container, die in Amazon EC2 ausgeführt werden (entweder mit Amazon EKS oder einer benutzerdefinierten Kubernetes-Installation), werden unterstützt.
- Der CSI-Treiber für Mountpoint für Amazon S3 unterstützt nur die statische Bereitstellung. Dynamische Bereitstellung oder die Erstellung neuer Buckets wird nicht unterstützt.

> **Note**
>
> Statische Bereitstellung bezieht sich auf die Verwendung eines vorhandenen Amazon-S3-Buckets, der als `bucketName` in `volumeAttributes` des `PersistentVolume`-Objekts angegeben ist. Weitere Informationen finden Sie unter Statische Bereitstellung auf GitHub.

- Volumes, die mit dem CSI-Treiber für Mountpoint für Amazon S3 bereitgestellt wurden, unterstützen nicht alle Feature des POSIX-Dateisystems. Weitere Informationen zum Verhalten des Dateisystems finden Sie unter Dateisystemverhalten von Mountpoint für Amazon S3 auf GitHub.

Einzelheiten zum Bereitstellen des Treibers finden Sie unter the section called “Treiber bereitstellen”. Informationen zum Entfernen des Treibers finden Sie unter the section called “Treiber entfernen”.

Bereitstellung des CSI-Treibers für Mountpoint für Amazon S3

Mit dem Mountpoint für Amazon S3 Container Storage Interface (CSI)-Treiber können Ihre Kubernetes-Anwendungen über eine Dateisystemschnittstelle auf Amazon-S3-Objekte zugreifen und so einen hohen Gesamtdurchsatz erzielen, ohne dass Änderungen am Anwendungscode erforderlich sind.

Dieses Verfahren zeigt Ihnen, wie Sie den CSI-Amazon-EKS-Treiber für Mountpoint für Amazon S3 bereitstellen. Lesen Sie sich die Hinweise durch, bevor Sie fortfahren.

Voraussetzungen

- Ein vorhandener AWS Identity and Access Management (IAM) OpenID Connect (OIDC) -Anbieter für Ihren Cluster. Informationen zum Feststellen, ob Sie bereits über einen verfügen oder einen erstellen müssen, finden Sie unter the section called “IAM-OIDC-Anbieter”.
- Version 2.12.3 oder höher der AWS CLI, die auf Ihrem Gerät installiert und konfiguriert ist, oder. AWS CloudShell

- Das `kubectl`-Befehlszeilen-Tool ist auf Ihrem Gerät oder in der AWS CloudShell installiert. Die Version kann mit der Kubernetes-Version Ihres Clusters identisch sein oder bis zu einer Nebenversion älter oder neuer sein. Wenn Ihre Clusterversion beispielsweise `1.29` ist, können Sie `kubectl`-Version `1.28`, `1.29`, oder `1.30` damit verwenden. Informationen zum Installieren oder Aktualisieren von `kubectl` finden Sie unter the section called “kubectl und eksctl einrichten”.

Schritt 1: Erstellen einer IAM-Richtlinie

Der CSI-Treiber für Mountpoint für Amazon S3 erfordert Amazon-S3-Berechtigungen, um mit Ihrem Dateisystem zu interagieren. In diesem Abschnitt wird erläutert, wie Sie eine IAM-Richtlinie erstellen, die die erforderlichen Berechtigungen gewährt.

Die folgende Beispielrichtlinie richtet sich nach den IAM-Berechtigungsempfehlungen für Mountpoint. Alternativ können Sie die AWS verwaltete Richtlinie AmazonS3 verwendenFullAccess, aber diese verwaltete Richtlinie gewährt mehr Berechtigungen, als für Mountpoint benötigt werden.

Weitere Informationen zu den empfohlenen Berechtigungen für Mountpoint finden Sie unter Mountpoint IAM-Berechtigungen für. GitHub

1. Öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im linken Navigationsbereich die Option Policies aus.
3. Wählen Sie auf der Seite Richtlinien die Option Richtlinie erstellen.
4. Wählen Sie für Richtlinien-Editor die Option JSON aus.
5. Kopieren Sie unter Richtlinien-Editor Folgendes und fügen Sie es ein:

> **Important**
>
> Ersetzen Sie `amzn-s3-demo-bucket1` durch den Namen Ihres eigenen Amazon-S3-Bucket.

```
{
   "Version":"2012-10-17",
   "Statement": [
        {
            "Sid": "MountpointFullBucketAccess",
            "Effect": "Allow",
            "Action": [
```

```
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1"
            ]
        },
        {
            "Sid": "MountpointFullObjectAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject",
                "s3:AbortMultipartUpload",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1/*"
            ]
        }
    ]
}
```

Verzeichnis-Buckets, die mit der Speicherklasse für Amazon S3 Express One Zone eingeführt wurden, verwenden einen anderen Authentifizierungs-Mechanismus als allgemeine Buckets. Anstatt `s3:*`-Aktionen zu verwenden, sollten Sie die `s3express:CreateSession`-Aktion verwenden. Informationen zu Verzeichnis-Buckets finden Sie unter Verzeichnis-Buckets im Amazon-S3-Benutzerhandbuch.

Nachfolgend finden Sie ein Beispiel für eine Richtlinie mit geringsten Berechtigungen, die Sie für einen Verzeichnis-Bucket verwenden würden.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3express:CreateSession",
            "Resource": "arn:aws:s3express:us-west-2:111122223333:bucket/amzn-s3-
demo-bucket1--usw2-az1--x-s3"
        }
    ]
}
```

6. Wählen Sie Weiter aus.
7. Geben Sie Ihrer Richtlinie auf der Seite Überprüfen und erstellen einen Namen. In dieser Beispielanleitung wird der Name `AmazonS3CSIDriverPolicy` verwendet.
8. Wählen Sie Richtlinie erstellen aus.

Schritt 2: Erstellen einer IAM-Rolle

Der CSI-Treiber für Mountpoint für Amazon S3 erfordert Amazon-S3-Berechtigungen, um mit Ihrem Dateisystem zu interagieren. In diesem Abschnitt wird erläutert, wie Sie eine IAM-Rolle erstellen, mit der Sie diese Berechtigungen delegieren. Um diese Rolle zu erstellen, können Sie eines der folgenden Tools verwenden:

- the section called “eksctl”
- the section called “AWS-Managementkonsole”
- the section called “AWS CLI”

> **Note**
>
> Die IAM-Richtlinie `AmazonS3CSIDriverPolicy` wurde im vorherigen Abschnitt erstellt.

eksctl

So erstellen Sie Ihre IAM-Rolle für CSI-Treiber für Mountpoint für Amazon S3 mit **`eksctl`**

Führen Sie die folgenden Befehle aus, um die IAM-Rolle und das Kubernetes-Servicekonto zu erstellen. Über diese Befehle wird außerdem die `AmazonS3CSIDriverPolicy` IAM-Richtlinie an die Rolle angefügt, das Kubernetes-Servicekonto (`s3-csi-controller-sa`) mit dem Amazon-Ressourcennamen (ARN) der IAM-Rolle versehen und der Name des Kubernetes-Servicekontos zur Vertrauensrichtlinie für die IAM-Rolle hinzugefügt.

```
CLUSTER_NAME=my-cluster
REGION=region-code
ROLE_NAME=AmazonEKS_S3_CSI_DriverRole
POLICY_ARN=AmazonEKS_S3_CSI_DriverRole_ARN
eksctl create iamserviceaccount \
    --name s3-csi-driver-sa \
    --namespace kube-system \
```

```
    --cluster $CLUSTER_NAME \
    --attach-policy-arn $POLICY_ARN \
    --approve \
    --role-name $ROLE_NAME \
    --region $REGION \
    --role-only
```

AWS-Managementkonsole

1. Öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im linken Navigationsbereich Roles aus.
3. Klicken Sie auf der Seite Roles (Rollen) auf Create role (Rolle erstellen).
4. Gehen Sie auf der Seite Select trusted entity (Vertrauenswürdige Entität auswählen) wie folgt vor:
   a. Wählen Sie im Abschnitt Trusted entity type (Typ der vertrauenswürdigen Entität) die Option Web identity (Web-Identität) aus.
   b. Wählen Sie für Identity provider (Identitätsanbieter) die Option OpenID Connect provider URL (OpenID-Connect-Anbieter-URL) für Ihren Cluster aus (wie unter Overview (Übersicht) in Amazon EKS gezeigt).

      Wenn keine angezeigt URLs werden, überprüfen Sie die Voraussetzungen.
   c. Wählen Sie für Audience (Zielgruppe) `sts.amazonaws.com`.
   d. Wählen Sie Weiter aus.
5. Gehen Sie auf der Seite Berechtigungen hinzufügen wie folgt vor:
   a. Geben Sie im Feld Richtlinien filtern den Text AmazonS3-Richtlinie einCSIDriver.

      > **Note**
      >
      > Diese Richtlinie wurde im vorherigen Abschnitt erstellt.

   b. Aktivieren Sie das Kontrollkästchen links neben dem Ergebnis für `AmazonS3CSIDriverPolicy`, das bei der Suche zurückgegeben wurde.
   c. Wählen Sie Weiter aus.
6. Gehen Sie auf der Seite Name, review, and create (Benennen, überprüfen und erstellen) wie folgt vor:
   a. Geben Sie als Rollenname einen eindeutigen Namen für Ihre Rolle ein, z. B. AmazonEKS_S3_CSI_. DriverRole

b. Fügen Sie der Rolle unter Tags hinzufügen (optional) Metadaten hinzu, indem Sie Tags als Schlüssel-Wert-Paare anfügen. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen im IAM-Benutzerhandbuch.

c. Wählen Sie Create role (Rolle erstellen) aus.

7. Nachdem die Rolle erstellt wurde, wählen Sie die Rolle in der Konsole aus, um sie zur Bearbeitung zu öffnen.
8. Wählen Sie die Registerkarte Trust Relationships (Vertrauensstellungen) und dann Edit trust policy (Vertrauensrichtlinie bearbeiten) aus.
9. Suchen Sie die Zeile, die wie folgt aussieht:

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud":
 "sts.amazonaws.com"
```

Fügen Sie am Ende der vorherigen Zeile ein Komma und anschließend die folgende Zeile hinzu. Ersetzen Sie es durch die Region, in der *region-code* sich Ihr Cluster befindet AWS . Ersetzen Sie *EXAMPLED539D4633E53DE1B71EXAMPLE* mit dem OIDC-Anbieter-ID Ihres Clusters.

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub":
 "system:serviceaccount:kube-system:s3-csi-driver-sa"
```

10. Stellen Sie sicher, dass der `Condition`-Operator auf `"StringEquals"` eingestellt ist.
11. Wählen Sie Update Policy (Richtlinie aktualisieren) aus, um den Vorgang abzuschließen.

AWS CLI

1. Zeigen Sie die OIDC-Anbieter-URL Ihres Clusters an. Ersetzen Sie *my-cluster* mit dem Namen Ihres Clusters. Wenn die Ausgabe des Befehls `None` ist, überprüfen Sie die Voraussetzungen.

```
aws eks describe-cluster --name my-cluster --query "cluster.identity.oidc.issuer" --
output text
```

Eine Beispielausgabe sieht wie folgt aus.

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

2. Erstellen Sie die IAM-Rolle und weisen Sie dem Kubernetes-Dienstkonto die `AssumeRoleWithWebIdentity`-Aktion zu.

a. Kopieren Sie den folgenden Inhalt in eine Datei namens `aws-s3-csi-driver-trust-policy.json`. Ersetzen Sie *111122223333* durch Ihre Konto-ID. Ersetzen Sie *EXAMPLED539D4633E53DE1B71EXAMPLE* und *region-code* mit den im vorherigen Schritt zurückgegebenen Werten.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/oidc.eks.us-
east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "oidc.eks.us-east-1.amazonaws.com/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-system:s3-csi-
driver-sa",
          "oidc.eks.us-east-1.amazonaws.com/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
        }
      }
    }
  ]
}
```

b. Erstellen Sie die -Rolle. Sie können *AmazonEKS_S3_CSI_DriverRole* in einen anderen Namen ändern, aber wenn Sie dies tun, stellen Sie sicher, dass Sie ihn auch in späteren Schritten ändern.

```
aws iam create-role \
  --role-name AmazonEKS_S3_CSI_DriverRole \
  --assume-role-policy-document file://"aws-s3-csi-driver-trust-policy.json"
```

3. Hängen Sie die zuvor erstellte IAM-Richtlinie mit dem folgenden Befehl an die Rolle an.

```
aws iam attach-role-policy \
  --policy-arn arn:aws: iam::aws:policy/AmazonS3CSIDriverPolicy \
  --role-name AmazonEKS_S3_CSI_DriverRole
```

> **Note**
>
> Die IAM-Richtlinie `AmazonS3CSIDriverPolicy` wurde im vorherigen Abschnitt erstellt.

4. Überspringen Sie diesen Schritt, wenn Sie den Treiber als Amazon-EKS-Add-On installieren. Für selbstverwaltete Installationen des Treibers erstellen Sie Kubernetes-Servicekonten, die mit der ARN der von Ihnen erstellten IAM-Rolle vermerkt sind.

   a. Speichern Sie den folgenden Inhalt in einer Datei mit dem Namen `mountpoint-s3-service-account.yaml`. Ersetzen Sie *`111122223333`* durch Ihre Konto-ID.

   ```
   ---
   apiVersion: v1
   kind: ServiceAccount
   metadata:
     labels:
       app.kubernetes.io/name: aws-mountpoint-s3-csi-driver
     name: mountpoint-s3-csi-controller-sa
     namespace: kube-system
     annotations:
       eks.amazonaws.com/role-arn: arn:aws: iam::111122223333:role/
   AmazonEKS_S3_CSI_DriverRole
   ```

   b. Erstellen Sie das Kubernetes-Servicekonto in Ihrem Cluster. Das Kubernetes-Dienstkonto (`mountpoint-s3-csi-controller-sa`) ist mit der IAM-Rolle versehen, die Sie mit dem Namen erstellt haben. *`AmazonEKS_S3_CSI_DriverRole`*

   ```
   kubectl apply -f mountpoint-s3-service-account.yaml
   ```

   > **Note**
   >
   > Wenn Sie das Plugin in diesem Verfahren bereitstellen, wird es erstellt und für die Verwendung eines Servicekontos mit dem Namen `s3-csi-driver-sa` konfiguriert.

Schritt 3: CSI-Treiber für Mountpoint für Amazon S3 installieren

Sie können den CSI-Treiber für Mountpoint für Amazon S3 über das Amazon-EKS-Add-On installieren. Sie können die folgenden Tools verwenden, um das Add-On zu Ihrem Cluster hinzuzufügen:

- the section called “eksctl”
- the section called “AWS-Managementkonsole”
- the section called “AWS CLI”

Alternativ können Sie den CSI-Treiber für Mountpoint für Amazon S3 als selbstverwaltete Installation installieren. Anweisungen zur Durchführung einer selbstverwalteten Installation finden Sie im GitHub unter Installation.

Ab Version `v1.8.0` können Sie Taints konfigurieren, die für die Pods des CSI-Treibers toleriert werden sollen. Geben Sie dazu entweder einen benutzerdefinierten Satz von Taints an, die mit `node.tolerations` toleriert werden sollen, oder tolerieren Sie alle Taints mit `node.tolerateAllTaints`. Weitere Informationen finden Sie unter Taints und Toleranzen in der Kubernetes-Dokumentation.

eksctl

So fügen Sie das CSI-Add-On von Amazon S3 mit **`eksctl`** hinzu

Führen Sie den folgenden Befehl aus. Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters, *`111122223333`* mit Ihrer Konto-ID und *`AmazonEKS_S3_CSI_DriverRole`* mit dem Namen der zuvor erstellten IAM-Rolle.

```
eksctl create addon --name aws-mountpoint-s3-csi-driver --cluster my-cluster \
  --service-account-role-arn arn:aws: iam::111122223333:role/
AmazonEKS_S3_CSI_DriverRole --force
```

Wenn Sie die Option *`--force`* entfernen und eine der Amazon-EKS-Add-on-Einstellungen mit Ihren vorhandenen Einstellungen in Konflikt steht, schlägt die Aktualisierung des Amazon-EKS-Add-ons fehl und Sie erhalten eine Fehlermeldung, die Sie bei der Lösung des Konflikts unterstützt. Stellen Sie vor der Angabe dieser Option sicher, dass das Amazon-EKS-Add-On keine Einstellungen verwaltet, die Sie verwalten müssen, da diese Einstellungen mit dieser Option überschrieben werden. Weitere Informationen zu anderen Optionen für diese Einstellung finden Sie unter Add-Ons in der `eksctl`-

Dokumentation. Weitere Informationen zur Amazon-EKS-Kubernetes-Feldverwaltung finden Sie unter the section called “Felder, die angepasst werden können”.

Sie können `eksctl` über Konfigurationsdateien anpassen. Weitere Informationen finden Sie unter Arbeiten mit Konfigurationswerten in der `eksctl`-Dokumentation. Das folgende Beispiel zeigt, wie alle Taints toleriert werden.

```
# config.yaml
...

addons:
- name: aws-mountpoint-s3-csi-driver
  serviceAccountRoleARN: arn:aws: iam::111122223333:role/AmazonEKS_S3_CSI_DriverRole
  configurationValues: |-
    node:
      tolerateAllTaints: true
```

AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich Cluster aus.
3. Wählen Sie den Namen des Clusters, für den Sie den CSI-Treiber für Mountpoint für Amazon S3 konfigurieren möchten.
4. Wählen Sie die Registerkarte Add-ons.
5. Wählen Sie Weitere Add-Ons erhalten.
6. Führen Sie auf der Seite Add-Ons auswählen die folgenden Schritte aus:
   a. Aktivieren Sie im Abschnitt Amazon-EKS-Add-Ons das Kontrollkästchen CSI-Treiber für Mountpoint für Amazon S3.
   b. Wählen Sie Weiter aus.
7. Gehen Sie auf der Seite Konfigurieren ausgewählter Add-Ons-Einstellungen wie folgt vor:
   a. Wählen Sie die Version aus, die Sie verwenden möchten.
   b. Wählen Sie für IAM-Rolle auswählen den Namen einer IAM-Rolle aus, an die Sie die IAM-Richtlinie „CSI-Treiber für Mountpoint für Amazon S3“ angefügt haben.
   c. (Optional) Aktualisieren Sie die Methode zur Konfliktlösung, nachdem Sie die Optionalen Konfigurationseinstellungen erweitert haben Wenn Sie Überschreiben auswählen, können eine oder mehrere Einstellungen für das vorhandene Add-On mit den Amazon-EKS-Add-On-

Einstellungen überschrieben werden. Wenn Sie diese Option nicht aktivieren und ein Konflikt mit Ihren vorhandenen Einstellungen vorliegt, schlägt der Vorgang fehl. Sie können die sich daraus ergebende Fehlermeldung heranziehen, um den Konflikt zu beheben. Stellen Sie vor der Auswahl dieser Option sicher, dass das Amazon-EKS-Add-On keine Einstellungen verwaltet, die Sie selbst verwalten müssen.

d. (Optional) Konfigurieren Sie Toleranzen im Feld Konfigurationswerte, nachdem Sie die Optionalen Konfigurationseinstellungen erweitert haben.

e. Wählen Sie Weiter aus.

8. Wählen Sie auf der Seite Überprüfen und hinzufügen die Option Erstellen aus. Nachdem die Installation der Add-Ons abgeschlossen ist, wird Ihr installiertes Add-On angezeigt.

AWS CLI

So fügen Sie das Mountpoint for Amazon S3 CSI Add-on mit der CLI hinzu AWS

Führen Sie den folgenden Befehl aus. Ersetzen Sie *my-cluster* mit dem Namen Ihres Clusters, *111122223333* mit Ihrer Konto-ID und *AmazonEKS_S3_CSI_DriverRole* mit dem Namen der zuvor erstellten Rolle.

```
aws eks create-addon --cluster-name my-cluster --addon-name aws-mountpoint-s3-csi-
driver \
  --service-account-role-arn arn:aws: iam::111122223333:role/
AmazonEKS_S3_CSI_DriverRole
```

Sie können den Befehl mit dem `--configuration-values`-Flag anpassen. Das folgende alternative Beispiel veranschaulicht, wie alle Taints toleriert werden.

```
aws eks create-addon --cluster-name my-cluster --addon-name aws-mountpoint-s3-csi-
driver \
  --service-account-role-arn arn:aws: iam::111122223333:role/
AmazonEKS_S3_CSI_DriverRole \
  --configuration-values '{"node":{"tolerateAllTaints":true}}'
```

Schritt 4: Mountpoint für Amazon S3 konfigurieren

In den meisten Fällen können Sie Mountpoint für Amazon S3 mit lediglich einem Bucket-Namen konfigurieren. Anweisungen zur Konfiguration von Mountpoint für Amazon S3 finden Sie unter Mountpoint für Amazon S3 konfigurieren auf. GitHub

Schritt 5: Beispielanwendung bereitstellen

Sie können die statische Bereitstellung des Treibers in einem vorhandenen Amazon S3-Bucket einsetzen. Weitere Informationen finden Sie unter Statische Bereitstellung auf. GitHub

Amazon-EKS-Add-On von Mountpoint für Amazon S3 entfernen

Sie haben zwei Möglichkeiten, den CSI-Treiber für Mountpoint für Amazon S3 zu entfernen.

- Beibehalten von Add-on-Software auf Ihrem Cluster - Diese Option entfernt die Amazon-EKS-Verwaltung aller Einstellungen. Amazon EKS wird außerdem die Möglichkeit aufgehoben, Sie über Updates zu informieren und das Amazon-EKS-Add-on automatisch zu aktualisieren, nachdem Sie ein Update eingeleitet haben. Es behält jedoch die Add-on-Software auf Ihrem Cluster bei. Diese Option macht das Add-on zu einem selbstverwalteten Add-on statt einem Amazon-EKS-Add-on. Bei dieser Option gibt es keine Ausfallzeiten für das Add-On. Die Befehle in diesem Verfahren verwenden diese Option.
- Entfernen Sie die Add-on-Software vollständig aus Ihrem Cluster – Wir empfehlen, dass Sie das Amazon-EKS-Add-on nur aus Ihrem Cluster entfernen, wenn auf Ihrem Cluster keine Ressourcen vorhanden sind, die davon abhängig sind. Um diese Option durchzuführen, löschen Sie `--preserve` aus dem Befehl, den Sie für dieses Verfahren verwenden.

Wenn dem Add-On ein IAM-Konto zugeordnet ist, wird das IAM-Konto nicht entfernt.

Sie können die folgenden Tools verwenden, um das Amazon-S3-CSI-Add-On zu entfernen:

- the section called “eksctl”
- the section called “AWS-Managementkonsole”
- the section called “AWS-CLI”

eksctl

So entfernen Sie das Amazon-S3-CSI-Add-On mithilfe von **`eksctl`**

Ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters und führen Sie dann den folgenden Befehl aus.

```
eksctl delete addon --cluster my-cluster --name aws-mountpoint-s3-csi-driver --preserve
```

AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich die Option Cluster aus.
3. Wählen Sie den Namen des Clusters aus, für den Sie das Amazon-EBS-CSI-Add-on entfernen möchten.
4. Wählen Sie die Registerkarte Add-ons.
5. Wählen Sie CSI-Treiber für Mountpoint für Amazon S3 aus.
6. Wählen Sie Remove (Entfernen) aus.
7. Führen Sie im Bestätigungsdialogfeld Entfernen: aws-mountpoint-s3-csi-driver die folgenden Schritte aus:
   a. Wenn Amazon EKS die Verwaltung von Einstellungen für das Add-on einstellen soll, wählen Sie Auf dem Cluster beibehalten. Tun Sie dies, wenn Sie die Add-on-Software auf Ihrem Cluster behalten möchten. Auf diese Weise können Sie alle Einstellungen des Add-ons selbst verwalten.
   b. Geben Sie `aws-mountpoint-s3-csi-driver` ein.
   c. Wählen Sie Entfernen aus.

AWS-CLI

So entfernen Sie das Amazon-S3-CSI-Add-On mithilfe der AWS-CLI

Ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters und führen Sie dann den folgenden Befehl aus.

```
aws eks delete-addon --cluster-name my-cluster --addon-name aws-mountpoint-s3-csi-
driver --preserve
```

Snapshot-Funktion für CSI-Volumes aktivieren

Mit der Snapshot-Funktion können Sie zeitpunktgenaue Kopien Ihrer Daten erstellen. Damit diese Funktion in Kubernetes funktioniert, benötigen Sie sowohl einen CSI-Treiber mit Snapshot-Support (z. B. den Amazon-EBS-CSI-Treiber) als auch einen CSI-Snapshot-Controller. Der Snapshot-Controller ist entweder als verwaltetes Amazon-EKS-Add-On oder als selbstverwaltete Installation verfügbar.

Die folgenden Punkte sind bei der Verwendung des CSI-Snapshot-Controllers zu beachten.

- Der Snapshot-Controller muss zusammen mit einem CSI-Treiber mit Snapshot-Funktion installiert werden. Installationsanweisungen für den Amazon-EBS-CSI-Treiber finden Sie unter the section called “Amazon EBS”.
- Kubernetes unterstützt keine Snapshots von Volumes, die über die CSI-Migration bereitgestellt werden, wie z. B. Amazon-EBS-Volumes, die eine `StorageClass` mit Provisioner `kubernetes.io/aws-ebs` verwenden. Volumes müssen mit einer `StorageClass` erstellt werden, die auf den CSI-Treiber-Provisioner `ebs.csi.aws.com` verweist.
- Amazon EKS Auto Mode beinhaltet keinen Snapshot-Controller. Die Speicher-Kapazität von EKS Auto Mode ist mit dem Snapshot-Controller kompatibel.

Wir empfehlen, den CSI-Snapshot über das von Amazon EKS verwaltete Add-on zu installieren. Dieses Add-On enthält die benutzerdefinierten Ressourcendefinitionen (CRDs), die zum Erstellen und Verwalten von Snapshots in Amazon EKS benötigt werden. Wenn Sie ein Amazon-EKS-Add-on zu Ihrem Cluster hinzufügen möchten, lesen Sie the section called “Add-On erstellen”. Weitere Informationen zu Add-ons finden Sie unter the section called “Amazon-EKS-Add-ons”.

Wenn Sie alternativ eine selbstverwaltete Installation des CSI-Snapshot-Controllers wünschen, finden Sie weitere Informationen unter Verwendung im Upstream-Kubernetes `external-snapshotter` auf GitHub.

Konfigurieren des Netzwerks für Amazon-EKS-Cluster

Ihr Amazon-EKS-Cluster wird in einer VPC erstellt. Pod-Netzwerke werden durch das Amazon VPC Container Network Interface (CNI)-Plugin für Knoten bereitgestellt, die in der AWS-Infrastruktur ausgeführt werden. Wenn Sie Knoten in Ihrer eigenen Infrastruktur ausführen, lesen Sie the section called “CNI konfigurieren”. In diesem Kapitel finden Sie die folgenden Themen, um mehr über Networking für Ihren Cluster zu erfahren.

Themen

- Hinzufügen eines vorhandenen VPC-Subnetzes zu einem Amazon-EKS-Cluster über die Management-Konsole
- Amazon-EKS-Netzwerkanforderungen für VPC und Subnetze
- Erstellung einer Amazon VPC für Ihren Amazon-EKS-Cluster
- Anforderungen der Amazon-EKS-Sicherheitsgruppe für Cluster anzeigen
- Verwaltung von Netzwerk-Add-Ons für Amazon-EKS-Cluster

Hinzufügen eines vorhandenen VPC-Subnetzes zu einem Amazon-EKS-Cluster über die Management-Konsole

1. Navigieren Sie in der Management-Konsole zu Ihrem Cluster
2. Wählen Sie auf der Registerkarte Netzwerk die Option VPC-Ressourcen verwalten aus
3. Wählen Sie aus der Dropdown-Liste Subnetze weitere Subnetze aus der VPC Ihres Clusters aus.

So erstellen Sie ein neues VPC-Subnetz:

- EKS-Subnetz-Anforderungen überprüfen
- Beachten Sie den Abschnitt Erstellen eines Subnetzes im Benutzerhandbuch zur Amazon Virtual Private Cloud.

Amazon-EKS-Netzwerkanforderungen für VPC und Subnetze

Wenn Sie einen Cluster erstellen, geben Sie eine VPC und mindestens zwei Subnetze an, die sich in unterschiedlichen Availability Zones befinden. Dieses Thema bietet einen Überblick über

die spezifischen Amazon-EKS-Anforderungen und Überlegen zur VPC und den Subnetzen, die Sie in Ihrem Cluster verwenden. Wenn Sie nicht über VPC für die Verwendung mit Amazon EKS verfügen, lesen Sie the section called “Erstellen einer VPC”. Wenn Sie einen lokalen oder erweiterten Cluster auf AWS Outposts erstellen, lesen Sie the section called “Erstellen von VPC und Subnetzen” statt dieses Themas weiter. Der Inhalt dieses Themas bezieht sich auf Amazon-EKS-Cluster mit Hybridknoten. Weitere Netzwerkanforderungen für Hybridknoten finden Sie unter the section called “Netzwerk vorbereiten”.

VPC-Anforderungen und -Überlegungen

Wenn Sie einen Cluster erstellen, muss die von Ihnen angegebene VPC die folgenden Anforderungen und Überlegungen erfüllen:

- Die VPC muss über eine ausreichende Anzahl von IP-Adressen für den Cluster, alle Knoten und andere Kubernetes-Ressourcen, die Sie erstellen möchten, verfügen. Wenn die VPC, die Sie verwenden möchten, nicht über eine ausreichende Anzahl von IP-Adressen verfügt, versuchen Sie, die Anzahl der verfügbaren IP-Adressen zu erhöhen.

  Sie können dies tun, indem Sie die Clusterkonfiguration aktualisieren, um zu ändern, welche Subnetze und Sicherheitsgruppen der Cluster verwendet. Sie können von der AWS-Managementkonsole, der neuesten Version der AWS CLI AWS CloudFormation, und `eksctl` Version `v0.164.0-rc.0` oder höher aus aktualisieren. Möglicherweise müssen Sie dies tun, um Subnetzen mehr verfügbare IP-Adressen zur Verfügung zu stellen, damit eine Clusterversion erfolgreich aktualisiert werden kann.

  > **Important**
  >
  > Alle Subnetze, die Sie hinzufügen, müssen sich in derselben Gruppe befinden, die ursprünglich bei der AZs Erstellung des Clusters bereitgestellt wurde. Neue Subnetze müssen alle anderen Anforderungen erfüllen, z. B. müssen sie über ausreichend IP-Adressen verfügen.
  > Angenommen, Sie haben einen Cluster erstellt und vier Subnetze angegeben. In der Reihenfolge, in der Sie sie angegeben haben, befindet sich das erste Subnetz in der `us-west-2a` Availability Zone, das zweite und dritte Subnetz in der `us-west-2b` Availability Zone und das vierte Subnetz in der `us-west-2c` Availability Zone. Wenn Sie die Subnetze ändern möchten, müssen Sie in jeder der drei Availability Zones mindestens ein Subnetz

bereitstellen, und die Subnetze müssen sich in derselben VPC wie die ursprünglichen Subnetze befinden.

Wenn Sie mehr IP-Adressen benötigen, als die CIDR-Blöcke in der VPC haben, können Sie zusätzliche CIDR-Blöcke hinzufügen, indem Sie Ihrer VPC zusätzliche Classless Inter-Domain Routing (CIDR)-Blöcke zuordnen. Sie können entweder vor oder nach der Erstellung Ihres Clusters private (RFC 1918) und öffentliche (nicht RFC 1918) CIDR-Blöcke mit Ihrer VPC verbinden.

Sie können Knoten hinzufügen, die den neuen CIDR-Block verwenden, unmittelbar nachdem Sie ihn hinzugefügt haben. Da die Steuerebene den neuen CIDR-Block jedoch erst nach Abschluss der Abstimmung erkennt, kann es bis zu einer Stunde dauern, bis ein CIDR-Block, den Sie einer VPC zugeordnet haben, von einem Cluster erkannt wird. Anschließend können Sie die Befehle`kubectl attach`,, `kubectl cp` `kubectl exec`kubectl `kubectl logs`, und (diese `kubectl port-forward` Befehle verwenden die`kubelet API`) für Knoten und Pods im neuen CIDR-Block ausführen. Wenn Sie Pods haben, die als Webhook-Backend fungieren, müssen Sie außerdem warten, bis die Abstimmung der Steuerebene abgeschlossen ist.

- Vermeiden Sie Überschneidungen von IP-Adressbereichen, wenn Sie Ihren EKS-Cluster VPCs über Transit Gateway, VPC-Peering oder andere Netzwerkkonfigurationen mit anderen verbinden. CIDR-Konflikte treten auf, wenn sich der Service-CIDR Ihres EKS-Clusters mit dem CIDR einer verbundenen VPC überschneidet. In diesen Szenarien haben Service-IP-Adressen Vorrang vor Ressourcen, die VPCs mit derselben IP-Adresse verbunden sind, obwohl das Routing des Datenverkehrs unvorhersehbar werden kann und Anwendungen möglicherweise keine Verbindung zu den vorgesehenen Ressourcen herstellen können.

  Um CIDR-Konflikte zu vermeiden, stellen Sie sicher, dass sich Ihr EKS-Dienst-CIDR nicht mit einer verbundenen VPC überschneidet, CIDRs und führen Sie eine zentrale Aufzeichnung aller CIDR-Zuweisungen. Wenn Sie CIDR-Überschneidungen feststellen, können Sie ein Transit-Gateway mit einer freigegebenen Service-VPC verwenden. Weitere Informationen finden Sie unter Isoliert VPCs mit Shared Services und Muster zur Aufbewahrung routingfähiger IP-Adressen von Amazon EKS VPC in einem Hybridnetzwerk. Weitere Informationen finden Sie auf der Seite Überlegungen zu VPC und Subnetzen im EKS Best Practices Guide im VPCs Abschnitt Kommunikation zwischen beiden Seiten.
- Wenn Sie möchten, dass Kubernetes `IPv6`-Adressen zu Pods und Services zuweist, ordnen Sie Ihrer VPC einen `IPv6`-CIDR-Block zu. Weitere Informationen finden Sie unter Verknüpfen eines IPv6 CIDR-Blocks mit Ihrer VPC im Amazon VPC-Benutzerhandbuch. Sie können keine `IPv6`-Adressen mit Pods und Services verwenden, die auf Hybridknoten ausgeführt werden, und Sie

können keine Hybridknoten mit Clustern verwenden, die mit der IPv6-IP-Adressfamilie konfiguriert sind.

- Die VPC muss DNS-Hostnamen und die DNS-Auflösung unterstützen. Andernfalls können keine Knoten dem Cluster beitreten. Weitere Informationen finden Sie unter DNS-Attribute für Ihre VPC im Amazon-VPC-Benutzerhandbuch.
- Die VPC erfordert möglicherweise die Verwendung von VPC-Endpunkten. AWS PrivateLink Weitere Informationen finden Sie unter the section called “Subnetz-Anforderungen und -Überlegungen”.

Wenn Sie einen Cluster mit Kubernetes `1.14` oder früher erstellt haben, hat Amazon EKS das folgende Tag zu Ihrer VPC hinzugefügt:

Schlüssel	Wert
`kubernetes.io/cluster/ my-cluster`	`owned`

Dieses Tag wurde nur von Amazon EKS verwendet. Sie können das Tag entfernen, ohne Ihre Services zu beeinträchtigen. Es wird nicht mit Clustern der Version `1.15` oder höher verwendet.

Subnetz-Anforderungen und -Überlegungen

Wenn Sie einen Cluster erstellen, erstellt Amazon EKS 2–4 Elastic-Network-Schnittstellen in den von Ihnen angegebenen Subnetzen. Diese Netzwerkschnittstellen ermöglichen die Kommunikation zwischen Ihrem Cluster und Ihrer VPC. Diese Netzwerkschnittstellen ermöglichen auch Kubernetes-Funktionen, die die verwenden. `kubelet API` Die Verbindungen zur `kubelet` API werden in den Befehlen`kubectl attach`,, `kubectl cp` `kubectl exec`kubectl `logs`, und `kubectl port-forward` verwendet. Jede von Amazon EKS erstellte Netzwerkschnittstelle enthält den Text `Amazon EKS cluster-name` in ihrer Beschreibung.

Amazon EKS kann Netzwerkschnittstellen in jedem Subnetz erstellen, dass Sie bei der Erstellung eines Clusters angeben. Nach der Erstellung Ihres Clusters können Sie ändern, in welchen Subnetzen Amazon EKS seine Netzwerkschnittstellen erstellt. Wenn Sie die Kubernetes-Version eines Clusters aktualisieren, löscht Amazon EKS die ursprünglichen Netzwerkschnittstellen, die es erstellt hat, und erstellt neue Netzwerkschnittstellen. Diese Netzwerkschnittstellen können in denselben Subnetzen wie die ursprünglichen Netzwerkschnittstellen oder in anderen Subnetzen als die ursprünglichen Netzwerkschnittstellen erstellt werden. Um zu steuern, in welchen Subnetzen

Netzwerkschnittstellen erstellt werden, können Sie die Anzahl der von Ihnen angegebenen Subnetze beim Erstellen eines Clusters auf zwei beschränken oder die Subnetze nach der Erstellung des Clusters aktualisieren.

Subnetzanforderungen für Cluster

Die Subnetze, die Sie angeben, wenn Sie einen Cluster erstellen oder aktualisieren, müssen die folgenden Anforderungen erfüllen:

- Die Subnetze müssen jeweils mindestens 6 IP-Adressen zur Verwendung durch Amazon EKS haben. Wir empfehlen jedoch mindestens 16 IP-Adressen.
- Die Subnetze müssen sich in mindestens zwei verschiedenen Availability Zones befinden.
- Die Subnetze können sich nicht in AWS Outposts oder Wavelength befinden. AWS Wenn Sie sich in Ihrer VPC befinden, können Sie jedoch selbstverwaltete Knoten und Kubernetes-Ressourcen für diese Subnetz-Typen bereitstellen. Weitere Informationen zu Ihren selbstverwalteten Knoten finden Sie unter the section called “Selbstverwaltete Knoten”.
- Die Subnetze können öffentlich oder privat sein. Es wird jedoch empfohlen, wenn möglich private Subnetze anzugeben. Ein öffentliches Subnetz ist ein Subnetz mit einer Routing-Tabelle, die eine Route zu einem Internet-Gateway enthält. Ein privates Subnetz ist ein Subnetz mit einer Routing-Tabelle, die keine Route zu einem Internet-Gateway enthält.
- Die Subnetze dürfen sich nicht in folgenden Availability Zones befinden:

AWS Region	Name der Region	Unzulässige Verfügbarkeitszone IDs
`us-east-1`	USA Ost (Nord-Virginia)	`use1-az3`
`us-west-1`	USA West (Nordkalifornien)	`usw1-az2`
`ca-central-1`	Kanada (Zentral)	`cac1-az3`

Verwendung der IP-Adressfamilie nach Komponenten

Die folgende Tabelle enthält die IP-Adressfamilie, die von jeder Komponente von Amazon EKS verwendet wird. Sie können eine Network Address Translation (NAT) oder ein anderes Kompatibilitätssystem verwenden, um eine Verbindung zu diesen Komponenten von Quell-IP-Adressen in Familien herzustellen, deren Tabelleneintrag den Wert „Nein“ aufweist.

Die Funktionalität kann je nach Einstellung der IP-Familie (`ipFamily`) des Clusters variieren. Diese Einstellung ändert den Typ der IP-Adressen, die für den CIDR-Block verwendet werden, den Kubernetes den Services zuweist. Ein Cluster mit dem Einstellungswert von IPv4 wird als IPv4 Cluster bezeichnet, und ein Cluster mit dem Einstellungswert von IPv6 wird als IPv6 Cluster bezeichnet.

Komponente	IPv4 Adressen	IPv6 adressen	Dual-Stack-Adressen
Öffentlicher Endpunkt der EKS-API	Ja[1,3]	Ja[1,3]	Ja[1,3]
EKS-API-VPC-Endpunkt	Ja	Nein	Nein
Öffentlicher Endpunkt der EKS-Auth-API (EKS Pod Identity)	Ja[1]	Ja[1]	Ja[1]
VPC-Endpunkt der EKS-Auth-API (EKS Pod Identity)	Ja[1]	Ja[1]	Ja[1]
Öffentlicher Endpunkt des `IPv4`-Kubernetes-Clusters[2]	Ja	Nein	Nein
Privater Endpunkt des `IPv4`-Kubernetes-Clusters[2]	Ja	Nein	Nein
Öffentlicher Endpunkt des `IPv6`-Kubernetes-Clusters[2]	Ja[1,4]	Ja[1,4]	Ja[4]
Privater Endpunkt des `IPv6`-Kubernetes-Clusters[2]	Ja[1,4]	Ja[1,4]	Ja[4]

Komponente	IPv4 Adressen	IPv6 adressen	Dual-Stack-Adressen
Kubernetes-Cluster-Subnetze	Ja[2]	Nein	Ja[2]
Primäre IP-Adressen der Knoten	Ja[2]	Nein	Ja[2]
Cluster-CIDR-Bereich für Service-IP-Adressen	Ja[2]	Ja[2]	Nein
Pod-IP-Adressen vom VPC CNI	Ja[2]	Ja[2]	Nein
IRSA OIDC-Emittent URLs	Ja[1,3]	Ja[1,3]	Ja[1,3]

Note

[1] Der Endpunkt ist ein Dual-Stack mit sowohl `IPv4`- als auch `IPv6`-Adressen. Ihre Anwendungen außerhalb AWS, Ihre Knoten für den Cluster und Ihre Pods innerhalb des Clusters können diesen Endpunkt entweder mit oder erreichen. `IPv4` `IPv6`

[2] Sie wählen beim Erstellen eines Clusters in der IP-Familieneinstellung (`ipFamily`) des Clusters zwischen einem `IPv4`-Cluster und einem `IPv6`-Cluster. Dies kann nicht geändert werden. Stattdessen müssen Sie beim Erstellen eines weiteren Clusters und beim Migrieren Ihrer Workloads eine andere Einstellung wählen.

[3] Der Dual-Stack-Endpunkt wurde im August 2024 eingeführt. Informationen zur Verwendung der Dual-Stack-Endpunkte mit der AWS CLI finden Sie in der Konfiguration von Dual-Stack- und FIPS-Endpunkten im AWS SDKs Referenzhandbuch zu Tools. Nachfolgend sind die neuen Endpunkte aufgeführt:

Öffentlicher Endpunkt der EKS-API

```
eks.region.api.aws
```

IRSA OIDC-Emittent URLs

```
oidc-eks.region.api.aws
```

[4] Der Dual-Stack-Cluster-Endpunkt wurde im Oktober 2024 eingeführt. EKS erstellt den folgenden Endpunkt für neue Cluster, die nach diesem Datum erstellt werden und die IPv6 in der IP-Familieneinstellung (ipFamily) des Clusters auswählen

public/private EKS-Cluster-Endpunkt

```
eks-cluster.region.api.aws
```

Subnetzanforderungen für Knoten

Sie können Knoten und Kubernetes-Ressourcen in denselben Subnetzen bereitstellen, die Sie beim Erstellen Ihres Clusters angeben. Das ist aber nicht unbedingt notwendig. Sie können Knoten und Kubernetes-Ressourcen auch in Subnetzen bereitstellen, die Sie nicht beim Erstellen des Clusters angegeben haben. Wenn Sie Knoten in verschiedenen Subnetzen bereitstellen, erstellt Amazon EKS keine Cluster-Netzwerkschnittstellen in diesen Subnetzen. Jedes Subnetz, für das Sie Knoten und Kubernetes-Ressourcen bereitstellen, muss die folgenden Anforderungen erfüllen:

- Die Subnetze müssen über genügend verfügbare IP-Adressen verfügen, um alle Ihre Knoten und Kubernetes-Ressourcen bereitzustellen.
- Wenn Kubernetes Pods und Services IPv6-Adressen zuweisen soll, müssen Sie über einen IPv6-CIDR-Block und einen IPv4-CIDR-Block verfügen, die Ihrem Subnetz zugeordnet sind. Weitere Informationen finden Sie unter Zuordnen eines IPv6 CIDR-Blocks zu Ihrem Subnetz im Amazon VPC-Benutzerhandbuch. Die Routing-Tabellen, die Ihren Subnetzen zugeordnet sind, müssen Routen zu IPv4- und IPv6-Adressen enthalten. Weitere Informationen finden Sie unter Routen im Amazon-VPC-Benutzerhandbuch. Pods werden nur einer IPv6-Adresse zugewiesen. Die Netzwerkschnittstellen, die Amazon EKS für Ihren Cluster und Ihre Knoten erstellt, werden jedoch einer IPv4- und einer IPv6-Adresse zugewiesen.
- Wenn Sie eingehenden Zugriff aus dem Internet auf Ihre Pods benötigen, stellen Sie sicher, dass Sie mindestens ein öffentliches Subnetz mit genügend verfügbaren IP-Adressen haben, in denen Sie Load Balancer und Ingresse bereitstellen können. Sie können Load Balancer in öffentlichen Subnetzen bereitstellen. Load Balancer können ein Load Balancing zu Pods in privaten und öffentlichen Subnetzen vornehmen. Wir empfehlen, Ihre Knoten nach Möglichkeit in privaten Subnetzen bereitzustellen.
- Wenn Sie planen, Knoten in einem öffentlichen Subnetz bereitzustellen, muss das Subnetz automatisch öffentliche IPv4- oder IPv6-Adressen zuweisen. Wenn Sie Knoten in einem privaten

Subnetz bereitstellen, das einen zugeordneten IPv6-CIDR-Block hat, muss das private Subnetz zudem IPv6-Adressen automatisch zuweisen. Wenn Sie die von Amazon EKS bereitgestellte AWS CloudFormation Vorlage für die Bereitstellung Ihrer VPC nach dem 26. März 2020 verwendet haben, ist diese Einstellung aktiviert. Wenn Sie die Vorlagen verwendet haben, um Ihre VPC vor diesem Datum bereitzustellen oder Ihre eigene VPC verwenden, müssen Sie diese Einstellung manuell aktivieren. Informationen zur Vorlage finden Sie unter the section called “Erstellen einer VPC”. Weitere Informationen finden Sie unter Ändern des öffentlichen IPv4 Adressierungsattributs für Ihr Subnetz und Ändern des IPv6 Adressierungsattributs für Ihr Subnetz im Amazon VPC-Benutzerhandbuch.

- Wenn es sich bei dem Subnetz, in dem Sie einen Knoten bereitstellen, um ein privates Subnetz handelt und die Routing-Tabelle keine Route zu einem NAT-Gerät (Network Address Translation) () oder einem reinen Ausgangsgateway (IPv4IPv6) enthält, fügen Sie Ihrer VPC VPC-Endpunkte hinzu, die Sie verwenden. AWS PrivateLink VPC-Endpunkte werden für alle AWS Dienste benötigt, mit denen Ihre Knoten und Pods kommunizieren müssen. Beispiele hierfür sind Amazon ECR, Elastic Load Balancing CloudWatch, Amazon, AWS Security Token Service und Amazon Simple Storage Service (Amazon S3). Der Endpunkt muss das Subnetz enthalten, in dem sich die Knoten befinden. Nicht alle AWS Dienste unterstützen VPC-Endpunkte. Weitere Informationen finden Sie unter Was ist? AWS PrivateLink und AWS Dienste, die sich in integrieren lassen AWS PrivateLink. Eine Liste weiterer Amazon-EKS-Anforderungen finden Sie unter the section called “Private Cluster”.
- Wenn Sie Load Balancer in einem Subnetz bereitstellen möchten, muss das Subnetz das folgende Tag haben:
  - Private Subnetze

Schlüssel	Wert
kubernetes.io/role/internal-elb	1

  - Öffentliche Subnetze

Schlüssel	Value (Wert)
kubernetes.io/role/elb	1

Wenn ein Kubernetes-Cluster mit der Version `1.18` oder früher erstellt wurde, fügt Amazon EKS das folgende Tag zu allen Subnetzen hinzu, die angegeben wurden.

Key (Schlüssel)	Value (Wert)
`kubernetes.io/cluster/` *`my-cluster`*	`shared`

Wenn Sie jetzt einen neuen Kubernetes-Cluster erstellen, fügt Amazon EKS das Tag nicht zu Ihren Subnetzen hinzu. Wenn sich das Tag in Subnetzen befand, die von einem Cluster verwendet wurden, der zuvor eine Version vor `1.19` war, wurde das Tag nicht automatisch aus den Subnetzen entfernt, als der Cluster auf eine neuere Version aktualisiert wurde. Für Version `2.1.1` oder früher des Load AWS Balancer Controllers ist dieses Tag erforderlich. Wenn Sie eine neuere Version des Load Balancer Controllers verwenden, können Sie das Tag entfernen, ohne Ihre Services zu unterbrechen. Weitere Informationen über den Controller finden Sie unter the section called “ AWS Load Balancer Balancer-Controller”.

Wenn Sie eine VPC mithilfe einer `eksctl` oder einer der Amazon AWS CloudFormation EKS-VPC-Vorlagen bereitgestellt haben, gilt Folgendes:

- Am oder nach dem 26. März 2020 – Öffentliche `IPv4`-Adressen werden von öffentlichen Subnetzen automatisch neuen Knoten zugewiesen, die in öffentlichen Subnetzen bereitgestellt werden.
- Vor dem 26. März 2020 – Öffentliche `IPv4`-Adressen werden von öffentlichen Subnetzen nicht automatisch neuen Knoten zugewiesen, die in öffentlichen Subnetzen bereitgestellt werden.

Diese Änderung wirkt sich folgendermaßen auf neue Knotengruppen aus, die in öffentlichen Subnetzen bereitgestellt werden:

- Verwaltete Knotengruppen – Wenn die Knotengruppe am oder nach dem 22. April 2020 in einem öffentlichen Subnetz bereitgestellt wird, muss für das öffentliche Subnetz die automatische Zuweisung öffentlicher IP-Adressen aktiviert sein. Weitere Informationen finden Sie unter Ändern des Public IPv4 Addressing-Attributs für Ihr Subnetz.
- Selbstverwaltete Linux-, Windows- oder Arm-Knotengruppen – Wenn die Knotengruppe am oder nach dem 26. März 2020 in einem öffentlichen Subnetz bereitgestellt wird, muss für das öffentliche Subnetz die automatische Zuweisung öffentlicher IP-Adressen aktiviert sein. Andernfalls müssen die Knoten mit einer öffentlichen IP-Adresse gestartet werden. Weitere Informationen finden Sie

unter Ändern des Attributs für die öffentliche IPv4 Adressierung für Ihr Subnetz oder Zuweisen einer öffentlichen IPv4 Adresse beim Instance-Start.

Anforderungen und Überlegungen für gemeinsam genutzte Subnetze

Sie können VPC Sharing verwenden, um Subnetze mit anderen AWS Konten innerhalb derselben AWS Organizations zu teilen. Sie können Amazon-EKS-Cluster in gemeinsam genutzten Subnetzen erstellen. Beachten Sie dabei die folgenden Überlegungen:

- Der Eigentümer des VPC-Subnetzes muss ein Subnetz für ein Teilnehmerkonto freigeben, bevor dieses Konto darin einen Amazon-EKS-Cluster erstellen kann.
- Sie können keine Ressourcen mit der Standardsicherheitsgruppe für die VPC starten, da diese dem Eigentümer gehört. Außerdem können Teilnehmer keine Ressourcen mit Sicherheitsgruppen starten, die anderen Teilnehmern oder dem Eigentümer gehören.
- In einem gemeinsam genutzten Subnetz kontrollieren der Teilnehmer und der Eigentümer die Sicherheitsgruppen innerhalb des jeweiligen Kontos separat. Der Subnetzbesitzer kann von den Teilnehmern erstellte Sicherheitsgruppen zwar anzeigen, jedoch keine Aktionen bei diesen durchführen. Wenn der Subnetzbesitzer diese Sicherheitsgruppen entfernen oder ändern möchte, muss der Teilnehmer, der die Sicherheitsgruppe erstellt hat, die Aktion durchführen.
- Wenn ein Cluster von einem Teilnehmer erstellt wird, gelten die folgenden Überlegungen:
  - Die Cluster-IAM-Rolle und die Knoten-IAM-Rollen müssen in diesem Konto erstellt werden. Weitere Informationen erhalten Sie unter the section called “Cluster-IAM-Rolle” und the section called “Knoten-IAM-Rolle”.
  - Alle Knoten müssen vom selben Teilnehmer erstellt werden, einschließlich verwalteter Knotengruppen.
- Der Eigentümer einer gemeinsam genutzten VPC kann einen Cluster, den ein Teilnehmer im gemeinsam genutzten Subnetz erstellt, nicht anzeigen, aktualisieren oder löschen. Dies gilt zusätzlich zu den VPC-Ressourcen, auf die jedes Konto unterschiedlich zugreifen kann. Weitere Informationen finden Sie unter Verantwortlichkeiten und Berechtigungen für Besitzer und Teilnehmer im Amazon-VPC-Benutzerhandbuch.
- Wenn Sie das Feature für das benutzerdefinierte Netzwerk des Amazon-VPC-CNI-Plugins für Kubernetes verwenden, müssen Sie die im Eigentümerkonto aufgeführten Zuordnungen der Availability-Zone-IDs verwenden, um jede `ENIConfig` zu erstellen. Weitere Informationen finden Sie unter the section called “Benutzerdefinierte Netzwerke”.

Weitere Informationen zur Freigabe von VPC-Subnetzen finden Sie unter Freigeben Ihrer VPC für andere Konten im Amazon-VPC-Benutzerhandbuch.

Erstellung einer Amazon VPC für Ihren Amazon-EKS-Cluster

Sie können Amazon Virtual Private Cloud (Amazon VPC) verwenden, um AWS Ressourcen in einem von Ihnen definierten virtuellen Netzwerk zu starten. Dieses virtuelle Netzwerk ist einem herkömmlichen Netzwerk, das Sie in Ihrem eigenen Rechenzentrum betreiben, sehr ähnlich. Es bietet jedoch die Vorzüge, die mit der Nutzung der skalierbaren Infrastruktur von Amazon Web Services einhergehen. Bevor Sie sich ein umfassendes Verständnis des Amazon-VPC-Service aneignet, empfehlen wir, dass Sie sich ein umfassendes Verständnis des Amazon-VPC-Service aneignet, bevor Sie sich ein umfassendes Verständnis aneignet. Weitere Informationen finden Sie im Amazon VPC-Benutzerhandbuch.

Ein Amazon-EKS-Cluster, Knoten und Kubernetes-Ressourcen werden auf einer VPC bereitgestellt. Wenn Sie eine vorhandene VPC mit Amazon EKS verwenden möchten, muss diese VPC die unter the section called “VPC- und Subnetz-Anforderungen” beschriebenen Voraussetzungen erfüllen. In diesem Thema wird beschrieben, wie Sie mithilfe einer von Amazon EKS bereitgestellten AWS CloudFormation Vorlage eine VPC erstellen, die die Amazon EKS-Anforderungen erfüllt. Sobald Sie eine Vorlage bereitgestellt haben, können Sie sich die mit der Vorlage erstellten Ressourcen ansehen, um genau zu erfahren, welche Ressourcen sie erstellt hat und wie diese Ressourcen konfiguriert sind. Wenn Sie Hybridknoten verwenden, muss Ihre VPC in ihrer Routing-Tabelle Routen für Ihr On-Premises-Netzwerk enthalten. Weitere Informationen zu den Netzwerk-Anforderungen für Hybridknoten finden Sie unter the section called “Netzwerk vorbereiten”.

Voraussetzungen

Um eine VPC für Amazon EKS zu erstellen, benötigen Sie die erforderlichen IAM-Berechtigungen, um Amazon-VPC-Ressourcen zu erstellen. Bei diesen Ressourcen handelt es sich VPCs um Subnetze, Sicherheitsgruppen, Routing-Tabellen und Routen sowie Internet- und NAT-Gateways. Weitere Informationen finden Sie unter Erstellen einer VPC mit einer Beispielrichtlinie für ein öffentliches Subnetz im Amazon-VPC-Benutzerhandbuch und in der vollständigen Liste der Aktionen in der Service-Autorisierungsreferenz.

Sie können eine VPC mit öffentlichen und privaten Subnetzen, nur öffentlichen Subnetzen oder nur privaten Subnetzen erstellen.

Öffentliche und private Subnetze

Diese VPC verfügt über zwei öffentliche und zwei private Subnetze. Eine Routing-Tabelle, die einem öffentlichen Subnetz zugeordnet ist, verfügt über eine Route zu einem Internet-Gateway. Die Routing-Tabelle eines privaten Subnetzes verfügt jedoch nicht über eine Route zu einem Internet-Gateway. Ein öffentliches und ein privates Subnetz werden in derselben Availability Zone bereitgestellt. Die anderen öffentlichen und privaten Subnetze werden in einer zweiten Availability Zone in derselben Region bereitgestellt. AWS Diese Option wird für die meisten Bereitstellungen empfohlen.

Mit dieser Option können Sie Ihre Knoten in privaten Subnetzen bereitstellen. Mit dieser Option können Sie es Kubernetes ermöglichen, Load Balancer in den öffentlichen Subnetzen bereitzustellen, die das Load Balancing für den Datenverkehr auf Pods, die auf Knoten in den privaten Subnetzen ausgeführt werden, durchführen können. Öffentliche `IPv4`-Adressen werden automatisch den Knoten zugewiesen, die in öffentlichen Subnetzen bereitgestellt werden, öffentliche `IPv4`-Adressen werden jedoch nicht Knoten zugewiesen, die in privaten Subnetzen bereitgestellt werden.

Sie können Knoten in öffentlichen und privaten Subnetzen auch `IPv6`-Adressen zuweisen. Die Knoten in privaten Subnetzen können mit dem Cluster und anderen AWS Diensten kommunizieren. Pods können über ein NAT-Gateway mit `IPv4`-Adressen oder über ein ausschließlich ausgehendes Internet-Gateway mit `IPv6`-Adressen, die in jeder Availability Zone bereitgestellt werden, mit dem Internet kommunizieren. Es wird eine Sicherheitsgruppe bereitgestellt, die über Regeln verfügt, die den gesamten eingehenden Datenverkehr von anderen Quellen als dem Cluster oder den Knoten ablehnen, jedoch den gesamten ausgehenden Datenverkehr zulassen. Die Subnetze sind getaggt, damit Kubernetes ihnen Load Balancer bereitstellen kann.

a. Öffnen Sie die AWS CloudFormation -Konsole.

b. Wählen Sie in der Navigationsleiste eine AWS Region aus, die Amazon EKS unterstützt.

c. Klicken Sie auf Create stack (Stack erstellen), With new resources (standard) (Mit neuen Ressourcen (Standard)).

d. Stellen Sie unter Vorraussetzung – Vorbereiten der Vorlage sicher, dass Vorlage ist bereit ausgewählt ist und wählen Sie dann unter Vorlage angeben Amazon-S3-URL aus.

e. Sie können eine VPC erstellen, die nur `IPv4` unterstützt, oder eine VPC, die `IPv4` und `IPv6` unterstützt. Fügen Sie eine der folgenden Optionen URLs in den Textbereich unter Amazon S3 S3-URL ein und wählen Sie Weiter aus:

   - `IPv4`

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml
```

- IPv4 und IPv6

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-ipv6-vpc-public-private-subnets.yaml
```

a. Ändern Sie auf der Seite Specify stack details (Stack-Details angeben) alle Parameter und wählen Sie dann Next (Weiter) aus.

   - Stack-Name: Wählen Sie einen Stack-Namen für Ihren AWS CloudFormation Stack. Sie können beispielsweise den im vorigen Schritt verwendeten Vorlagennamen nutzen. Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein.
   - VpcBlock: Wählen Sie einen IPv4 CIDR-Bereich für Ihre VPC. Jedem von Ihnen bereitgestellten Knoten, Pod und jedem Load Balancer wird eine IPv4-Adresse aus diesem Block zugewiesen. Die IPv4-Standardwerte bieten für die meisten Implementierungen genügend IP-Adressen, aber wenn dies nicht der Fall ist, können Sie ihn ändern. Weitere Informationen finden Sie unter Dimensionierung der VPC und der Subnetze im Amazon VPC Benutzerhandbuch. Sie können der VPC nach der Erstellung auch zusätzliche CIDR-Blöcke hinzufügen. Wenn Sie eine IPv6-VPC erstellen, werden Ihnen IPv6-CIDR-Bereiche automatisch aus dem globalen Unicast-Adressbereich von Amazon zugewiesen.
   - PublicSubnet01Block: Geben Sie einen IPv4 CIDR-Block für das öffentliche Subnetz 1 an. Der Standardwert bietet für die meisten Implementierungen genügend IP-Adressen, aber wenn dies nicht der Fall ist, können Sie ihn ändern. Wenn Sie eine IPv6-VPC erstellen, wird dieser Block in der Vorlage für Sie angegeben.
   - PublicSubnet02Block: Geben Sie einen IPv4 CIDR-Block für das öffentliche Subnetz 2 an. Der Standardwert bietet für die meisten Implementierungen genügend IP-Adressen, aber wenn dies nicht der Fall ist, können Sie ihn ändern. Wenn Sie eine IPv6-VPC erstellen, wird dieser Block in der Vorlage für Sie angegeben.
   - PrivateSubnet01Block: Geben Sie einen IPv4 CIDR-Block für das private Subnetz 1 an. Der Standardwert bietet für die meisten Implementierungen genügend IP-Adressen, aber wenn dies

nicht der Fall ist, können Sie ihn ändern. Wenn Sie eine `IPv6`-VPC erstellen, wird dieser Block in der Vorlage für Sie angegeben.

- PrivateSubnet02Block: Geben Sie einen `IPv4` CIDR-Block für das private Subnetz 2 an. Der Standardwert bietet für die meisten Implementierungen genügend IP-Adressen, aber wenn dies nicht der Fall ist, können Sie ihn ändern. Wenn Sie eine `IPv6`-VPC erstellen, wird dieser Block in der Vorlage für Sie angegeben.

b. (Optional) Auf der Seite Configure stack options (Stack-Optionen konfigurieren) können Sie Ihre Stack-Ressourcen markieren und dann Next (Weiter) auswählen.

c. Wählen Sie auf der Seite Prüfen Stack erstellen aus.

d. Wenn Ihr Stack erstellt wurde, wählen Sie ihn in der Konsole aus und klicken Sie auf Outputs (Ausgänge).

e. Notieren Sie das VpcIdfür die VPC, die erstellt wurde. Dies benötigen Sie, wenn Sie Ihren Cluster und Ihre Knoten erstellen.

f. Notieren Sie SubnetIdsfür die Subnetze, die erstellt wurden, und ob Sie sie als öffentliche oder private Subnetze erstellt haben. Sie benötigen mindestens zwei davon, wenn Sie Ihren Cluster und Ihre Knoten erstellen.

g. Wenn Sie eine `IPv4`-VPC erstellt haben, überspringen Sie diesen Schritt. Wenn Sie eine `IPv6`-VPC erstellt haben, müssen Sie die Option zum automatischen Zuweisen von `IPv6`-Adressen für die öffentlichen Subnetze aktivieren, die von der Vorlage erstellt wurden. Diese Einstellung ist für die privaten Subnetze bereits aktiviert. Führen Sie die folgenden Schritte aus, um die Einstellung zu aktivieren:

   i. Öffnen Sie die Amazon VPC-Konsole unter https://console.aws.amazon.com/vpc/.

   ii. Wählen Sie im linken Navigationsbereich die Option Subnets (Subnetze) aus.

   iii. Wählen Sie eines Ihrer öffentlichen Subnetze aus (***stack-name***/SubnetPublic01 oder ***stack-name***/SubnetPublic02 enthält das Wort public) und wählen Sie Aktionen, Subnetzeinstellungen bearbeiten.

   iv. Aktivieren Sie das Kontrollkästchen Automatische IPv6 Adresszuweisung aktivieren und wählen Sie dann Speichern.

   v. Wiederholen Sie die vorherigen Schritte für Ihr anderes öffentliches Subnetz.

Nur öffentliche Subnetze

Diese VPC hat drei öffentliche Subnetze, die in verschiedenen Availability Zones in einer AWS Region bereitgestellt werden. Allen Knoten werden automatisch öffentliche `IPv4`-Adressen

zugewiesen und alle Worker-Knoten können Internetdatenverkehr über ein Internet-Gateway senden und empfangen. Es wird eine Sicherheitsgruppe bereitgestellt, die den gesamten eingehenden Datenverkehr ablehnt und den gesamten ausgehenden Datenverkehr zulässt. Die Subnetze sind getaggt, damit Kubernetes ihnen Load Balancer bereitstellen kann.

a. Öffnen Sie die AWS CloudFormation -Konsole.

b. Wählen Sie in der Navigationsleiste eine AWS Region aus, die Amazon EKS unterstützt.

c. Klicken Sie auf Create stack (Stack erstellen), With new resources (standard) (Mit neuen Ressourcen (Standard)).

d. Stellen Sie unter Vorbereiten der Vorlage sicher, dass Vorlage ist bereit ausgewählt ist und wählen Sie dann unterVorlagenquelle Amazon-S3-URLaus.

e. Fügen Sie die folgende URL in den Textbereich unter Amazon-S3-URL ein und wählen Sie Weiter aus:

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-vpc-sample.yaml
```

a. Füllen Sie auf der Seite Specify Details (Details angeben) die Parameter entsprechend aus und klicken Sie dann auf Next (Weiter).

- Stack-Name: Wählen Sie einen Stack-Namen für Ihren AWS CloudFormation Stack. Sie können ihn beispielsweise *amazon-eks-vpc-sample* nennen. Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein.
- VpcBlock: Wählen Sie einen CIDR-Block für Ihre VPC. Jedem von Ihnen bereitgestellten Knoten, Pod und jedem Load Balancer wird eine `IPv4`-Adresse aus diesem Block zugewiesen. Die `IPv4`-Standardwerte bieten für die meisten Implementierungen genügend IP-Adressen, aber wenn dies nicht der Fall ist, können Sie ihn ändern. Weitere Informationen finden Sie unter Dimensionierung der VPC und der Subnetze im Amazon VPC Benutzerhandbuch. Sie können der VPC nach der Erstellung auch zusätzliche CIDR-Blöcke hinzufügen.
- Subnet01Block: Geben Sie einen CIDR-Block für das Subnetz 1 an. Der Standardwert bietet für die meisten Implementierungen genügend IP-Adressen, aber wenn dies nicht der Fall ist, können Sie ihn ändern.

- Subnet02Block: Geben Sie einen CIDR-Block für das Subnetz 2 an. Der Standardwert bietet für die meisten Implementierungen genügend IP-Adressen, aber wenn dies nicht der Fall ist, können Sie ihn ändern.
- Subnet03Block: Geben Sie einen CIDR-Block für das Subnetz 3 an. Der Standardwert bietet für die meisten Implementierungen genügend IP-Adressen, aber wenn dies nicht der Fall ist, können Sie ihn ändern.

b. (Optional) Markieren Sie auf der Seite Options (Optionen) Ihre Stack-Ressourcen. Wählen Sie Weiter aus.

c. Klicken Sie auf der Seite Review auf Create.

d. Wenn Ihr Stack erstellt wurde, wählen Sie ihn in der Konsole aus und klicken Sie auf Outputs (Ausgänge).

e. Notieren Sie das VpcIdfür die VPC, die erstellt wurde. Dies benötigen Sie, wenn Sie Ihren Cluster und Ihre Knoten erstellen.

f. Notieren Sie das SubnetIdsfür die Subnetze, die erstellt wurden. Sie benötigen mindestens zwei davon, wenn Sie Ihren Cluster und Ihre Knoten erstellen.

g. (Optional) Jeder Cluster, den Sie für diese VPC bereitstellen, kann Ihren Pods und Servicese private `IPv4`-Adressen zuweisen. Wenn Sie Cluster in dieser VPC bereitstellen möchten, um Ihren Pods und Services private `IPv6`-Adressen zuzuweisen, müssen Sie Aktualisierungen an Ihrer VPC, Ihrem Subnetz, Routing-Tabellen und Sicherheitsgruppen vornehmen. Weitere Informationen finden Sie unter Vorhandenes migrieren VPCs von IPv4 zu IPv6 im Amazon VPC-Benutzerhandbuch. Amazon EKS verlangt, dass bei Ihren Subnetzen die Option zum `Auto-assign` von `IPv6`-Adressen aktiviert ist. Dies ist standardmäßig deaktiviert.

Nur private Subnetze

Diese VPC hat drei private Subnetze, die in verschiedenen Availability Zones in der AWS Region bereitgestellt werden. Ressourcen, die in den Subnetzen bereitgestellt werden, können weder auf das Internet zugreifen, noch kann das Internet auf Ressourcen in den Subnetzen zugreifen. Die Vorlage erstellt VPC-Endpunkte, die AWS PrivateLink für mehrere AWS Dienste verwendet werden, auf die Knoten normalerweise zugreifen müssen. Wenn Ihre Knoten einen ausgehenden Internetzugang benötigen, können Sie ein öffentliches NAT-Gateway in der Availability Zone jedes Subnetzes hinzufügen, nachdem die VPC erstellt wurde. Eine Sicherheitsgruppe wird erstellt, der den gesamten eingehenden Datenverkehr verweigert, mit Ausnahme von Ressourcen, die in den Subnetzen bereitgestellt werden. Eine Sicherheitsgruppe lässt auch den gesamten ausgehenden Datenverkehr zu. Die Subnetze sind getaggt, damit Kubernetes ihnen interne Load Balancer bereitstellen kann.

Wenn Sie mit dieser Konfiguration eine VPC erstellen, finden Sie unter the section called “Private Cluster” weitere Anforderungen und Überlegungen.

a. Öffnen Sie die AWS CloudFormation -Konsole.
b. Wählen Sie in der Navigationsleiste eine AWS Region aus, die Amazon EKS unterstützt.
c. Klicken Sie auf Create stack (Stack erstellen), With new resources (standard) (Mit neuen Ressourcen (Standard)).
d. Stellen Sie unter Vorbereiten der Vorlage sicher, dass Vorlage ist bereit ausgewählt ist und wählen Sie dann unterVorlagenquelle Amazon-S3-URLaus.
e. Fügen Sie die folgende URL in den Textbereich unter Amazon-S3-URL ein und wählen Sie Weiter aus:

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-fully-private-vpc.yaml
```

a. Füllen Sie auf der Seite Specify Details (Details angeben) die Parameter entsprechend aus und klicken Sie dann auf Next (Weiter).
   - Stack-Name: Wählen Sie einen Stack-Namen für Ihren AWS CloudFormation Stack. Sie können ihn beispielsweise *amazon-eks-fully-private-vpc* nennen. Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein.
   - VpcBlock: Wählen Sie einen CIDR-Block für Ihre VPC. Jedem von Ihnen bereitgestellten Knoten, Pod und jedem Load Balancer wird eine `IPv4`-Adresse aus diesem Block zugewiesen. Die `IPv4`-Standardwerte bieten für die meisten Implementierungen genügend IP-Adressen, aber wenn dies nicht der Fall ist, können Sie ihn ändern. Weitere Informationen finden Sie unter Dimensionierung der VPC und der Subnetze im Amazon VPC Benutzerhandbuch. Sie können der VPC nach der Erstellung auch zusätzliche CIDR-Blöcke hinzufügen.
   - PrivateSubnet01Block: Geben Sie einen CIDR-Block für Subnetz 1 an. Der Standardwert bietet für die meisten Implementierungen genügend IP-Adressen, aber wenn dies nicht der Fall ist, können Sie ihn ändern.
   - PrivateSubnet02Block: Geben Sie einen CIDR-Block für Subnetz 2 an. Der Standardwert bietet für die meisten Implementierungen genügend IP-Adressen, aber wenn dies nicht der Fall ist, können Sie ihn ändern.

- PrivateSubnet03Block: Geben Sie einen CIDR-Block für Subnetz 3 an. Der Standardwert bietet für die meisten Implementierungen genügend IP-Adressen, aber wenn dies nicht der Fall ist, können Sie ihn ändern.

b. (Optional) Markieren Sie auf der Seite Options (Optionen) Ihre Stack-Ressourcen. Wählen Sie Weiter aus.

c. Klicken Sie auf der Seite Review auf Create.

d. Wenn Ihr Stack erstellt wurde, wählen Sie ihn in der Konsole aus und klicken Sie auf Outputs (Ausgänge).

e. Notieren Sie das VpcIdfür die VPC, die erstellt wurde. Dies benötigen Sie, wenn Sie Ihren Cluster und Ihre Knoten erstellen.

f. Notieren Sie das SubnetIdsfür die Subnetze, die erstellt wurden. Sie benötigen mindestens zwei davon, wenn Sie Ihren Cluster und Ihre Knoten erstellen.

g. (Optional) Jeder Cluster, den Sie für diese VPC bereitstellen, kann Ihren Pods und Servicese private `IPv4`-Adressen zuweisen. Wenn Sie Cluster in dieser VPC bereitstellen möchten, um Ihren Pods und Services private `IPv6`-Adressen zuzuweisen, müssen Sie Aktualisierungen an Ihrer VPC, Ihrem Subnetz, Routing-Tabellen und Sicherheitsgruppen vornehmen. Weitere Informationen finden Sie unter Vorhandenes migrieren VPCs von IPv4 zu IPv6 im Amazon VPC-Benutzerhandbuch. Amazon EKS erfordert, dass in Ihren Subnetzen die `Auto-assign IPv6`-Adressoption aktiviert ist (standardmäßig ist sie deaktiviert).

Anforderungen der Amazon-EKS-Sicherheitsgruppe für Cluster anzeigen

In diesem Thema geht es um die Anforderungen an Sicherheitsgruppen in einem Amazon-EKS-Cluster.

Standard-Cluster-Sicherheitsgruppe

Wenn Sie einen Cluster erstellen, erstellt Amazon EKS eine Sicherheitsgruppe mit dem Namen `eks-cluster-sg-`*`my-cluster`*`-`*`uniqueID`* . Diese Sicherheitsgruppe hat die folgenden Standardregeln:

Regeltyp	Protocol (Protokoll)	Ports	Quelle	Ziel
Eingehend	Alle	Alle	Selbst	
Ausgehend	Alle	Alle		0.0.0.0/0(IPv4) oder ::/0 (IPv6)
Ausgehend	Alle	Alle		Selbst (für EFA-Datenverkehr)

Die Standard-Sicherheitsgruppe enthält eine ausgehende Regel, die den Datenverkehr des Elastic Fabric Adapters (EFA) mit dem Ziel derselben Sicherheitsgruppe zulässt. Dadurch wird EFA-Datenverkehr innerhalb des Clusters ermöglicht, was für KI/ML- und High Performance Computing (HPC)-Workloads von Vorteil ist. Weitere Informationen finden Sie unter Elastic Fabric Adapter für KI-/ML- und HPC-Workloads in Amazon EC2 im Benutzerhandbuch für Amazon Elastic Compute Cloud.

> **Important**
>
> Wenn Ihr Cluster die ausgehende Regel nicht benötigt, können Sie sie entfernen. Wenn Sie es entfernen, müssen Sie immer noch über die Mindestregeln verfügen, die unter Einschränkung des Cluster-Datenverkehrs aufgeführt sind. Wenn Sie die eingehende Regel entfernen, wird sie von Amazon EKS bei jeder Aktualisierung des Clusters neu erstellt.

Amazon EKS fügt der Sicherheitsgruppe die folgenden Tags hinzu. Wenn Sie die Tags entfernen, fügt Amazon EKS sie bei jeder Aktualisierung Ihres Clusters wieder zur Sicherheitsgruppe hinzu.

Schlüssel	Value (Wert)
kubernetes.io/cluster/ *my-cluster*	owned
aws:eks:cluster-name	*my-cluster*
Name	eks-cluster-sg- *my-cluster* -*uniqueid*

Amazon EKS verknüpft die Sicherheitsgruppe automatisch mit den folgenden Ressourcen, die es ebenfalls erstellt:

- 2 bis 4 Elastic-Network-Schnittstellen (für den Rest dieses Dokuments als Netzwerkschnittstelle bezeichnet), die beim Anlegen des Clusters erstellt werden.
- Netzwerkschnittstellen der Knoten in jeder verwalteten Knotengruppe, die Sie erstellen.

Die Standardregeln ermöglichen es, dass der gesamte Datenverkehr frei zwischen Ihrem Cluster und Ihren Knoten fließt. Außerdem lassen sie den gesamten ausgehenden Datenverkehr zu jedem Ziel zu. Beim Erstellen eines Clusters können Sie (optional) Ihre eigenen Sicherheitsgruppen angeben. Wenn Sie das tun, verknüpft Amazon EKS auch die von Ihnen angegebenen Sicherheitsgruppen mit den Netzwerkschnittstellen, die es für Ihren Cluster erstellt. Er verknüpft sie jedoch mit keiner der Knotengruppen, die Sie erstellen.

Sie können die ID Ihrer Cluster-Sicherheitsgruppe in der AWS-Managementkonsole im Abschnitt Netzwerk für Ihren Cluster ermitteln. Sie können auch den folgenden AWS-CLI-Befehl ausführen.

```
aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId
```

Einschränken des Clusterverkehrs

Wenn Sie die offenen Ports zwischen dem Cluster und den Knoten einschränken müssen, können Sie die standardmäßige Ausgangsregel entfernen und die folgenden Mindestregeln hinzufügen, die für den Cluster erforderlich sind. Wenn Sie die standardmäßige Eingangsregel entfernen, erstellt Amazon EKS sie bei jeder Aktualisierung des Clusters neu.

Regeltyp	Protocol (Protokoll)	Port	Bestimmungsort
Ausgehend	TCP	443	Cluster-Sicherheitsgruppe
Ausgehend	TCP	10250	Cluster-Sicherheitsgruppe
Ausgehend (DNS)	TCP und UDP	53	Cluster-Sicherheitsgruppe

Sie müssen auch Regeln für den folgenden Datenverkehr hinzufügen:

- Jedes Protokoll und alle Ports, von dem Sie erwarten, dass Ihre Knoten für die Kommunikation zwischen Knoten verwenden
- Ausgehenden Internetzugang, sodass die Knoten zwecks Cluster-Introspektion und Knotenregistrierung beim Start auf die Amazon-EKS-APIs zugreifen können. Wenn Ihre Knoten keinen Internetzugang haben, lesen Sie den Abschnitt Bereitstellung privater Cluster mit eingeschränktem Internetzugang, um weitere Überlegungen zu berücksichtigen.
- Zugriff auf Knoten, um Container-Images aus Amazon ECR abzurufen, oder andere Container-Registry-APIs, die zum Laden von Images benötigt werden, wie z. B. DockerHub. Weitere Informationen finden Sie unter AWS-IP-Adressbereiche in der AWS Allgemeinen Referenz.
- Knotenzugriff auf Amazon S3.
- Für `IPv4`- und `IPv6`-Adressen sind separate Regeln erforderlich
- Wenn Sie Hybridknoten verwenden, müssen Sie Ihrem Cluster eine zusätzliche Sicherheitsgruppe hinzufügen, um die Kommunikation mit Ihren On-Premises-Knoten und Pods zu ermöglichen. Weitere Informationen finden Sie unter the section called “Netzwerk vorbereiten”.

Wenn Sie Regelbeschränkungen planen, empfehlen wir, alle Pods gründlich zu testen, bevor Sie Ihre geänderten Regeln auf einen Produktions-Cluster anwenden.

Wenn Sie ursprünglich einen Cluster mit Kubernetes `1.14` und der Plattformversion `eks.3` oder älter bereitgestellt haben, dann bedenken Sie Folgendes:

- Möglicherweise gibt es Sicherheitsgruppen für die Steuerebene und Knoten. Als diese Gruppen erstellt wurden, enthielten sie die in der vorherigen Tabelle aufgeführten eingeschränkten Regeln. Diese Sicherheitsgruppen sind nicht mehr erforderlich und können entfernt werden. Sie müssen jedoch sicherstellen, dass Ihre Cluster-Sicherheitsgruppe die Regeln enthält, die diese Gruppen enthalten.
- Wenn Sie den Cluster direkt mithilfe der API bereitgestellt haben oder ein Tool wie die AWS-CLI oder AWS CloudFormation zum Erstellen des Clusters verwendet haben und bei der Clustererstellung keine Sicherheitsgruppe angegeben haben, wurde die Standardsicherheitsgruppe für die VPC auf die von Amazon EKS erstellten Cluster-Netzwerkschnittstellen angewendet.

Freigegebene Sicherheitsgruppen

Amazon EKS unterstützt freigegebene Sicherheitsgruppen.

- Sicherheitsgruppen-VPC-Zuordnungen ordnen Sicherheitsgruppen mehreren VPCs im selben Konto und in derselben Region zu.
  - Erfahren Sie im Amazon-VPC-Benutzerhandbuch, wie Sie Sicherheitsgruppen mehreren VPCs zuordnen können.
- MIt Freigegebenen Sicherheitsgruppen können Sie Sicherheitsgruppen für andere AWS-Konten freigeben. Die Konten müssen sich in derselben AWS-Organisation befinden.
  - Erfahren Sie im Amazon-VPC-Benutzerhandbuch, wie Sie Sicherheitsgruppen mit Organisationen freigeben können.
- Sicherheitsgruppen sind immer auf eine einzelne AWS-Region beschränkt.

Überlegungen zu Amazon EKS

- EKS stellt dieselben Anforderungen an freigegebene oder Multi-VPC-Sicherheitsgruppen wie an Standard-Sicherheitsgruppen.

Verwaltung von Netzwerk-Add-Ons für Amazon-EKS-Cluster

Für Ihren Amazon-EKS-Cluster sind mehrere Netzwerk-Add-Ons verfügbar.

Integrierte Add-Ons

> **Note**
>
> Bei der Erstellung eines EKS-Clusters:
>
> - Verwenden der AWS Konsole: Die integrierten Add-Ons (wie CoreDNS, Kube-Proxy usw.) werden automatisch als Amazon EKS Add-ons installiert. Diese können einfach über die AWS Konsole, CLI oder aktualisiert SDKs werden.
> - Verwendung anderer Methoden (CLI SDKs usw.): Dieselben integrierten Add-Ons werden als selbstverwaltete Versionen installiert, die als reguläre Kubernetes-Bereitstellungen ausgeführt werden. Diese erfordern eine manuelle Konfiguration und Updates, da sie nicht über Tools verwaltet werden können. AWS

Wir empfehlen, Amazon EKS-Add-ons anstelle von selbstverwalteten Versionen zu verwenden, um die Add-On-Verwaltung zu vereinfachen und eine zentrale Konfiguration und Updates über AWS Services zu ermöglichen.

Amazon-VPC-CNI-Plugin für Kubernetes

Dieses CNI-Add-on erstellt elastische Netzwerkschnittstellen und verbindet sie mit Ihren EC2 Amazon-Knoten. Das Add-On weist außerdem jedem Pod und jedem Service eine private `IPv4`- oder `IPv6`-Adresse aus Ihrer VPC zu. Dieses Add-on ist standardmäßig auf Ihrem Cluster installiert. Weitere Informationen finden Sie unter the section called “Amazon VPC CNI”. Wenn Sie Hybridknoten verwenden, wird die VPC CNI weiterhin standardmäßig installiert, jedoch wird ihre Ausführung auf Ihren Hybridknoten durch eine Anti-Affinitätsregel verhindert. Weitere Informationen zu Ihren CNI-Optionen für Hybridknoten finden Sie unter the section called “CNI konfigurieren”.

CoreDNS

CoreDNS ist ein flexibler, erweiterbarer DNS-Server, der als Kubernetes-Cluster-DNS dienen kann. CoreDNS bietet die Namensauflösung für alle Pods im Cluster. Dieses Add-on ist standardmäßig auf Ihrem Cluster installiert. Weitere Informationen finden Sie unter the section called “CoreDNS”.

`kube-proxy`

Dieses Add-on verwaltet die Netzwerkregeln auf Ihren EC2 Amazon-Knoten und ermöglicht die Netzwerkkommunikation mit Ihren Pods. Dieses Add-on ist standardmäßig auf Ihrem Cluster installiert. Weitere Informationen finden Sie unter the section called “kube-proxy”.

Optionale AWS Netzwerk-Add-Ons

AWS Load Balancer Controller

Wenn Sie Kubernetes-Dienstobjekte des Typs bereitstellen`loadbalancer`, erstellt der Controller AWS Network Load Balancer. Wenn Sie Kubernetes-Eingangsobjekte erstellen, erstellt der Controller Application Load Balancer. AWS Wir empfehlen, diesen Controller für die Bereitstellung von Network Load Balancers zu verwenden, anstatt den in Kubernetes integrierten veralteten

Cloud Provider-Controller zu nutzen. Weitere Informationen finden Sie unter Ressourcen-Tags in der AWS -Load-Balancer-Controller-Dokumentation.

AWS -Gateway-API-Controller

Mit diesem Controller können Sie mithilfe der Kubernetes-Gateway-API Services über mehrere Kubernetes-Cluster hinweg verbinden. Der Controller verbindet Kubernetes-Services, die auf EC2 Amazon-Instances, Containern und serverlosen Funktionen ausgeführt werden, mithilfe des Amazon VPC Lattice-Dienstes. Weitere Informationen finden Sie in der Dokumentation zu AWS -Gateway-API-Controller.

Weitere Informationen zu Add-ons finden Sie unter the section called “Amazon-EKS-Add-ons”.

Pods mit dem Amazon VPC CNI zuweisen IPs

> **Tip**
>
> Melden Sie sich für bevorstehende Amazon EKS-Workshops an.

> **Tip**
>
> Mit Amazon EKS Auto Mode ist es nicht erforderlich, Netzwerk-Add-Ons zu installieren oder zu aktualisieren. Der Automatikmodus umfasst Funktionen für Pod-Netzwerke und Load Balancing.
> Weitere Informationen finden Sie unter EKS Auto Mode.

Das Amazon VPC CNI-Plug-In für Kubernetes wird auf jedem EC2 Amazon-Knoten in Ihrem Amazon EKS-Cluster bereitgestellt. Das Add-on erstellt elastische Netzwerkschnittstellen und verbindet sie mit Ihren EC2 Amazon-Knoten. Das Add-on weist jedem Pod außerdem eine private `IPv4`- oder `IPv6`-Adresse aus Ihrem VPC zu.

Eine Version des Add-Ons wird mit jedem Fargate-Knoten in Ihrem Cluster bereitgestellt, aber Sie aktualisieren sie nicht auf Fargate-Knoten. Andere kompatible CNI-Plug-ins sind für die Verwendung auf Amazon EKS-Clustern verfügbar, aber dies ist das einzige CNI-Plugin, das von Amazon EKS für Knoten unterstützt wird, die auf der AWS Infrastruktur ausgeführt werden. Weitere Informationen zu den anderen kompatiblen CNI-Plugins finden Sie unter the section called “Abweichende CNI-Plugins”.

Die Verwendung des VPC CNI mit Hybridknoten wird nicht unterstützt. Weitere Informationen zu Ihren CNI-Optionen für Hybridknoten finden Sie unter the section called “CNI konfigurieren”.

In der folgenden Tabelle finden Sie die neueste Version des Amazon-EKS-Add-Ons, die für jede Kubernetes-Version verfügbar ist.

Amazon-VPC-CNI-Versionen

Kubernetes-Version	VPC-CNI-Version vom Typ Amazon EKS
1.34	v1.20.4-eksbuild.1
1,33	v1.20.4-eksbuild.1
1.32	v1.20.4-eksbuild.1
1,31	v1.20.4-eksbuild.1
1,30	v1.20.4-eksbuild.1
1,29	v1.20.4-eksbuild.1
1,28	v1.20.4-eksbuild.1

Important

Wenn Sie dieses Add-On selbst verwalten, stimmen die Versionen in der Tabelle möglicherweise nicht mit den verfügbaren selbstverwalteten Versionen überein. Weitere Hinweise zur Aktualisierung des selbstverwalteten Typs dieses Add-ons finden Sie unter the section called “Aktualisierung (selbstverwaltet)”.

Important

Um ein Upgrade auf VPC CNI v1.12.0 oder höher durchzuführen, ist es erforderlich, zunächst ein Upgrade auf VPC CNI v1.7.0 durchzuführen. Wir empfehlen Ihnen, jeweils eine Nebenversion zu aktualisieren.

Überlegungen

Nachfolgend finden Sie einige Überlegungen zur Verwendung dieses Features.

- Versionen sind angegeben als `major-version.minor-version.patch-version-eksbuild.build-number`.
- Überprüfen Sie die Versionskompatibilität für jedes Feature. Einige Features jeder Version des Amazon-VPC-CNI-Plugins für Kubernetes erfordern bestimmte Kubernetes-Versionen. Wenn Sie verschiedene Amazon-EKS-Features verwenden und eine bestimmte Version des Add-Ons erforderlich ist, ist dies in der Feature-Dokumentationen vermerkt. Falls Sie keinen bestimmten Grund haben, eine frühere Version auszuführen, empfehlen wir, die neueste Version auszuführen.

Amazon-VPC-CNI (Amazon-EKS-Add-On) erstellen

Führen Sie die folgenden Schritte aus, um das Amazon-VPC-CNI-Plugin für das Kubernetes Amazon-EKS-Add-On zu erstellen.

Lesen Sie die Überlegungen noch einmal durch, bevor Sie beginnen. Weitere Informationen finden Sie unter the section called “Überlegungen”.

Voraussetzungen

Die folgenden Voraussetzungen müssen für das Amazon-VPC-CNI-Plugin für das Kubernetes Amazon-EKS-Add-On erfüllt sein.

- Ein vorhandener Amazon-EKS-Cluster. Informationen zum Bereitstellen finden Sie unter Erste Schritte.
- Ein vorhandener AWS Identity and Access Management (IAM) OpenID Connect (OIDC) -Anbieter für Ihren Cluster. Informationen zum Feststellen, ob Sie bereits über einen verfügen oder einen erstellen müssen, finden Sie unter the section called “IAM-OIDC-Anbieter”.
- Eine IAM-Rolle, an die die IAM-Richtlinie Amazoneks_CNI_Policy (wenn Ihr Cluster die Familie verwendet) oder eine IPv6 Richtlinie (wenn Ihr Cluster die `IPv4` Familie verwendet) angehängt ist. `IPv6` Weitere Informationen zur VPC CNI-Rolle finden Sie unter the section called “Konfiguration für IRSA”. Informationen zu der Richtlinie finden Sie unter. IPv6 the section called “Erstellen Sie eine IAM-Richtlinie für Cluster, die die `IPv6`-Familie verwendet”

> **Important**
>
> Das Amazon-VPC-CNI-Plugin für Kubernetes-Versionen `v1.16.0` bis `v1.16.1` implementiert die CNI-Spezifikationsversion `v1.0.0`. Weitere Informationen `v1.0.0` zur CNI-Spezifikation finden Sie unter Container Network Interface (CNI) -Spezifikation unter. GitHub

Verfahren

Nachdem Sie die Voraussetzungen erfüllt haben, führen Sie die folgenden Schritte aus, um das Add-On zu erstellen.

1. Sehen Sie, welche Version des Container-Images derzeit auf Ihrem Cluster installiert ist.

   ```
   kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni: |
    cut -d : -f 3
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   v1.16.4-eksbuild.2
   ```

2. Sehen Sie, welche Version des Container-Images derzeit auf Ihrem Cluster installiert ist. Je nachdem, mit welchem Tool Sie Ihr Cluster erstellt haben, ist der Add-on vom Typ Amazon EKS möglicherweise derzeit nicht auf Ihrem Cluster installiert. Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
    addon.addonVersion --output text
   ```

   Wenn eine Versionsnummer zurückgegeben wird, ist das Add-On vom Typ Amazon EKS auf Ihrem Cluster installiert und Sie müssen die restlichen Schritte in diesem Verfahren nicht ausführen. Wenn ein Fehler zurückgegeben wird, ist der Amazon-EKS-Typ des Add-Ons nicht auf Ihrem Cluster installiert. Führen Sie die verbleibenden Schritte dieses Prozesses aus, um es zu installieren.

3. Speichern Sie die Konfiguration Ihres aktuell installierten Add-ons ab.

   ```
   kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
   ```

4. Erstellen Sie das Add-on mit der AWS CLI. Wenn Sie das AWS-Managementkonsole oder verwenden möchten, um das Add-on `eksctl` zu erstellen, finden Sie weitere Informationen unter the section called “Add-On erstellen” und geben Sie `vpc-cni` den Namen des Add-ons an. Kopieren Sie den folgenden Befehl auf Ihr Gerät. Nehmen Sie bei Bedarf die folgenden Änderungen am Befehl vor, und führen Sie dann den geänderten Befehl aus.
   - Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters.
   - *`v1.20.3-eksbuild.1`*Ersetzen Sie es durch die neueste Version, die in der Tabelle mit den neuesten Versionen für Ihre Clusterversion aufgeführt ist. Die neueste Versionstabelle finden Sie unter the section called “Amazon-VPC-CNI-Versionen”.
   - *`111122223333`*Ersetzen Sie durch Ihre Konto-ID und *`AmazonEKSVPCCNIRole`* den Namen einer vorhandenen IAM-Rolle, die Sie erstellt haben. Zur Angabe einer Rolle ist es erforderlich, dass Sie über einen IAM OpenID Connect (OIDC)-Anbieter für Ihren Cluster verfügen. Um festzustellen, ob Sie über einen solchen für Ihren Cluster verfügen, oder um einen zu erstellen, lesen Sie the section called “IAM-OIDC-Anbieter”.

   ```
   aws eks create-addon --cluster-name my-cluster --addon-name vpc-cni --addon-version
    v1.20.3-eksbuild.1 \
       --service-account-role-arn arn:aws: iam::111122223333:role/AmazonEKSVPCCNIRole
   ```

   Wenn Sie benutzerdefinierte Einstellungen auf Ihr aktuelles Add-on angewendet haben, die mit den Standardeinstellungen des Amazon-EKS-Add-Ons in Konflikt stehen, schlägt die Erstellung möglicherweise fehl. Wenn die Erstellung fehlschlägt, erhalten Sie eine Fehlermeldung, die Sie bei der Problembehebung unterstützt. Alternativ können Sie den vorherigen Befehl mit `--resolve-conflicts OVERWRITE` ergänzen. Dadurch kann das Add-on alle vorhandenen benutzerdefinierten Einstellungen überschreiben. Sobald Sie das Add-On erstellt haben, können Sie es mit Ihren benutzerdefinierten Einstellungen aktualisieren.
5. Überprüfen Sie, ob die neueste Version des Add-Ons für die Kubernetes-Version Ihres Clusters zu Ihrem Cluster hinzugefügt wurde. Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
    addon.addonVersion --output text
   ```

   Es kann einige Sekunden dauern, bis die Erstellung des Add-ons abgeschlossen ist.

   Eine Beispielausgabe sieht wie folgt aus.

```
v1.20.3-eksbuild.1
```

6. Wenn Sie vor der Erstellung des Amazon-EKS-Add-Ons benutzerdefinierte Einstellungen an Ihrem ursprünglichen Add-on vorgenommen haben, verwenden Sie die im vorherigen Schritt gespeicherte Konfiguration, um das EKS-Add-On mit Ihren benutzerdefinierten Einstellungen zu aktualisieren. Führen Sie die Schritte unter the section called “Aktualisierung (EKS-Add-On)” aus.
7. (Optional) Installieren Sie `cni-metrics-helper` in Ihrem Cluster. Es erfasst elastic network interface- und IP-Adressinformationen, aggregiert sie auf Clusterebene und veröffentlicht die Metriken auf Amazon. CloudWatch Weitere Informationen finden Sie unter cni-metrics-helper. GitHub

Aktualisieren der Amazon VPC CNI (Amazon-EKS-Add-On)

Aktualisieren Sie den Amazon-EKS-Typ des Amazon-VPC-CNI-Plugins für das Kubernetes-Add-On. Wenn Sie den Amazon-EKS-Typ des Add-Ons nicht zu Ihrem Cluster hinzugefügt haben, können Sie ihn installieren, indem Sie the section called “Create” befolgen. Alternativ können Sie die andere Art der VPC-CNI-Installation aktualisieren, indem Sie the section called “Aktualisierung (selbstverwaltet)” befolgen.

1. Sehen Sie, welche Version des Container-Images derzeit auf Ihrem Cluster installiert ist. Ersetzen Sie *`my-cluster`* mit Ihrem Clusternamen.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
    "addon.addonVersion" --output text
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   v1.20.0-eksbuild.1
   ```

   Vergleichen Sie die Version mit der Tabelle der neuesten Versionen unter the section called “Amazon-VPC-CNI-Versionen”. Wenn die zurückgegebene Version mit der Version für die Kubernetes-Version Ihres Clusters in der aktuellen Versionstabelle übereinstimmt, haben Sie die neueste Version bereits auf Ihrem Cluster installiert und müssen den Rest dieses Verfahrens nicht abschließen. Wenn Sie eine Fehlermeldung statt einer Versionsnummer in Ihrer Ausgabe erhalten, haben Sie den Add-On vom Typ Amazon EKS nicht auf Ihrem Cluster installiert. Sie müssen das

Add-on erstellen, bevor Sie es mit diesem Verfahren aktualisieren können. Um den Amazon-EKS-Typ des VPC-CNI-Add-Ons zu erstellen, befolgen Sie the section called “Create”.

2. Speichern Sie die Konfiguration Ihres aktuell installierten Add-ons ab.

```
kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
```

3. Aktualisieren Sie Ihr Add-on mit der AWS CLI. Wenn Sie das AWS-Managementkonsole oder verwenden möchten, um das Add-on `eksctl` zu aktualisieren, finden Sie weitere Informationen unterthe section called “Add-On aktualisieren”. Kopieren Sie den folgenden Befehl auf Ihr Gerät. Nehmen Sie bei Bedarf die folgenden Änderungen am Befehl vor, und führen Sie dann den geänderten Befehl aus.
   - Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters.
   - *`v1.20.0-eksbuild.1`*Ersetzen Sie es durch die neueste Version, die in der Tabelle mit den neuesten Versionen für Ihre Clusterversion aufgeführt ist.
   - *`111122223333`*Ersetzen Sie durch Ihre Konto-ID und *`AmazonEKSVPCCNIRole`* den Namen einer vorhandenen IAM-Rolle, die Sie erstellt haben. Informationen zum Erstellen einer IAM-Rolle für das VPC CNI finden Sie unter the section called “Schritt 1: Amazon-VPC-CNI-Plugin für die Kubernetes-IAM-Rolle erstellen”. Zur Angabe einer Rolle ist es erforderlich, dass Sie über einen IAM OpenID Connect (OIDC)-Anbieter für Ihren Cluster verfügen. Um festzustellen, ob Sie über einen solchen für Ihren Cluster verfügen, oder um einen zu erstellen, lesen Sie the section called “IAM-OIDC-Anbieter”.
   - Die `--resolve-conflicts PRESERVE`-Option behält die vorhandenen Konfigurationswerte für das Add-On bei. Wenn Sie benutzerdefinierte Werte für Zusatzeinstellungen festgelegt haben und diese Option nicht verwenden, überschreibt Amazon EKS Ihre Werte mit seinen Standardwerten. Wenn Sie diese Option verwenden, empfehlen wir, dass Sie alle Feld- und Wertänderungen auf einem Nicht-Produktionscluster testen, bevor Sie das Add-on auf Ihrem Produktionscluster aktualisieren. Wenn Sie diesen Wert auf `OVERWRITE` ändern, werden alle Einstellungen auf die Amazon-EKS-Standardwerte geändert. Wenn Sie benutzerdefinierte Werte für Einstellungen festgelegt haben, werden diese möglicherweise mit den Amazon-EKS-Standardwerten überschrieben. Wenn Sie diesen Wert auf `none` ändern, ändert Amazon EKS den Wert der Einstellungen nicht, aber die Aktualisierung schlägt möglicherweise fehl. Wenn das Update fehlschlägt, erhalten Sie eine Fehlermeldung, die Sie bei der Behebung des Konflikts unterstützt.
   - Wenn Sie eine Konfigurationseinstellung nicht aktualisieren, entfernen Sie `--configuration-values '{`*`"env":{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}`*`}'` aus dem Befehl. Wenn Sie eine Konfigurationseinstellung aktualisieren, *`"env":`*

{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"} ersetzen Sie sie durch die Einstellung, die Sie festlegen möchten. In diesem Beispiel ist die Umgebungsvariable `AWS_VPC_K8S_CNI_EXTERNALSNAT` auf `true` gesetzt. Der von Ihnen angegebene Wert muss für das Konfigurationsschema gültig sein. Wenn Sie das Konfigurationsschema nicht kennen`aws eks describe-addon-configuration --addon-name vpc-cni --addon-version` *v1.20.0-eksbuild.1* , führen Sie es aus und *v1.20.0-eksbuild.1* ersetzen Sie es durch die Versionsnummer des Add-ons, für das Sie die Konfiguration sehen möchten. Das Schema wird in der Ausgabe zurückgegeben. Wenn Sie über eine bestehende benutzerdefinierte Konfiguration verfügen, diese alle entfernen und die Werte für alle Einstellungen auf die Amazon EKS-Standardwerte zurücksetzen möchten, entfernen Sie sie *"env":{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}* aus dem Befehl, sodass Sie leer {} haben. Eine Erläuterung der einzelnen Einstellungen finden Sie unter CNI-Konfigurationsvariablen unter. GitHub

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-version
 v1.20.3-eksbuild.1 \
    --service-account-role-arn arn:aws: iam::111122223333:role/AmazonEKSVPCCNIRole
 \
    --resolve-conflicts PRESERVE --configuration-values '{"env":
{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}}'
```

Es kann einige Sekunden dauern, bis das Update abgeschlossen ist.

4. Vergewissern Sie sich, dass die Add-on-Version aktualisiert wurde. Ersetzen Sie *my-cluster* mit dem Namen Ihres Clusters.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni
```

Es kann einige Sekunden dauern, bis das Update abgeschlossen ist.

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "addon": {
        "addonName": "vpc-cni",
        "clusterName": "my-cluster",
        "status": "ACTIVE",
        "addonVersion": "v1.20.3-eksbuild.1",
        "health": {
            "issues": []
```

```
        },
        "addonArn": "arn:aws: eks:region:111122223333:addon/my-cluster/vpc-
cni/74c33d2f-b4dc-8718-56e7-9fdfa65d14a9",
        "createdAt": "2023-04-12T18:25:19.319000+00:00",
        "modifiedAt": "2023-04-12T18:40:28.683000+00:00",
        "serviceAccountRoleArn": "arn:aws: iam::111122223333:role/
AmazonEKSVPCCNIRole",
        "tags": {},
        "configurationValues": "{\"env\":{\"AWS_VPC_K8S_CNI_EXTERNALSNAT\":\"true
\"}}"
    }
}
```

Fehlerbehebung

Wenn Sie das VPC-CNI von einer Version aktualisieren, die älter als v1.13.2 ist, müssen Sie nach dem Update alle Knoten im Cluster ersetzen. Versionen vor v1.13.2 verwenden das iptables-Legacy-Backend, um iptables-Regeln einzufügen, die für eine einwandfreie Funktionalität erforderlich sind, z. B. Source NAT (SNAT).

Version v1.13.2 war eine wichtige Veröffentlichung, die den iptables-wrapper einführte, der automatisch das passende iptables-Backend (iptables-legacy oder iptables-nft) zum Einfügen von Ketten und Regeln erkennt. Diese Änderung stand im Einklang mit der Entscheidung von Upstream Kubernetes, sich aufgrund von Leistungseinschränkungen vom Legacy-Backend zu entfernen.

Das Ersetzen von Knoten nach einem Upgrade von einer Version, die älter als v1.13.2 des VPC CNI ist, erforderlich, da die Einführung von Regeln sowohl in die iptables-legacy- als auch in die iptables-nft-Backends zu unerwartetem Verhalten bei Datenverkehr führen kann, der von nicht primär stammt. ENIs

Aktualisierung der Amazon VPC CNI (selbstverwaltetes Add-On)

> **Important**
>
> Wir empfehlen, den Amazon-EKS-Typ des Add-Ons zu Ihrem Cluster hinzuzufügen, anstatt den selbstverwalteten Typ des Add-Ons zu verwenden. Wenn Sie noch keine Erfahrung mit den Unterschieden zwischen den Typen haben, finden Sie weitere Informationen unter the section called “Amazon-EKS-Add-ons”. Weitere Informationen zum Hinzufügen eines Amazon-EKS-Add-ons zu Ihrem Cluster finden Sie unter the section called “Add-On erstellen”. Wenn Sie das Amazon EKS-Add-on nicht verwenden können, empfehlen

wir Ihnen, ein Problem mit der Begründung, warum Sie das nicht können, an das GitHub Container-Roadmap-Repository zu senden.

1. Vergewissern Sie sich, dass Sie nicht den Amazon-EKS-Typ des Add-Ons in Ihrem Cluster installiert haben. Ersetzen Sie *my-cluster* mit dem Namen Ihres Clusters.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

   Sollte eine Fehlermeldung angezeigt werden, ist der Amazon-EKS-Typ des Add-Ons nicht in Ihrem Cluster installiert. Um das Add-on selbst zu verwalten, führen Sie die verbleibenden Schritte in diesem Verfahren aus, um das Add-on zu aktualisieren. Wenn Sie eine Versionsnummer zurückgeben, wird der Amazon-EKS-Typ des Add-Ons auf Ihrem Cluster installiert. Verwenden Sie zum Aktualisieren das Verfahren in the section called “Add-On aktualisieren” und nicht das Verfahren in diesem Thema. Wenn Sie mit den Unterschieden zwischen den Add-On-Typen nicht vertraut sind, finden Sie Informationen unter the section called “Amazon-EKS-Add-ons”.
2. Sehen Sie, welche Version des Container-Images derzeit auf Ihrem Cluster installiert ist.

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni: |
 cut -d : -f 3
```

   Eine Beispielausgabe sieht wie folgt aus.

```
v1.20.0-eksbuild.1
```

   Ihre Ausgabe enthält möglicherweise nicht die Build-Nummer.
3. Sichern Sie Ihre aktuellen Einstellungen, damit Sie dieselben Einstellungen konfigurieren können, wenn Sie Ihre Version aktualisiert haben.

```
kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
```

   Eine Übersicht über die verfügbaren Versionen und die Änderungen in der Version, auf die Sie aktualisieren möchten, finden Sie unter Veröffentlichungen am GitHub. Beachten Sie, dass wir empfehlen, auf dieselbe Version zu aktualisieren`major`. `minor`. `patch`Version, die in der Tabelle mit den neuesten verfügbaren Versionen aufgeführt ist, auch wenn spätere Versionen verfügbar sind GitHub. Die aktuellste verfügbare Versionstabelle finden Sie unter the section called

"Amazon-VPC-CNI-Versionen". Die in der Tabelle aufgeführten Buildversionen sind in den unter GitHub aufgeführten selbstverwalteten Versionen nicht angegeben. Aktualisieren Sie Ihre Version, indem Sie die Aufgaben mit einer der folgenden Optionen ausführen:

- Wenn Sie keine benutzerdefinierten Einstellungen für das Add-on haben, führen Sie den Befehl unter der `To apply this release:` Überschrift GitHub für die Version aus, auf die Sie aktualisieren möchten.
- Wenn Sie benutzerdefinierte Einstellungen haben, laden Sie die Manifestdatei mit dem folgenden Befehl herunter, anstatt sie anzuwenden. Ändern *`https://raw.githubusercontent.com/aws/amazon-vpc-cni-k8s/v1.20.0/config/master/aws-k8s-cni.yaml`* Sie die URL für die Version GitHub , auf die Sie aktualisieren möchten.

```
curl -O https://raw.githubusercontent.com/aws/amazon-vpc-cni-k8s/v1.20.3/config/
master/aws-k8s-cni.yaml
```

Ändern Sie bei Bedarf die Datei mit den benutzerdefinierten Einstellungen aus dem von Ihnen erstellten Backup und wenden Sie die geänderte Datei dann auf Ihren Cluster an. Wenn Ihre Knoten keinen Zugriff auf die privaten Repositorys für Amazon EKS Amazon ECR haben, aus denen die Images abgerufen werden (siehe die Zeilen, die im Manifest mit `image:` beginnen), müssen Sie die Images herunterladen, sie in Ihr eigenes Repository kopieren und das Manifest so ändern, dass die Images aus Ihrem Repository abgerufen werden. Weitere Informationen finden Sie unter the section called "Kopieren eines Images in ein Repository".

```
kubectl apply -f aws-k8s-cni.yaml
```

4. Vergewissern Sie sich, dass die neue Version jetzt auf Ihrem Cluster installiert ist.

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni: |
 cut -d : -f 3
```

Eine Beispielausgabe sieht wie folgt aus.

```
v1.20.3
```

5. (Optional) Installieren Sie `cni-metrics-helper` in Ihrem Cluster. Es erfasst elastic network interface- und IP-Adressinformationen, aggregiert sie auf Clusterebene und veröffentlicht die Metriken auf Amazon. CloudWatch Weitere Informationen finden Sie unter cni-metrics-helper. GitHub

Konfiguration des Amazon-VPC-CNI-Plugins für die Verwendung von IRSA

Das Amazon VPC CNI-Plug-in für Kubernetes ist das Netzwerk-Plug-In für Pod-Netzwerke in Amazon-EKS-Clustern. Das Plugin ist für die Zuweisung von VPC-IP-Adressen zu Kubernetes-Pods und die Konfiguration der erforderlichen Vernetzung für Pods auf jedem Knoten verantwortlich.

> **Note**
>
> Das Amazon-VPC-CNI-Plugin unterstützt auch Amazon EKS Pod Identities. Weitere Informationen finden Sie unter the section called “IAM-Rolle zuweisen”.

Das Plugin:

- Erfordert AWS Identity and Access Management (IAM) -Berechtigungen. Wenn Ihr Cluster die `IPv4` Familie verwendet, sind die Berechtigungen in der `AmazonEKS_CNI_Policy` AWS verwalteten Richtlinie angegeben. Wenn Ihr Cluster die `IPv6`-Familie verwendet, müssen die Berechtigungen einer von Ihnen erstellten IAM-Richtlinie hinzugefügt werden. Anweisungen finden Sie unter the section called “Erstellen Sie eine IAM-Richtlinie für Cluster, die die `IPv6`-Familie verwendet”. Sie können diese Richtlinie der Amazon-EKS-Knoten-IAM-Rolle oder einer separaten IAM-Rolle anfügen. Anweisungen zum Anfügen der Richtlinie an die IAM-Rolle des Amazon-EKS-Knotens finden Sie unter the section called “Knoten-IAM-Rolle”. Wir empfehlen Ihnen, sie einer separaten Rolle zuzuweisen, wie in diesem Thema beschrieben.
- Erstellt und konfiguriert ein Kubernetes-Servicekonto mit dem Namen `aws-node`, wenn es bereitgestellt wird. Das Servicekonto ist an ein Kubernetes `clusterrole` namens `aws-node` gebunden, dem die erforderlichen Kubernetes-Berechtigungen zugewiesen sind.

> **Note**
>
> Die Pods für das Amazon-VPC-CNI-Plugin für Kubernetes haben Zugriff auf die der IAM-Rolle des Amazon-EKS-Knotens zugewiesenen Berechtigungen, sofern Sie den Zugriff auf IMDS nicht blockieren Weitere Informationen finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.

- Erfordert einen vorhandenen Amazon-EKS-Cluster. Informationen zum Bereitstellen finden Sie unter Erste Schritte.

- Erfordert einen vorhandenen AWS Identity and Access Management (IAM) OpenID Connect (OIDC) -Anbieter für Ihren Cluster. Informationen zum Feststellen, ob Sie bereits über einen verfügen oder einen erstellen müssen, finden Sie unter the section called “IAM-OIDC-Anbieter”.

Schritt 1: Amazon-VPC-CNI-Plugin für die Kubernetes-IAM-Rolle erstellen

1. Ermitteln Sie die IP-Familie Ihres Clusters.

   ```
   aws eks describe-cluster --name my-cluster | grep ipFamily
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   "ipFamily": "ipv4"
   ```

   Die Ausgabe kann stattdessen `ipv6` zurückgeben.

2. Erstellen Sie die IAM-Rolle. Sie können `eksctl` oder `kubectl` und die AWS CLI verwenden, um Ihre IAM-Rolle zu erstellen.

   eksctl

   - Erstellen Sie eine IAM-Rolle und fügen Sie der Rolle die IAM-Richtlinie mit dem Befehl zu, der der IP-Familie Ihres Clusters entspricht. Der Befehl erstellt und stellt einen AWS CloudFormation Stack bereit, der eine IAM-Rolle erstellt, ihr die von Ihnen angegebene Richtlinie anhängt und das vorhandene `aws-node` Kubernetes-Dienstkonto mit dem ARN der erstellten IAM-Rolle annotiert.

     - `IPv4`

       Ersetzen Sie *`my-cluster`* durch Ihren eigenen Wert.

       ```
       eksctl create iamserviceaccount \
           --name aws-node \
           --namespace kube-system \
           --cluster my-cluster \
           --role-name AmazonEKSVPCCNIRole \
           --attach-policy-arn arn:aws: iam::aws:policy/AmazonEKS_CNI_Policy \
           --override-existing-serviceaccounts \
           --approve
       ```

     - `IPv6`

Ersetzen Sie *my-cluster* durch Ihren eigenen Wert. Ersetzen Sie *111122223333* durch den Namen Ihres Clusters und *AmazonEKS_CNI_IPv6_Policy* durch Ihre IPv6-Richtlinie. Wenn Sie noch keine IPv6-Richtlinie haben, lesen Sie the section called "Erstellen Sie eine IAM-Richtlinie für Cluster, die die IPv6-Familie verwendet", um eine zu erstellen. Um IPv6 mit Ihrem Cluster zu verwenden, muss dieser mehrere Anforderungen erfüllen. Weitere Informationen finden Sie unter the section called "IPv6".

```
eksctl create iamserviceaccount \
    --name aws-node \
    --namespace kube-system \
    --cluster my-cluster \
    --role-name AmazonEKSVPCCNIRole \
    --attach-policy-arn arn:aws: iam::111122223333:policy/
AmazonEKS_CNI_IPv6_Policy \
    --override-existing-serviceaccounts \
    --approve
```

kubectl und die CLI AWS

i. Zeigen Sie die OIDC-Anbieter-URL Ihres Clusters an.

```
aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text
```

Eine Beispielausgabe sieht wie folgt aus.

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

Wenn keine Ausgabe erfolgt, müssen Sie einen IAM-OIDC-Anbieter für Ihr Cluster erstellen.

ii. Kopieren Sie den folgenden Inhalt in eine Datei namens *vpc-cni-trust-policy.json*. Ersetzen Sie *111122223333* durch die ID Ihres Kontos und *EXAMPLED539D4633E53DE1B71EXAMPLE* durch den Wert, der im vorherigen Schritt zurückgegeben wurde. *region-code*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.us-east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.us-east-1.amazonaws.com/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
                    "oidc.eks.us-east-1.amazonaws.com/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-system:aws-
node"
                }
            }
        }
    ]
}
```

iii. Erstellen Sie die Rolle. Sie können *AmazonEKSVPCCNIRole* mit einem beliebigen Namen ersetzen, den Sie wählen.

```
aws iam create-role \
  --role-name AmazonEKSVPCCNIRole \
  --assume-role-policy-document file://"vpc-cni-trust-policy.json"
```

iv. Fügen Sie der Rolle die erforderliche IAM-Richtlinie an. Führen Sie den Befehl aus, der der IP-Familie Ihres Clusters entspricht.

- IPv4

  ```
  aws iam attach-role-policy \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKS_CNI_Policy \
    --role-name AmazonEKSVPCCNIRole
  ```

- IPv6

  Ersetzen Sie *111122223333* durch den Namen Ihres Clusters und *AmazonEKS_CNI_IPv6_Policy* durch Ihre IPv6-Richtlinie. Wenn Sie noch keine IPv6-Richtlinie haben, lesen Sie the section called “Erstellen Sie eine IAM-Richtlinie für Cluster, die die IPv6-Familie verwendet”, um eine zu erstellen. Um IPv6 mit Ihrem Cluster zu verwenden, muss dieser mehrere Anforderungen erfüllen. Weitere Informationen finden Sie unter the section called “IPv6”.

```
aws iam attach-role-policy \
  --policy-arn arn:aws: iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy \
  --role-name AmazonEKSVPCCNIRole
```

v. Führen Sie den folgenden Befehl aus und fügen Sie dem `aws-node`-Servicekonto den ARN der IAM-Rolle hinzu, die Sie zuvor erstellt haben. Ersetzen Sie die Beispielwerte durch eigene Werte.

```
kubectl annotate serviceaccount \
    -n kube-system aws-node \
    eks.amazonaws.com/role-arn=arn:aws: iam::111122223333:role/
AmazonEKSVPCCNIRole
```

3. (Optional) Konfigurieren Sie den AWS Security Token Service-Endpunkttyp, der von Ihrem Kubernetes-Dienstkonto verwendet wird. Weitere Informationen finden Sie unter the section called “STS-Endpunkte”.

Schritt 2: Amazon-VPC-CNI-Plugin für Kubernetes-Pods erneut bereitstellen

1. Löschen Sie alle vorhandenen Pods, die dem Servicekonto zugeordnet sind, und erstellen Sie sie neu, um die Umgebungsvariablen für Anmeldeinformationen anzuwenden. Die Annotation wird nicht auf Pods angewendet, die derzeit ohne die Annotation ausgeführt werden. Mit dem folgenden Befehl werden die vorhandenen `aws-node` DaemonSet Pods gelöscht und mit der Anmerkung zum Dienstkonto bereitgestellt.

```
kubectl delete Pods -n kube-system -l k8s-app=aws-node
```

2. Bestätigen Sie, dass alle Pods neu gestartet wurden.

```
kubectl get pods -n kube-system -l k8s-app=aws-node
```

3. Beschreiben Sie einen der Pods und überprüfen Sie, ob die Umgebungsvariablen `AWS_WEB_IDENTITY_TOKEN_FILE` und `AWS_ROLE_ARN` vorhanden sind. *`cpjw7`*Ersetzen Sie ihn durch den Namen eines Ihrer Pods, der in der Ausgabe des vorherigen Schritts zurückgegeben wurde.

```
kubectl describe pod -n kube-system aws-node-cpjw7 | grep 'AWS_ROLE_ARN:\|
AWS_WEB_IDENTITY_TOKEN_FILE:'
```

Eine Beispielausgabe sieht wie folgt aus.

```
AWS_ROLE_ARN: arn:aws: iam::111122223333:role/AmazonEKSVPCCNIRole
      AWS_WEB_IDENTITY_TOKEN_FILE:  /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
      AWS_ROLE_ARN: arn:aws: iam::111122223333:role/AmazonEKSVPCCNIRole
      AWS_WEB_IDENTITY_TOKEN_FILE:            /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
```

Zwei Sätze doppelter Ergebnisse werden zurückgegeben, da der Pod zwei Container enthält. Beide Container haben die gleichen Werte.

Wenn Ihr Pod den AWS regionalen Endpunkt verwendet, wird die folgende Zeile auch in der vorherigen Ausgabe zurückgegeben.

```
AWS_STS_REGIONAL_ENDPOINTS=regional
```

Schritt 3: Entfernen Sie die CNI-Richtlinie aus der Knoten-IAM-Rolle

Wenn Ihrer Amazon EKS-Knoten-IAM-Rolle derzeit die `AmazonEKS_CNI_Policy` IAM (`IPv4`) -Richtlinie oder eine IPv6 Richtlinie angehängt ist und Sie eine separate IAM-Rolle erstellt, ihr stattdessen die Richtlinie angehängt und sie dem `aws-node` Kubernetes-Servicekonto zugewiesen haben, empfehlen wir Ihnen, die Richtlinie mit dem AWS CLI-Befehl, der der IP-Familie Ihres Clusters entspricht, aus Ihrer Knotenrolle zu entfernen. Ersetzen Sie *`AmazonEKSNodeRole`* durch den Namen Ihrer Knotenrolle.

- `IPv4`

```
aws iam detach-role-policy --role-name AmazonEKSNodeRole --policy-arn arn:aws:
 iam::aws:policy/AmazonEKS_CNI_Policy
```

- `IPv6`

  Ersetzen Sie *`111122223333`* durch den Namen Ihres Clusters und *`AmazonEKS_CNI_IPv6_Policy`* durch Ihre `IPv6`-Richtlinie.

```
aws iam detach-role-policy --role-name AmazonEKSNodeRole --policy-arn arn:aws:
 iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy
```

Erstellen Sie eine IAM-Richtlinie für Cluster, die die **`IPv6`**-Familie verwendet

Wenn Sie einen Cluster erstellt haben, der die `IPv6`-Familie verwendet, und für den Cluster die Version `1.10.1` oder höher des Amazon-VPC-CNI-Plugins für das Kubernetes-Add-On konfiguriert ist, müssen Sie eine IAM-Richtlinie erstellen, die Sie einer IAM-Rolle zuweisen können. Wenn Sie einen vorhandenen Cluster haben, den Sie bei der Erstellung nicht mit der `IPv6`-Familie konfiguriert haben, müssen Sie zur Verwendung von `IPv6` einen neuen Cluster erstellen. Weitere Informationen zur Verwendung von `IPv6` mit Ihrem Cluster finden Sie unter the section called “IPv6”.

1. Kopieren Sie den folgenden Text und speichern Sie ihn in einer Datei mit dem Namen `vpc-cni-ipv6-policy.json`.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AssignIpv6Addresses",
                "ec2:DescribeInstances",
                "ec2:DescribeTags",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeInstanceTypes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ]
        }
    ]
}
```

2. Erstellen Sie die IAM-Richtlinie.

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-document
 file://vpc-cni-ipv6-policy.json
```

Weitere Informationen zu VPC-CNI-Modi und deren Konfiguration

Das Amazon-VPC-CNI-Plugin für Kubernetes bietet Netzwerke für Pods. Verwenden Sie die folgende Tabelle, um mehr über die verfügbaren Netzwerk-Feature zu erfahren.

Netzwerk-Feature	Weitere Informationen
Konfiguration Ihres Clusters zur Zuweisung von IPv6-Adressen an Cluster, Pods und Services	the section called “IPv6”
Verwendung der IPv4-Quellnetzwerk-Adressübersetzung für Pods	the section called “Ausgehenden Datenverkehr”
Einschränkung des Netzwerk-Datenverkehrs zu und von Ihren Pods	the section called “Datenverkehr einschränken”
Anpassung der sekundären Netzwerkschnittstelle in Knoten	the section called “Benutzerdefinierte Netzwerke”
Erhöhung der IP-Adressen für Ihren Knoten	the section called “IP-Adressen erhöhen”
Verwendung von Sicherheitsgruppen für den Netzwerkverkehr von Pods	the section called “Sicherheitsgruppen für Pods”
Verwendung mehrerer Netzwerkschnittstellen für Pods	the section called “Mehrere Schnittstellen”

Erfahren Sie mehr über IPv6 Adressen für Cluster, Pods und Dienste

Gilt für: Pods mit EC2 Amazon-Instances und Fargate Pods

Standardmäßig weist Kubernetes Ihren Pods und Services `IPv4`-Adressen zu. Anstatt Ihren Pods und Services `IPv4`-Adressen zuzuweisen, können Sie Ihren Cluster so konfigurieren, dass er ihnen `IPv6`-Adressen zuweist. Amazon EKS unterstützt keine Dual-Stack-Pods oder -Services, obwohl Kubernetes dies tut. Daher können Sie Ihren Pods und Services nicht sowohl `IPv4`- als auch `IPv6`-Adressen zuweisen.

Sie wählen aus, welche IP-Familie Sie für Ihren Cluster verwenden möchten, wenn Sie ihn erstellen. Sie können die Familie nach dem Erstellen des Clusters nicht mehr ändern.

Ein Tutorial zum Bereitstellen eines Amazon-EKS-IPv6-Clusters finden Sie unter the section called “Bereitstellen”.

Nachfolgend finden Sie einige Überlegungen zur Verwendung des Features:

IPv6-Feature-Support

- Kein Windows-Support: Windows-Pods und -Services werden nicht unterstützt.
- Nitro-basierte EC2 Knoten erforderlich: Sie können sie nur IPv6 mit AWS Nitro-basierten Amazon EC2 - oder Fargate-Knoten verwenden.
- EC2 und unterstützte Fargate-Knoten: Sie können sie IPv6 the section called “Sicherheitsgruppen für Pods” mit EC2 Amazon-Knoten und Fargate-Knoten verwenden.
- Outposts werden nicht unterstützt: Sie können IPv6 nicht mit Amazon EKS in AWS Outposts verwenden.
- FSx for Lustre wird nicht unterstützt: Das the section called “Amazon FSx für Lustre” wird nicht unterstützt.
- Benutzerdefinierte Netzwerke werden nicht unterstützt: Wenn Sie zuvor the section called “Benutzerdefinierte Netzwerke” verwendet haben, um die Erschöpfung von IP-Adressen zu verringern, können Sie stattdessen IPv6 verwenden. Sie können benutzerdefinierte Netzwerke nicht mit IPv6 verwenden. Wenn Sie benutzerdefinierte Netzwerke für die Netzwerkisolierung verwenden, müssen Sie möglicherweise weiterhin benutzerdefinierte Netzwerke und die IPv4-Produktfamilie für Ihre Cluster verwenden.

Zuweisung von IP-Adressen

- Kubernetes- Services: Kubernetes-Services werden lediglich IPv6-Adressen zugewiesen. Ihnen sind keine IPv4 Adressen zugewiesen.
- Pods: Pods werden eine IPv6 Adresse und eine lokale IPv4 Hostadresse zugewiesen. Die hostlokale IPv4 Adresse wird mithilfe eines hostlokalen CNI-Plug-ins zugewiesen, das mit VPC CNI verkettet ist, und die Adresse wird nicht an die Kubernetes-Steuerebene gemeldet. Es wird nur verwendet, wenn ein Pod mit externen IPv4 Ressourcen in einer anderen Amazon VPC oder im Internet kommunizieren muss. Die hostlokale IPv4 Adresse wird SNATed (per VPC CNI) an die primäre IPv4 Adresse der primären ENI des Worker-Knotens weitergeleitet.
- Pods und Services: Pods und Services erhalten nur IPv6-Adressen, keine IPv4-Adressen. Wenn Pods mit externen IPv4-Endpunkten kommunizieren müssen, verwenden sie NAT auf dem Knoten selbst. Diese integrierte NAT-Fähigkeit macht und überflüssig. DNS64 NAT64 Für Datenverkehr,

der einen öffentlichen Internetzugang erfordert, wird die Quellnetzwerkadresse des Pods in eine öffentliche IP-Adresse übersetzt.

- Weiterleitung von Adressen: Wenn ein Pod außerhalb der VPC kommuniziert, bleibt seine ursprüngliche IPv6-Adresse erhalten (sie wird nicht in die IPv6-Adresse des Knotens übersetzt). Dieser Datenverkehr wird direkt über ein Internet-Gateway oder ein nur für den Ausgang bestimmtes Internet-Gateway weitergeleitet.
- Knoten: Allen Knoten wird eine IPv4- und IPv6-Adresse zugewiesen.
- Fargate-Pods: Jeder Fargate-Pod erhält eine IPv6-Adresse vom CIDR, die für das Subnetz angegeben ist, in dem er bereitgestellt wird. Die zugrunde liegende Hardwareeinheit, auf der Fargate Pods ausgeführt werden, erhält eine eindeutige IPv4 IPv6 Adresse von den CIDRs , die dem Subnetz zugewiesen sind, in dem die Hardwareeinheit bereitgestellt wird.

Vorgehensweise zur Verwendung von **IPv6** mit EKS

- Neuen Cluster erstellen: Sie müssen einen neuen Cluster erstellen und angeben, dass Sie die IPv6-Familie für diesen Cluster verwenden möchten. Sie können die IPv6-Familie nicht für einen Cluster aktivieren, den Sie von einer früheren Version aktualisiert haben. Anweisungen zum Erstellen eines neuen Clusters finden Sie unter Überlegungen.
- Aktuelles VPC CNI verwenden: Stellen Sie Amazon VPC CNI Version 1.10.1 oder höher bereit. Diese Version oder höher wird standardmäßig bereitgestellt. Nachdem Sie das Add-On bereitgestellt haben, können Sie Ihr Amazon-VPC-CNI-Add-On nicht auf eine niedrigere Version als 1.10.1 herabstufen, ohne zuerst alle Knoten in allen Knotengruppen in Ihrem Cluster zu entfernen.
- VPC CNI konfigurieren für **IPv6**: Wenn Sie EC2 Amazon-Knoten verwenden, müssen Sie das Amazon VPC CNI-Add-on mit IP-Präfix-Delegierung und konfigurieren. IPv6 Wenn Sie beim Erstellen Ihres Clusters die IPv6-Familie wählen, wird die Version 1.10.1 des Add-ons standardmäßig auf diese Konfiguration festgelegt. Dies ist sowohl bei einem selbstverwalteten als auch bei Amazon EKS Add-on der Fall. Weitere Informationen über die IP-Präfix-Delegation finden Sie unter the section called “IP-Adressen erhöhen”.
- **IPv4**- und **IPv6**-Adressen konfigurieren: Wenn Sie einen Cluster erstellen, müssen die VPC und die von Ihnen angegebenen Subnetze über einen IPv6-CIDR-Block verfügen, der der VPC und den von Ihnen angegebenen Subnetzen zugewiesen ist. Sie müssen auch über einen IPv4-CIDR-Block verfügen. Dies liegt daran, dass eine VPC, selbst wenn Sie nur IPv6 verwenden möchten, immer noch einen IPv4-CIDR-Block benötigt, um zu funktionieren. Weitere Informationen finden Sie unter Einen IPv6 CIDR-Block mit Ihrer VPC verknüpfen im Amazon VPC-Benutzerhandbuch.

- Automatische Zuweisung von IPv6 Adressen zu Knoten: Wenn Sie Ihre Knoten erstellen, müssen Sie Subnetze angeben, die für die automatische IPv6 Zuweisung von Adressen konfiguriert sind. Andernfalls können Sie Ihre Knoten nicht bereitstellen. Diese Konfiguration ist standardmäßig deaktiviert. Weitere Informationen finden Sie unter Ändern des IPv6 Adressierungsattributs für Ihr Subnetz im Amazon VPC-Benutzerhandbuch.
- Routing-Tabellen für die Verwendung von **IPv6** festlegen: Die Routing-Tabellen, die Ihren Subnetzen zugewiesen sind, müssen Routen für IPv6-Adressen haben. Weitere Informationen finden Sie unter Migrieren zu IPv6 im Amazon VPC-Benutzerhandbuch.
- Sicherheitsgruppen für **IPv6** festlegen: Ihre Sicherheitsgruppen müssen IPv6-Adressen zulassen. Weitere Informationen finden Sie unter Migrieren zu IPv6 im Amazon VPC-Benutzerhandbuch.
- Load Balancer einrichten: Verwenden Sie Version 2.3.1 oder höher des Load Balancer Controllers, um den AWS Lastenausgleich von HTTP-Anwendungen mithilfe des the section called “Application Load Balancing” oder des Netzwerkverkehrs mithilfe der the section called “Netzwerk-Load-Balancer” To IPv6 Pods durchzuführen, wobei sich einer der Load Balancer im IP-Modus, aber nicht im Instanzmodus befindet. Weitere Informationen finden Sie unter the section called “ AWS Load Balancer Balancer-Controller”.
- **IPv6**-IAM-Richtlinie hinzufügen: Sie müssen eine IPv6-IAM-Richtlinie an die IAM- oder CNI-IAM-Rolle Ihres Knotens anfügen. Zwischen den beiden empfehlen wir, dass Sie sie an eine CNI IAM-Rolle anfügen. Weitere Informationen erhalten Sie unter the section called “Erstellen Sie eine IAM-Richtlinie für Cluster, die die IPv6-Familie verwendet” und the section called “Schritt 1: Amazon-VPC-CNI-Plugin für die Kubernetes-IAM-Rolle erstellen”.
- Evaluieren Sie alle Komponenten: Führen Sie vor der Bereitstellung von IPv6 Clustern eine gründliche Bewertung Ihrer Anwendungen, Amazon EKS-Add-ons und AWS Services durch, die Sie integrieren. Dies soll sicherstellen, dass mit IPv6 alles wie erwartet funktioniert.

Bereitstellen eines **IPv6**-Clusters von Amazon EKS und verwalteter Amazon-Linux-Knoten

In diesem Tutorial stellen Sie eine IPv6 Amazon VPC, einen Amazon EKS-Cluster mit der IPv6 Familie und eine verwaltete Knotengruppe mit Amazon EC2 Amazon Linux-Knoten bereit. Sie können keine Amazon EC2 Windows-Knoten in einem IPv6 Cluster bereitstellen. Sie können Fargate-Knoten auch in Ihrem Cluster bereitstellen, obwohl diese Anweisungen in diesem Thema der Einfachheit halber nicht enthalten sind.

Voraussetzungen

Führen Sie die folgenden Schritte aus, bevor Sie mit dem Tutorial beginnen:

Installieren und konfigurieren Sie die folgenden Tools und Ressourcen, die Sie zum Erstellen und Verwalten eines Amazon-EKS-Clusters benötigen.

- Wir empfehlen Ihnen, sich mit allen Einstellungen vertraut zu machen und einen Cluster mit den Einstellungen bereitzustellen, die Ihren Anforderungen entsprechen. Weitere Informationen finden Sie unter the section called “Erstellen eines Clusters”, the section called “Verwaltete Knotengruppen” und in den Überlegungen zu diesem Thema. Sie können nur einige Einstellungen aktivieren, wenn Sie Ihren Cluster erstellen.
- Das `kubectl`-Befehlszeilen-Tool ist auf Ihrem Gerät oder in der AWS CloudShell installiert. Die Version kann mit der Kubernetes-Version Ihres Clusters identisch sein oder bis zu einer Nebenversion älter oder neuer sein. Wenn Ihre Clusterversion beispielsweise `1.29` ist, können Sie `kubectl`-Version `1.28`, `1.29`, oder `1.30` damit verwenden. Informationen zum Installieren oder Aktualisieren von `kubectl` finden Sie unter the section called “kubectl und eksctl einrichten”.
- Der von Ihnen verwendete IAM-Sicherheitsprinzipal muss über Berechtigungen für die Arbeit mit Amazon EKS-IAM-Rollen, serviceverknüpften Rollen AWS CloudFormation, einer VPC und verwandten Ressourcen verfügen. Weitere Informationen finden Sie unter Aktionen und Verwendung von serviceverknüpften Rollen im IAM-Benutzerhandbuch.
- Wenn Sie eksctl verwenden, installieren Sie Version `0.215.0` oder höher auf Ihrem Computer. Informationen zum Installieren oder Aktualisieren finden Sie in der Dokumentation zu `eksctl` unter Installation.
- Version `2.12.3` oder höher oder Version `1.27.160` oder höher der auf Ihrem Gerät installierten und konfigurierten AWS Befehlszeilenschnittstelle (AWS CLI) oder AWS CloudShell. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Paketmanager wie `yum`, `apt-get`, oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der AWS CLI. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Die AWS CLI-Version, in der installiert ist, AWS CloudShell kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zur Aktualisierung finden Sie im AWS CloudShell Benutzerhandbuch unter AWS CLI in Ihrem Home-Verzeichnis installieren. Wenn Sie die verwenden AWS CloudShell, müssen Sie möglicherweise Version 2.12.3 oder höher oder 1.27.160 oder höher der AWS CLI installieren, da es sich bei der in der installierten AWS Standard-CLI-Version AWS CloudShell möglicherweise um eine frühere Version handelt.

Sie können eksctl oder die CLI verwenden, um einen `IPv6`-Cluster bereitzustellen.

Stellen Sie einen Cluster mit eksctl bereit IPv6

a. Erstellen Sie die Datei `ipv6-cluster.yaml`. Kopieren Sie den folgenden Befehl auf Ihr Gerät. Nehmen Sie nach Bedarf die folgenden Änderungen am Befehl vor und führen Sie anschließend den geänderten Befehl aus:

- Ersetzen Sie *`my-cluster`* durch Ihren Cluster-Namen. Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein.
- *`region-code`*Ersetzen Sie durch eine beliebige AWS Region, die von Amazon EKS unterstützt wird. Eine Liste der AWS Regionen finden Sie unter Amazon EKS-Endpunkte und Kontingente im AWS Allgemeinen Referenzhandbuch.
- Der Wert für `version` mit der Version Ihres Clusters. Weitere Informationen finden Sie unter Unterstützte Versionen von Amazon EKS.
- Ersetzen Sie *`my-nodegroup`* durch einen Namen für Ihre Knotengruppe. Der Name der Knotengruppe darf nicht länger als 63 Zeichen sein. Er muss mit einem Buchstaben oder einer Ziffer beginnen, kann danach aber auch Bindestriche und Unterstriche enthalten.
- Ersetzen Sie *`t3.medium`* durch jeden beliebigen Instance-Typ von AWS Nitro System.

```
cat >ipv6-cluster.yaml <<EOF
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code
  version: "X.XX"

kubernetesNetworkConfig:
  ipFamily: IPv6

addons:
  - name: vpc-cni
    version: latest
  - name: coredns
    version: latest
  - name: kube-proxy
```

```
    version: latest

iam:
  withOIDC: true

managedNodeGroups:
  - name: my-nodegroup
    instanceType: t3.medium
EOF
```

b. Erstellen Sie Ihren Cluster.

```
eksctl create cluster -f ipv6-cluster.yaml
```

Die Clustererstellung dauert mehrere Minuten. Fahren Sie nicht fort, bis Sie die letzte Ausgabezeile sehen, die der folgenden Ausgabe ähnelt.

```
[...]
[#]  EKS cluster "my-cluster" in "region-code" region is ready
```

c. Bestätigen Sie, dass den Standard-Pods IPv6-Adressen zugewiesen sind.

```
kubectl get pods -n kube-system -o wide
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME                       READY   STATUS    RESTARTS   AGE     IP
                    NODE                                              NOMINATED NODE
 READINESS GATES
aws-node-rslts             1/1     Running   1          5m36s
 2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
aws-node-t74jh             1/1     Running   0          5m32s
 2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-cw7w2   1/1     Running   0          56m
 2600:1f13:b66:8203:34e5::                ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-tx6n8   1/1     Running   0          56m
 2600:1f13:b66:8203:34e5::1               ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
```

```
kube-proxy-btpbk           1/1     Running   0          5m36s
 2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
kube-proxy-jjk2g           1/1     Running   0          5m33s
 2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
```

d. Vergewissern Sie sich, dass Standard-Services IPv6-Adressen zugewiesen wurden.

```
kubectl get services -n kube-system -o wide
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME       TYPE        CLUSTER-IP          EXTERNAL-IP   PORT(S)         AGE
 SELECTOR
kube-dns   ClusterIP   fd30:3087:b6c2::a   <none>        53/UDP,53/TCP   57m   k8s-
app=kube-dns
```

e. (Optional) Stellen Sie eine Beispielanwendung bereit oder stellen Sie den AWS Load Balancer Controller und eine Beispielanwendung bereit, um eine Lastenverteilung für HTTP-Anwendungen mit the section called “Application Load Balancing” oder Netzwerkverkehr mit the section called “Netzwerk-Load-Balancer” in IPv6-Pods durchzuführen.

f. Nachdem Sie mit dem Cluster und den Knoten fertig sind, die Sie für dieses Tutorial erstellt haben, sollten Sie die Ressourcen, die Sie erstellt haben, mit dem folgenden Befehl bereinigen.

```
eksctl delete cluster my-cluster
```

Bereitstellen eines IPv6 Clusters mit AWS CLI

Important

- Sie müssen alle Schritte in diesem Verfahren als derselbe Benutzer ausführen. Führen Sie den folgenden Befehl aus, um den aktuellen Benutzer zu überprüfen:

  ```
  aws sts get-caller-identity
  ```

- Sie müssen alle Schritte in diesem Verfahren in derselben Shell ausführen. In mehreren Schritten in den vorherigen Schritten werden festgelegte Variablen verwendet. Schritte, die Variablen verwenden, funktionieren nicht ordnungsgemäß, wenn die Variablenwerte

in einer anderen Shell festgelegt sind. Wenn Sie das AWS CloudShellfolgende Verfahren verwenden, denken Sie daran, dass Ihre Shell-Sitzung beendet wird, wenn Sie etwa 20 bis 30 Minuten lang nicht mit der Tastatur oder dem Zeiger damit interagieren. Laufende Prozesse zählen nicht als Interaktionen.

- Die Anweisungen sind für die Bash-Shell geschrieben und müssen möglicherweise in anderen Shells angepasst werden.

Ersetzen Sie alle Beispielwerte in den Schritten dieses Verfahrens mit eigenen Werten.

a. Führen Sie die folgenden Befehle aus, um einige in späteren Schritten verwendete Variablen festzulegen. *region-code*Ersetzen Sie es durch die AWS Region, in der Sie Ihre Ressourcen bereitstellen möchten. Der Wert kann eine beliebige AWS Region sein, die von Amazon EKS unterstützt wird. Eine Liste der AWS Regionen finden Sie unter Amazon EKS-Endpunkte und Kontingente im AWS Allgemeinen Referenzhandbuch. Ersetzen Sie *my-cluster* durch Ihren Cluster-Namen. Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein. Ersetzen Sie *my-nodegroup* durch einen Namen für Ihre Knotengruppe. Der Name der Knotengruppe darf nicht länger als 63 Zeichen sein. Er muss mit einem Buchstaben oder einer Ziffer beginnen, kann danach aber auch Bindestriche und Unterstriche enthalten. Ersetzen Sie *111122223333* durch Ihre Konto-ID.

```
export region_code=region-code
export cluster_name=my-cluster
export nodegroup_name=my-nodegroup
export account_id=111122223333
```

b. Erstellen Sie eine Amazon VPC mit öffentlichen und privaten Subnetzen, die die Anforderungen von Amazon EKS und IPv6 erfüllt.

   i. Führen Sie den folgenden Befehl aus, um eine Variable für Ihren AWS CloudFormation Stack-Namen festzulegen. Sie können *my-eks-ipv6-vpc* mit einem beliebigen Namen ersetzen, den Sie wählen.

   ```
   export vpc_stack_name=my-eks-ipv6-vpc
   ```

   ii. Erstellen Sie eine IPv6 VPC mithilfe einer AWS CloudFormation Vorlage.

```
aws cloudformation create-stack --region $region_code --stack-name $vpc_stack_name
 \
  --template-url https://s3.us-west-2.amazonaws.com/amazon-eks/
cloudformation/2020-10-29/amazon-eks-ipv6-vpc-public-private-subnets.yaml
```

Das Erstellen des Stacks nimmt einige Minuten in Anspruch. Führen Sie den folgenden Befehl aus. Fahren Sie erst mit dem nächsten Schritt fort, wenn die Ausgabe des Befehls `CREATE_COMPLETE` ist.

```
aws cloudformation describe-stacks --region $region_code --stack-name
 $vpc_stack_name --query Stacks[].StackStatus --output text
```

iii. Rufen Sie IDs die öffentlichen Subnetze ab, die erstellt wurden.

```
aws cloudformation describe-stacks --region $region_code --stack-name
 $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SubnetsPublic`].OutputValue' --output
 text
```

Eine Beispielausgabe sieht wie folgt aus.

```
subnet-0a1a56c486EXAMPLE,subnet-099e6ca77aEXAMPLE
```

iv. Aktivieren Sie die Option „`IPv6-Adressen automatisch zuweisen“ für die erstellten öffentlichen Subnetze.

```
aws ec2 modify-subnet-attribute --region $region_code --subnet-id
 subnet-0a1a56c486EXAMPLE --assign-ipv6-address-on-creation
aws ec2 modify-subnet-attribute --region $region_code --subnet-id
 subnet-099e6ca77aEXAMPLE --assign-ipv6-address-on-creation
```

v. Rufen Sie die Namen der Subnetze und Sicherheitsgruppen, die mit der Vorlage erstellt wurden, aus dem bereitgestellten AWS CloudFormation Stack ab und speichern Sie sie in Variablen, um sie in einem späteren Schritt zu verwenden.

```
security_groups=$(aws cloudformation describe-stacks --region $region_code --
stack-name $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SecurityGroups`].OutputValue' --output
 text)
```

```
public_subnets=$(aws cloudformation describe-stacks --region $region_code --stack-
name $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SubnetsPublic`].OutputValue' --output
 text)

private_subnets=$(aws cloudformation describe-stacks --region $region_code --
stack-name $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SubnetsPrivate`].OutputValue' --output
 text)

subnets=${public_subnets},${private_subnets}
```

c. Erstellen Sie eine Cluster-IAM-Rolle und fügen Sie ihr die erforderliche von Amazon EKS IAM verwaltete Richtlinie hinzu. Von Amazon EKS verwaltete Kubernetes-Cluster rufen in Ihrem Namen andere AWS Services auf, um die Ressourcen zu verwalten, die Sie mit dem Service verwenden.

   i. Führen Sie den folgenden Befehl aus, um die Datei `eks-cluster-role-trust-policy.json` zu erstellen.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

   ii. Führen Sie den folgenden Befehl aus, um eine Variable für Ihren Rollennamen festzulegen. Sie können *myAmazonEKSClusterRole* mit einem beliebigen Namen ersetzen, den Sie wählen.

```
export cluster_role_name=myAmazonEKSClusterRole
```

   iii. Erstellen Sie die Rolle.

```
aws iam create-role --role-name $cluster_role_name --assume-role-policy-document
 file://"eks-cluster-role-trust-policy.json"
```

iv. Rufen Sie den ARN der IAM-Rolle ab und speichern Sie ihn für einen späteren Schritt in einer Variablen.

```
CLUSTER_IAM_ROLE=$(aws iam get-role --role-name $cluster_role_name --
query="Role.Arn" --output text)
```

v. Hängen Sie die erforderliche von Amazon EKS verwaltete IAM-Richtlinie an die Rolle an.

```
aws iam attach-role-policy --policy-arn arn:aws: iam::aws:policy/
AmazonEKSClusterPolicy --role-name $cluster_role_name
```

d. Erstellen Sie Ihren Cluster.

```
aws eks create-cluster --region $region_code --name $cluster_name --kubernetes-
version 1.XX \
   --role-arn $CLUSTER_IAM_ROLE --resources-vpc-config subnetIds=
$subnets,securityGroupIds=$security_groups \
   --kubernetes-network-config ipFamily=ipv6
```

i. HINWEIS: Sie erhalten möglicherweise eine Fehlermeldung, dass eine der Availability Zones in Ihrer Anfrage nicht über genügend Kapazität zum Erstellen eines Amazon-EKS-Clusters verfügt. Wenn dies der Fall ist, enthält die Fehlerausgabe die Availability Zones, die einen neuen Cluster unterstützen können. Versuchen Sie, Ihren Cluster mit mindestens zwei Subnetzen erneut zu erstellen, die sich in den unterstützten Availability Zones für Ihr Konto befinden. Weitere Informationen finden Sie unter the section called “Unzureichende Kapazität”.

   Die Erstellung des Clusters dauert mehrere Minuten. Führen Sie den folgenden Befehl aus. Fahren Sie erst mit dem nächsten Schritt fort, wenn die Ausgabe aus dem Befehl `ACTIVE` ist.

```
aws eks describe-cluster --region $region_code --name $cluster_name --query
 cluster.status
```

e. Erstellen oder aktualisieren Sie eine `kubeconfig`-Datei für Ihren Cluster, sodass Sie mit Ihrem Cluster kommunizieren können.

```
aws eks update-kubeconfig --region $region_code --name $cluster_name
```

Standardmäßig wird die `config`-Datei in `~/.kube` erstellt oder die Konfiguration des neuen Clusters wird einer vorhandenen `config`-Datei in `~/.kube` hinzugefügt.

f. Erstellen Sie eine Knoten-IAM-Rolle.

i. Führen Sie den folgenden Befehl aus, um die Datei `vpc-cni-ipv6-policy.json` zu erstellen.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AssignIpv6Addresses",
                "ec2:DescribeInstances",
                "ec2:DescribeTags",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeInstanceTypes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ]
        }
    ]
}
```

ii. Erstellen Sie die IAM-Richtlinie.

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-document
 file://vpc-cni-ipv6-policy.json
```

iii. Führen Sie den folgenden Befehl aus, um die Datei `node-role-trust-relationship.json` zu erstellen.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

iv. Führen Sie den folgenden Befehl aus, um eine Variable für Ihren Rollennamen festzulegen. Sie können *AmazonEKSNodeRole* mit einem beliebigen Namen ersetzen, den Sie wählen.

```
export node_role_name=AmazonEKSNodeRole
```

v. Erstellen Sie die IAM-Rolle.

```
aws iam create-role --role-name $node_role_name --assume-role-policy-document
 file://"node-role-trust-relationship.json"
```

vi. Fügen Sie die IAM-Richtlinie an die IAM-Rolle an.

```
aws iam attach-role-policy --policy-arn arn:aws: iam::$account_id:policy/
AmazonEKS_CNI_IPv6_Policy \
    --role-name $node_role_name
```

> **Important**
>
> Der Einfachheit halber wird in diesem Tutorial die Richtlinie an diese IAM-Rolle angehängt. In einem Produktions-Cluster empfehlen wir jedoch, die Richtlinie an eine separate IAM-Rolle anzuhängen. Weitere Informationen finden Sie unter the section called “Konfiguration für IRSA”.

vii. Hängen Sie die beiden erforderlichen IAM-verwalteten Richtlinien an die IAM-Rolle an.

```
aws iam attach-role-policy --policy-arn arn:aws: iam::aws:policy/
AmazonEKSWorkerNodePolicy \
  --role-name $node_role_name
aws iam attach-role-policy --policy-arn arn:aws: iam::aws:policy/
AmazonEC2ContainerRegistryReadOnly \
  --role-name $node_role_name
```

viii. Rufen Sie den ARN der IAM-Rolle ab und speichern Sie ihn für einen späteren Schritt in einer Variablen.

```
node_iam_role=$(aws iam get-role --role-name $node_role_name --query="Role.Arn" --
output text)
```

g. Erstellen Sie eine verwaltete Knotengruppe.

i. Sehen Sie sich IDs die Subnetze an, die Sie in einem vorherigen Schritt erstellt haben.

```
echo $subnets
```

Eine Beispielausgabe sieht wie folgt aus.

```
subnet-0a1a56c486EXAMPLE,subnet-099e6ca77aEXAMPLE,subnet-0377963d69EXAMPLE,subnet-0c05f819
```

ii. Erstellen Sie die Knotengruppe. Ersetzen Sie *0a1a56c486EXAMPLE*, *099e6ca77aEXAMPLE*, *0377963d69EXAMPLE* und *0c05f819d5EXAMPLE* mit den im vorherigen Schritt zurückgegebenen Werten. Achten Sie darauf, die Kommas zwischen den Subnetzen IDs aus der vorherigen Ausgabe im folgenden Befehl zu entfernen. Sie können *t3.medium* mit jedem beliebigen AWS -Nitro-System-Instance-Typ ersetzen.

```
aws eks create-nodegroup --region $region_code --cluster-name $cluster_name --
nodegroup-name $nodegroup_name \
    --subnets subnet-0a1a56c486EXAMPLE subnet-099e6ca77aEXAMPLE
 subnet-0377963d69EXAMPLE subnet-0c05f819d5EXAMPLE \
    --instance-types t3.medium --node-role $node_iam_role
```

Das Erstellen der Knotengruppe nimmt einige Minuten in Anspruch. Führen Sie den folgenden Befehl aus. Fahren Sie nicht mit dem nächsten Schritt fort, bis die zurückgegebene Ausgabe `ACTIVE` ist.

```
aws eks describe-nodegroup --region $region_code --cluster-name $cluster_name --
nodegroup-name $nodegroup_name \
    --query nodegroup.status --output text
```

h. Vergewissern Sie sich, dass den Standard-Pods in der `IP`-Spalte `IPv6`-Adressen zugewiesen wurden.

```
kubectl get pods -n kube-system -o wide
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME                       READY   STATUS    RESTARTS   AGE     IP
                    NODE                                              NOMINATED NODE
 READINESS GATES
aws-node-rslts             1/1     Running   1          5m36s
 2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
aws-node-t74jh             1/1     Running   0          5m32s
 2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-cw7w2   1/1     Running   0          56m
 2600:1f13:b66:8203:34e5::                ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-tx6n8   1/1     Running   0          56m
 2600:1f13:b66:8203:34e5::1               ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
kube-proxy-btpbk           1/1     Running   0          5m36s
 2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
kube-proxy-jjk2g           1/1     Running   0          5m33s
 2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
```

i. Vergewissern Sie sich, dass den Standard-Services in der `IP`-Spalte `IPv6`-Adressen zugewiesen wurden.

```
kubectl get services -n kube-system -o wide
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME       TYPE        CLUSTER-IP          EXTERNAL-IP   PORT(S)         AGE
 SELECTOR
kube-dns   ClusterIP   fd30:3087:b6c2::a   <none>        53/UDP,53/TCP   57m   k8s-
app=kube-dns
```

j. (Optional) Stellen Sie eine Beispielanwendung bereit oder stellen Sie den AWS Load Balancer Controller und eine Beispielanwendung bereit, um eine Lastenverteilung für HTTP-Anwendungen

mit the section called “Application Load Balancing” oder Netzwerkverkehr mit the section called “Netzwerk-Load-Balancer” in IPv6-Pods durchzuführen.

k. Nachdem Sie mit dem Cluster und den Knoten fertig sind, die Sie für dieses Tutorial erstellt haben, sollten Sie die Ressourcen, die Sie erstellt haben, mit den folgenden Befehlen bereinigen. Stellen Sie sicher, dass Sie keine der Ressourcen außerhalb dieses Tutorials verwenden, bevor Sie sie löschen.

   i. Wenn Sie diesen Schritt in einer anderen Shell als die vorherigen Schritte ausführen, legen Sie die Werte aller in den vorherigen Schritten verwendeten Variablen fest und ersetzen Sie die Beispielwerte mit den Werten, die Sie angegeben haben, als Sie die vorherigen Schritte ausgeführt haben. Wenn Sie diesen Schritt in derselben Shell ausführen, in der Sie die vorherigen Schritte ausgeführt haben, fahren Sie mit dem nächsten Schritt fort.

```
export region_code=region-code
export vpc_stack_name=my-eks-ipv6-vpc
export cluster_name=my-cluster
export nodegroup_name=my-nodegroup
export account_id=111122223333
export node_role_name=AmazonEKSNodeRole
export cluster_role_name=myAmazonEKSClusterRole
```

   ii. Löschen Sie Ihre Knotengruppe.

```
aws eks delete-nodegroup --region $region_code --cluster-name $cluster_name --
nodegroup-name $nodegroup_name
```

   Dies dauert in der Regel einige Minuten. Führen Sie den folgenden Befehl aus. Fahren Sie nicht mit dem nächsten Schritt fort, bis die Ausgabe zurückgegeben wurde.

```
aws eks list-nodegroups --region $region_code --cluster-name $cluster_name --query
 nodegroups --output text
```

   iii. Löschen Sie den Cluster.

```
aws eks delete-cluster --region $region_code --name $cluster_name
```

   Das Löschen des Clusters nimmt einige Minuten in Anspruch. Stellen Sie vor dem Fortfahren sicher, dass der Cluster mit dem folgenden Befehl gelöscht wird.

```
aws eks describe-cluster --region $region_code --name $cluster_name
```

Fahren Sie erst mit dem nächsten Schritt fort, wenn Ihre Ausgabe der folgenden Ausgabe ähnelt.

```
An error occurred (ResourceNotFoundException) when calling the DescribeCluster
 operation: No cluster found for name: my-cluster.
```

iv. Löschen Sie die IAM-Ressourcen, die Sie erstellt haben. Ersetzen Sie *AmazonEKS_CNI_IPv6_Policy* mit dem von Ihnen gewählten Namen, wenn Sie einen anderen Namen als den in den vorherigen Schritten verwendeten gewählt haben.

```
aws iam detach-role-policy --role-name $cluster_role_name --policy-arn arn:aws:
 iam::aws:policy/AmazonEKSClusterPolicy
aws iam detach-role-policy --role-name $node_role_name --policy-arn arn:aws:
 iam::aws:policy/AmazonEKSWorkerNodePolicy
aws iam detach-role-policy --role-name $node_role_name --policy-arn arn:aws:
 iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
aws iam detach-role-policy --role-name $node_role_name --policy-arn arn:aws: iam::
$account_id:policy/AmazonEKS_CNI_IPv6_Policy
aws iam delete-policy --policy-arn arn:aws: iam::$account_id:policy/
AmazonEKS_CNI_IPv6_Policy
aws iam delete-role --role-name $cluster_role_name
aws iam delete-role --role-name $node_role_name
```

v. Löschen Sie den AWS CloudFormation Stack, der die VPC erstellt hat.

```
aws cloudformation delete-stack --region $region_code --stack-name $vpc_stack_name
```

Aktivierung des ausgehenden Internetzugangs für Pods

Gilt für: Linux-IPv4-Fargate-Knoten, Linux-Knoten mit Amazon-EC2-Instances

Wenn Sie Ihren Cluster mit der IPv6-Familie bereitgestellt haben, treffen die Informationen in diesem Thema nicht auf Ihren Cluster zu, da für IPv6-Adressen keine Netzwerkübersetzung möglich ist. Weitere Informationen zur Verwendung von IPv6 mit Ihrem Cluster finden Sie unter the section called “IPv6”.

Standardmäßig wird jedem Pod in Ihrem Cluster eine private `IPv4`-Adresse aus einem Classless Inter-Domain Routing (CIDR)-Block zugewiesen, der mit der VPC verknüpft ist, in welcher der Pod bereitgestellt wird. Pods im selben VPC kommunizieren miteinander und verwenden diese privaten IP-Adressen als Endpunkte. Wenn ein Pod mit einer `IPv4`-Adresse kommuniziert, die nicht Teil eines mit Ihrer VPC verknüpften CIDR-Blocks ist, übersetzt das Amazon-VPC-CNI-Plugin (für sowohl Linux oder Windows) die `IPv4`-Adresse des standardmäßig in die primäre private `IPv4`-Adresse der primären Elastic-Network-Schnittstelle des Knotens, auf dem der Pod ausgeführt wird *.

> **Note**
>
> Bei Windows-Knoten sind zusätzliche Details zu beachten. Standardmäßig ist das VPC CNI-Plugin für Windows mit einer Netzwerkkonfiguration definiert, in der der Datenverkehr zu einem Ziel innerhalb derselben VPC für SNAT ausgeschlossen ist. Dies bedeutet, dass für die interne VPC-Kommunikation SNAT deaktiviert ist und die einem Pod zugewiesene IP-Adresse innerhalb der VPC routingfähig ist. Beim Datenverkehr zu einem Ziel außerhalb der VPC wird die Quell-Pod-IP jedoch per SNAT an die primäre IP-Adresse der Instance-ENI angepasst. Diese Standardkonfiguration für Windows stellt sicher, dass der Pod auf die gleiche Weise wie die Host-Instance auf Netzwerke außerhalb Ihrer VPC zugreifen kann.

Folgen dieses Verhaltens:

- Ihre Pods können nur dann mit Internetressourcen kommunizieren, wenn dem Knoten, auf dem er ausgeführt wird, eine öffentliche oder elastische IP-Adresse zugeordnet ist und er sich in einem öffentlichen Subnetz befindet. Eine Routing-Tabelle, die einem öffentlichen Subnetz zugeordnet ist, verfügt über eine Route zu einem Internet-Gateway. Wir empfehlen, Knoten nach Möglichkeit in privaten Subnetzen bereitzustellen.
- Bei Versionen des Plugins vor `1.8.0` können Ressourcen in Netzwerken oder VPCs, die über VPC-Peering, eine Transit-VPC oder AWS Direct Connect mit Ihrer Cluster-VPC verbunden sind, keine Kommunikation mit Ihren Pods hinter sekundären elastischen Netzwerkschnittstellen initiieren. Ihre Pods können jedoch eine Kommunikation an diese Ressourcen initiieren und Antworten von ihnen erhalten.

Wenn eine der folgenden Aussagen in Ihrer Umgebung zutrifft, ändern Sie die Standardkonfiguration mit dem folgenden Befehl.

- Sie haben Ressourcen in Netzwerken oder VPCs, die über VPC-Peering, eine Transit-VPC oder AWS Direct Connect mit Ihrer Cluster-VPC verbunden sind und die Kommunikation mit Ihren Pods über eine IPv4-Adresse initiieren müssen, und Ihre Plugin-Version ist älter als `1.8.0`.
- Ihre Pods befinden sich in einem privaten Subnetz und müssen ausgehend mit dem Internet kommunizieren. Das Subnetz hat eine Route zu einem NAT-Gateway.

```
kubectl set env daemonset -n kube-system aws-node AWS_VPC_K8S_CNI_EXTERNALSNAT=true
```

> **Note**
>
> Die `AWS_VPC_K8S_CNI_EXTERNALSNAT`- und `AWS_VPC_K8S_CNI_EXCLUDE_SNAT_CIDRS`-CNI-Konfigurationsvariablen gelten nicht für Windows-Knoten. Die Deaktivierung von SNAT wird für Windows nicht unterstützt. Was den Ausschluss einer Liste von IPv4-CIDRs aus SNAT angeht, können Sie dies definieren, indem Sie im Windows-Bootstrap-Skript den `ExcludedSnatCIDRs`-Parameter angeben. Weitere Informationen zur Verwendung dieses Parameters finden Sie unter the section called “Bootstrap-Skript-Konfigurationsparameter”.

Host-Netzwerk

* Wenn die Spezifikation eines Pods `hostNetwork=true` enthält (Standard ist `false`), wird seine IP-Adresse nicht in eine andere Adresse übersetzt. Dies ist standardmäßig der Fall für die `kube-proxy` und Amazon-VPC-CNI-Plugin für Kubernetes-Pods, die in Ihrem Cluster ausgeführt werden. Für diese Pods ist die IP-Adresse dieselbe wie die primäre IP-Adresse des Knotens. Daher wird die IP-Adresse des Pods nicht übersetzt. Weitere Informationen über die `hostNetwork`-Einstellungen eines Pods finden Sie unter PodSpec v1 core in der Kubernetes-API-Referenz.

Begrenzen Sie den Pod-Datenverkehr mit Kubernetes-Netzwerkrichtlinien.

-Übersicht

Standardmäßig gibt es in Kubernetes keine Einschränkungen für IP-Adressen, Ports oder Verbindungen zwischen Pods in Ihrem Cluster oder zwischen Ihren Pods und Ressourcen in anderen Netzwerken. Sie können die Kubernetes-Netzwerkrichtlinie verwenden, um Netzwerk-Datenerkehr zu und von Ihren Pods einzuschränken. Weitere Informationen finden Sie unter Netzwerk-Richtlinien in der Kubernetes-Dokumentation.

Standardnetzwerkrichtlinie

Sie können den Standard verwenden`NetworkPolicy`, um den pod-to-pod Verkehr im Cluster zu segmentieren. Diese Netzwerkrichtlinien arbeiten auf den Ebenen 3 und 4 des OSI-Netzwerkmodells, sodass Sie den Datenverkehrsfluss auf IP-Adresse- oder Portebene innerhalb Ihres Amazon EKS-Clusters steuern können. Standardnetzwerkrichtlinien beziehen sich auf die Namespace-Ebene.

Anwendungsfälle

- Segmentieren Sie den Netzwerkverkehr zwischen Workloads, um sicherzustellen, dass nur verwandte Anwendungen miteinander kommunizieren können.
- Isolieren Sie Mandanten auf Namespace-Ebene mithilfe von Richtlinien, um die Netzwerktrennung durchzusetzen.

Beispiel

In der folgenden Richtlinie ist der ausgehende Datenverkehr aus den Webapp-Pods im Sun-Namespace eingeschränkt.

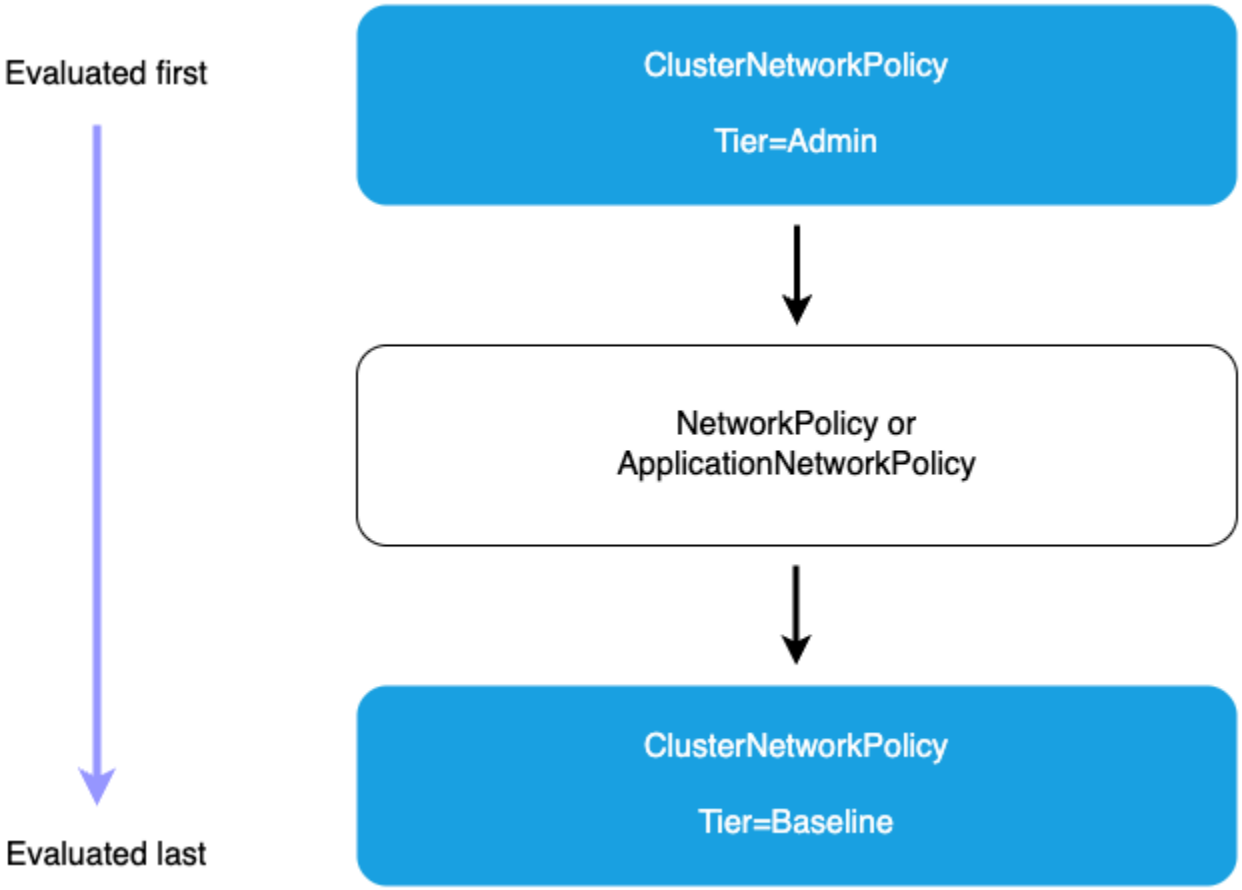
```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: webapp-egress-policy
  namespace: sun
spec:
  podSelector:
    matchLabels:
      role: webapp
  policyTypes:
  - Egress
  egress:
  - to:
    - namespaceSelector:
        matchLabels:
          name: moon
      podSelector:
        matchLabels:
          role: frontend
    ports:
    - protocol: TCP
      port: 8080
```

```
  - to:
    - namespaceSelector:
        matchLabels:
          name: stars
      podSelector:
        matchLabels:
          role: frontend
    ports:
    - protocol: TCP
      port: 8080
```

Die Richtlinie gilt für Pods mit dem Label `role: webapp` im Namespace. `sun`

- Zulässiger Verkehr: Pods mit dem Label `role: frontend` im `moon` Namespace am TCP-Port `8080`
- Zulässiger Verkehr: Pods mit der Labelrolle: Frontend im `stars` Namespace am TCP-Port `8080`
- Blockierter Verkehr: Jeglicher andere ausgehende Datenverkehr von `webapp` Pods wird implizit verweigert

Netzwerkrichtlinie für Administratoren (oder Cluster)



Sie können den verwenden`ClusterNetworkPolicy`, um einen Netzwerksicherheitsstandard durchzusetzen, der für den gesamten Cluster gilt. Anstatt wiederholt eine eigene Richtlinie für jeden Namespace zu definieren und zu verwalten, können Sie eine einzige Richtlinie verwenden, um die Netzwerkzugriffskontrollen für verschiedene Workloads im Cluster unabhängig von ihrem Namespace zentral zu verwalten.

Anwendungsfälle

- Verwalten Sie die Netzwerkzugriffskontrollen für alle (oder eine Teilmenge von) Workloads in Ihrem EKS-Cluster zentral.
- Definieren Sie einen standardmäßigen Netzwerksicherheitsstatus für den gesamten Cluster.
- Erweitern Sie die organisatorischen Sicherheitsstandards auf betrieblich effizientere Weise auf den gesamten Clusterbereich.

Beispiel

In der folgenden Richtlinie können Sie den Clusterverkehr von anderen Namespaces explizit blockieren, um den Netzwerkzugriff auf einen sensiblen Workload-Namespace zu verhindern.

```
apiVersion: networking.k8s.aws/v1alpha1
kind: ClusterNetworkPolicy
metadata:
  name: protect-sensitive-workload
spec:
  tier: Admin
  priority: 10
  subject:
    namespaces:
      matchLabels:
        kubernetes.io/metadata.name: earth
  ingress:
    - action: Deny
      from:
      - namespaces:
          matchLabels: {} # Match all namespaces.
      name: select-all-deny-all
```

Wichtige Hinweise

Netzwerkrichtlinien im Amazon VPC CNI-Plugin für Kubernetes werden in den unten aufgeführten Konfigurationen unterstützt.

- Version 1.21.0 (oder höher) des Amazon VPC CNI-Plug-ins für Standard- und Admin-Netzwerkrichtlinien.
- Für `IPv4`- oder `IPv6`-Adressen konfigurierter Cluster.
- Sie können Netzwerkrichtlinien mit Sicherheitsgruppen für Pods verwenden. Mit Netzwerkrichtlinien können Sie die gesamte Kommunikation innerhalb eines Clusters steuern. Mit Sicherheitsgruppen für Pods können Sie den Zugriff auf AWS Dienste von Anwendungen innerhalb eines Pods aus steuern.
- Sie können Netzwerkrichtlinien mit benutzerdefinierten Netzwerken und Präfixdelegierung verwenden.

Überlegungen

Architektur

- Wenn Sie Netzwerkrichtlinien des Amazon VPC CNI-Plug-ins für Kubernetes mit dem Amazon VPC CNI-Plugin für Kubernetes auf Ihren Cluster anwenden, können Sie die Richtlinien nur auf Amazon Linux-Knoten anwenden. EC2 Sie können nicht in Fargate- oder Windows-Knoten angewendet werden.
- Netzwerkrichtlinien gelten nur für `IPv4`- oder `IPv6`-Adressen, jedoch nicht für beide. In einem `IPv4`-Cluster weist die VPC CNI den Pods `IPv4`-Adressen zu und wendet `IPv4`-Richtlinien an. In einem `IPv6`-Cluster weist die VPC CNI den Pods `IPv6`-Adressen zu und wendet `IPv6`-Richtlinien an. Alle auf einen `IPv6`-Cluster angewendeten `IPv4`-Netzwerkrichtlinien werden ignoriert. Alle auf einen `IPv4`-Cluster angewendeten `IPv6`-Netzwerkrichtlinien werden ignoriert.

Netzwerkrichtlinien

- Netzwerkrichtlinien werden ausschließlich auf Pods angewendet, die Teil einer Bereitstellung sind. Auf eigenständige Pods, für die kein `metadata.ownerReferences` festgelegt ist, können keine Netzwerkrichtlinien angewendet werden.
- Sie können mehrere Netzwerkrichtlinien auf denselben Pods anwenden. Wenn zwei oder mehr Richtlinien konfiguriert werden, die denselben Pods auswählen, werden alle Richtlinien auf den Pod angewendet.
- Die maximale Anzahl an Kombinationen aus Ports und Protokollen für einen einzelnen IP-Adressbereich (CIDR) beträgt 24 über alle Ihre Netzwerkrichtlinien hinweg. Selektoren wie die Auflösung zu einem oder mehreren. `namespaceSelector` CIDRs Wenn mehrere Selektoren zu einem einzigen CIDR führen oder Sie denselben direkten CIDR mehrfach in denselben oder verschiedenen Netzwerkrichtlinien angeben, werden diese alle auf dieses Limit angerechnet.
- Für jeden Ihrer Kubernetes-Services muss der Port des Services mit dem Port des Containers identisch sein. Wenn Sie benannte Ports verwenden, verwenden Sie denselben Namen auch in der Servicespezifikation.

Netzwerkrichtlinien für Administratoren

1. Richtlinien auf Admin-Ebene (zuerst bewertet): Alle Richtlinien auf Admin-Ebene ClusterNetworkPolicies werden vor allen anderen Richtlinien bewertet. Innerhalb der Admin-

Ebene werden Richtlinien in der Reihenfolge ihrer Priorität verarbeitet (Nummer mit der niedrigsten Priorität zuerst). Der Aktionstyp bestimmt, was als Nächstes passiert.

- Aktion ablehnen (höchste Priorität): Wenn eine Admin-Richtlinie mit einer Ablehnungsaktion auf Traffic trifft, wird dieser Verkehr sofort blockiert, unabhängig von anderen Richtlinien. Es werden keine weiteren ClusterNetworkPolicy NetworkPolicy OD-Regeln verarbeitet. Dadurch wird sichergestellt, dass unternehmensweite Sicherheitskontrollen nicht durch Richtlinien auf Namespace-Ebene außer Kraft gesetzt werden können.
- Aktion zulassen: Nach der Auswertung der Ablehnungsregeln werden Administratorrichtlinien mit Zulassen-Aktionen in der Reihenfolge ihrer Priorität verarbeitet (die niedrigste Priorität zuerst). Wenn eine Aktion „Zulassen“ zutrifft, wird der Datenverkehr akzeptiert und es findet keine weitere Richtlinienbewertung statt. Diese Richtlinien können auf der Grundlage von Label-Selektoren Zugriff auf mehrere Namespaces gewähren und ermöglichen so eine zentrale Kontrolle darüber, welche Workloads auf bestimmte Ressourcen zugreifen können.
- Aktion weiterleiten: Bei der Weiterleitung von Aktionen in den Richtlinien der Admin-Ebene wird die Entscheidungsfindung an untergeordnete Ebenen delegiert. Wenn der Traffic mit einer Pass-Regel übereinstimmt, werden bei der Evaluierung alle verbleibenden Regeln auf Admin-Ebene für diesen Traffic übersprungen und direkt zur NetworkPolicy Stufe weitergeleitet. Auf diese Weise können Administratoren die Kontrolle über bestimmte Datenverkehrsmuster explizit an Anwendungsteams delegieren. Sie können beispielsweise Pass-Regeln verwenden, um die Verwaltung des Datenverkehrs innerhalb des Namespaces an Namespace-Administratoren zu delegieren und gleichzeitig strenge Kontrollen des externen Zugriffs aufrechtzuerhalten.

2. Netzwerkrichtlinienebene: Wenn keine Richtlinie auf Admin-Ebene mit „Verweigern“ oder „Zulassen“ übereinstimmt oder wenn eine Pass-Aktion gefunden wurde, werden Ressourcen mit NetworkPolicy Namespace-Bereich als Nächstes bewertet. Diese Richtlinien ermöglichen eine detaillierte Steuerung innerhalb der einzelnen Namespaces und werden von Anwendungsteams verwaltet. Namespace-bezogene Richtlinien können nur restriktiver sein als Admin-Richtlinien. Sie können die Ablehnungsentscheidung einer Admin-Richtlinie nicht außer Kraft setzen, aber sie können den Datenverkehr weiter einschränken, der durch Admin-Richtlinien zugelassen oder weitergeleitet wurde.
3. Admin-Richtlinien der Baseline-Stufe: Wenn keine Admin- oder Namespace-bezogenen Richtlinien dem Traffic entsprechen, wird die Baseline-Stufe bewertet. ClusterNetworkPolicies Diese bieten standardmäßige Sicherheitsvorkehrungen, die durch Richtlinien im Namespace-Bereich außer Kraft gesetzt werden können. So können Administratoren unternehmensweite Standardeinstellungen festlegen und den Teams gleichzeitig die Flexibilität geben, sie nach Bedarf

anzupassen. Basisrichtlinien werden in der Reihenfolge ihrer Priorität bewertet (niedrigste Priorität zuerst).

4. Standardverweigerung (wenn keine Richtlinien zutreffen): Dieses deny-by-default Verhalten stellt sicher, dass nur explizit zugelassene Verbindungen zugelassen werden, wodurch ein hohes Maß an Sicherheit gewährleistet wird.

Migration

- Falls Ihr Cluster derzeit eine Drittanbieterlösung zur Verwaltung von Kubernetes-Netzwerkrichtlinien verwendet, können Sie dieselben Richtlinien mit dem Amazon-VPC-CNI-Plugin für Kubernetes nutzen. Sie müssen jedoch Ihre vorhandene Lösung entfernen, damit sie nicht dieselben Richtlinien verwaltet.

> **Warning**
>
> Wir empfehlen, nach dem Entfernen einer Netzwerkrichtlinienlösung alle Knoten zu ersetzen, auf welche die Netzwerkrichtlinienlösung angewendet wurde. Der Grund dafür ist, dass die Datenverkehrsregeln möglicherweise von einem Pod der Lösung zurückgelassen werden, wenn dieser plötzlich beendet wird.

Installation

- Das Netzwerkrichtlinienfeature erstellt und erfordert die Definition einer benutzerdefinierten `PolicyEndpoint`-Ressource (Custom Resource Definition, CRD) namens `policyendpoints.networking.k8s.aws`. `PolicyEndpoint`-Objekte der benutzerdefinierten Ressource werden von Amazon EKS verwaltet. Ändern oder löschen Sie diese Ressourcen nicht.
- Wenn Sie Pods ausführen, die die IAM-Anmeldeinformationen der Instanzrolle verwenden oder eine Verbindung zum EC2 IMDS herstellen, achten Sie darauf, dass Netzwerkrichtlinien den EC2 Zugriff auf das IMDS blockieren würden. Möglicherweise müssen Sie eine Netzwerkrichtlinie hinzufügen, um den Zugriff auf IMDS zu ermöglichen. EC2 Weitere Informationen finden Sie unter Instance-Metadaten und Benutzerdaten im EC2 Amazon-Benutzerhandbuch.

  Pods, die IAM-Rollen für Dienstkonten oder EKS Pod Identity verwenden, greifen nicht auf EC2 IMDS zu.

- Das Amazon-VPC-CNI-Plugin für Kubernetes wendet keine Netzwerkrichtlinien auf zusätzliche Netzwerkschnittstellen für jeden Pod an, sondern nur auf die primäre Schnittstelle für jeden Pod (`eth0`). Das hat Auswirkungen auf folgende Architekturen:
  - `IPv6`-Pods, bei denen die Variable `ENABLE_V4_EGRESS` auf `true` festgelegt ist. Diese Variable ermöglicht es der `IPv4` Ausgangsfunktion, die IPv6 Pods mit Endpunkten zu verbinden, z. B. mit `IPv4` Endpunkten außerhalb des Clusters. Die `IPv4` Ausgangsfunktion erstellt eine zusätzliche Netzwerkschnittstelle mit einer lokalen Loopback-Adresse. IPv4
  - Bei Verwendung verketteter Netzwerk-Plugins wie Multus. Da diese Plugins jedem Pod Netzwerkschnittstellen hinzufügen, werden Netzwerkrichtlinien nicht auf die verketteten Netzwerk-Plugins angewendet.

Pod-Netzwerk-Datenverkehr mit Kubernetes-Netzwerkrichtlinien einschränken

Sie können eine Kubernetes-Netzwerkrichtlinie verwenden, um den Netzwerk-Datenverkehr zu und von Ihren Pods einzuschränken. Weitere Informationen finden Sie unter Netzwerk-Richtlinien in der Kubernetes-Dokumentation.

Um dieses Feature nutzen zu können, müssen Sie Folgendes konfigurieren:

1. Richten Sie die Richtlinien-Durchsetzung beim Pod-Startup ein. Dies erfolgt im `aws-node`-Container des VPC-CNI-`DaemonSet`.
2. Aktivieren Sie den Netzwerkrichtlinienparameter für das Add-On.
3. Cluster für die Verwendung der Kubernetes-Netzwerkrichtlinie konfigurieren

Lesen Sie die Überlegungen noch einmal durch, bevor Sie beginnen. Weitere Informationen finden Sie unter the section called “Überlegungen”.

Voraussetzungen

Für das Feature gelten folgende Voraussetzungen:

Minimale Cluster-Version

Ein vorhandener Amazon-EKS-Cluster. Informationen zum Bereitstellen finden Sie unter Erste Schritte. Auf dem Cluster muss eine der in der folgenden Tabelle aufgeführten Kubernetes-Versionen und Plattform-Versionen ausgeführt werden. Beachten Sie, dass alle Kubernetes- und Plattformversionen, die über die aufgeführten hinausgehen, ebenfalls unterstützt werden. Sie können

Ihre aktuelle Kubernetes-Version überprüfen, indem Sie *my-cluster* den folgenden Befehl durch den Namen Ihres Clusters ersetzen und dann den geänderten Befehl ausführen:

```
aws eks describe-cluster --name my-cluster --query cluster.version --output text
```

Kubernetes-Version	Plattformversion
1.27.4	eks.5
1.26.7	eks.6

VPC-CNI-Mindestversion

Um sowohl Standard-Kubernetes-Netzwerkrichtlinien als auch Admin-Netzwerkrichtlinien zu erstellen, müssen Sie die Version 1.21 des VPC-CNI-Plug-ins ausführen. Sie können Ihre aktuelle Version mit dem folgenden Befehl überprüfen:

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni: |
 cut -d : -f 3
```

Wenn Ihre Version älter als 1.21 ist, finden Sie unter the section called “Aktualisierung (EKS-Add-On)” Informationen zum Aktualisieren des Plugins auf Version 1.21 oder höher.

Linux-Kernel-Mindestversion

Ihre Knoten müssen eine Linux-Kernelversion ab Version 5.10 haben. Sie können Ihre Kernelversion mit uname -r überprüfen. Wenn Sie die neuesten Versionen von Amazon EKS-optimiertem Amazon Linux, Amazon EKS-optimiertem beschleunigtem Amazon Linux und Bottlerocket AMIs verwenden AMIs, verfügen diese bereits über die erforderliche Kernelversion.

Amazon-EKS-optimierte Amazon-Linux-AMI-Versionen ab v20231116 verfügen über Kernel-Version 5.10.

Schritt 1: Richtlinien-Durchsetzung beim Startup des Pods einrichten

Das Amazon-VPC-CNI-Plugin für Kubernetes konfiguriert die Netzwerkrichtlinien für Pods parallel zur Pod-Bereitstellung. Bis alle Richtlinien für den neuen Pod konfiguriert sind, starten die Container im neuen Pod mit einer Standard-Zulassungsrichtlinie. Dies wird als Standardmodus bezeichnet.

Eine Standard-Zulassungsrichtlinie bedeutet, dass der gesamte eingehende und ausgehende Datenverkehr zu und von den neuen Pods zugelassen wird. Beispielsweise werden für die Pods keine Firewall-Regeln durchgesetzt (der gesamte Datenverkehr ist zulässig), bis der neue Pod mit den aktiven Richtlinien aktualisiert wird.

Wenn die `NETWORK_POLICY_ENFORCING_MODE`-Variable auf `strict` gesetzt ist, starten Pods, die VPC CNI verwenden, mit einer Standard-Verweigerungsrichtlinie, anschließend werden die Richtlinien konfiguriert. Dies wird als strikter Modus bezeichnet. Im strikten Modus müssen Sie für jeden Endpunkt in Ihrem Cluster, auf den Ihre Pods zugreifen müssen, eine Netzwerkrichtlinie haben. Beachten Sie, dass diese Anforderung für die CoreDNS-Pods gilt. Die Standard-Verweigerungsrichtlinie ist für Pods mit Host-Netzwerk nicht konfiguriert.

Sie können die Standard-Netzwerkrichtlinie ändern, indem Sie die Umgebungsvariable `NETWORK_POLICY_ENFORCING_MODE` im `aws-node`-Container des VPC-CNI-`DaemonSet` auf `strict` setzen.

```
env:
  - name: NETWORK_POLICY_ENFORCING_MODE
    value: "strict"
```

Schritt 2: Netzwerk-Richtlinienparameter für das Add-On aktivieren

Das Netzwerkrichtlinien-Feature verwendet standardmäßig den Port `8162` auf dem Knoten für Metriken. Außerdem verwendet die Funktion den Port `8163` für Gesundheitstests. Wenn Sie eine andere Anwendung auf die Knoten oder innerhalb von Pods ausführen, die diese Ports verwenden müssen, kann die App nicht ausgeführt werden. Ab der VPC CNI-Version `v1.14.1` oder höher können Sie diese Ports ändern.

Führen Sie die folgenden Schritte aus, um den Netzwerk-Richtlinienparameter für das Add-On zu aktivieren.

AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich die Option Cluster aus. Wählen Sie dann den Namen des Clusters aus, für den Sie das Amazon-VPC-CNI-Add-on konfigurieren möchten.
3. Wählen Sie die Registerkarte Add-ons.
4. Wählen Sie das Kästchen oben rechts in der Add-On-Box aus und wählen Sie dann Edit (Bearbeiten).

5. Auf der Seite **`Amazon VPC CNI`** konfigurieren:

   a. Wählen Sie in der Liste Version eine Version vom Typ `v1.14.0-eksbuild.3` oder höher aus.

   b. Erweitern Sie Optionale Konfigurationseinstellungen.

   c. Geben Sie den JSON-Schlüssel `"enableNetworkPolicy":` und den Wert `"true"` in Konfigurationswerte ein. Der resultierende Text muss ein gültiges JSON-Objekt sein. Wenn dieser Schlüssel und dieser Wert die einzigen Daten im Textfeld sind, setzen Sie den Schlüssel und den Wert in geschweifte Klammern `{ }`.

      Im folgenden Beispiel ist das Netzwerkrichtlinien-Feature aktiviert, und Metriken und Integritätsprüfungen sind auf die Standardportnummern eingestellt:

      ```
      {
          "enableNetworkPolicy": "true",
          "nodeAgent": {
              "healthProbeBindAddr": "8163",
              "metricsBindAddr": "8162"
          }
      }
      ```

Helm

Wenn Sie das Amazon-VPC-CNI-Plugin für Kubernetes über `helm` installiert haben, können Sie die Konfiguration aktualisieren, um die Ports zu ändern.

1. Führen Sie den folgenden Befehl aus, um die Ports zu ändern. Legen Sie die Portnummer im Wert für Schlüssel `nodeAgent.metricsBindAddr` bzw. Schlüssel `nodeAgent.healthProbeBindAddr` fest.

   ```
   helm upgrade --set nodeAgent.metricsBindAddr=8162 --set
    nodeAgent.healthProbeBindAddr=8163 aws-vpc-cni --namespace kube-system eks/aws-vpc-
   cni
   ```

kubectl

1. Öffnen Sie das `DaemonSet` `aws-node` in Ihrem Editor.

   ```
   kubectl edit daemonset -n kube-system aws-node
   ```

2. Ersetzen Sie die Portnummern in den folgenden Befehlsargumenten im `args:` im `aws-network-policy-agent`-Container im `aws-node`-DaemonSet-Manifest von VPC CNI.

```
    - args:
            - --metrics-bind-addr=:8162
            - --health-probe-bind-addr=:8163
```

Schritt 3: Cluster für die Verwendung von Kubernetes-Netzwerkrichtlinien konfigurieren

Sie können dies für ein Amazon-EKS-Add-On oder ein selbstverwaltetes Add-On einstellen.

Amazon-EKS-Add-On

Mithilfe der AWS CLI können Sie den Cluster für die Verwendung von Kubernetes-Netzwerkrichtlinien konfigurieren, indem Sie den folgenden Befehl ausführen. Ersetzen Sie `my-cluster` durch den Namen Ihres Clusters und den IAM-Rollen-ARN durch die Rolle, die Sie verwenden.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-version
 v1.14.0-eksbuild.3 \
    --service-account-role-arn arn:aws: iam::123456789012:role/AmazonEKSVPCCNIRole \
    --resolve-conflicts PRESERVE --configuration-values '{"enableNetworkPolicy":
 "true"}'
```

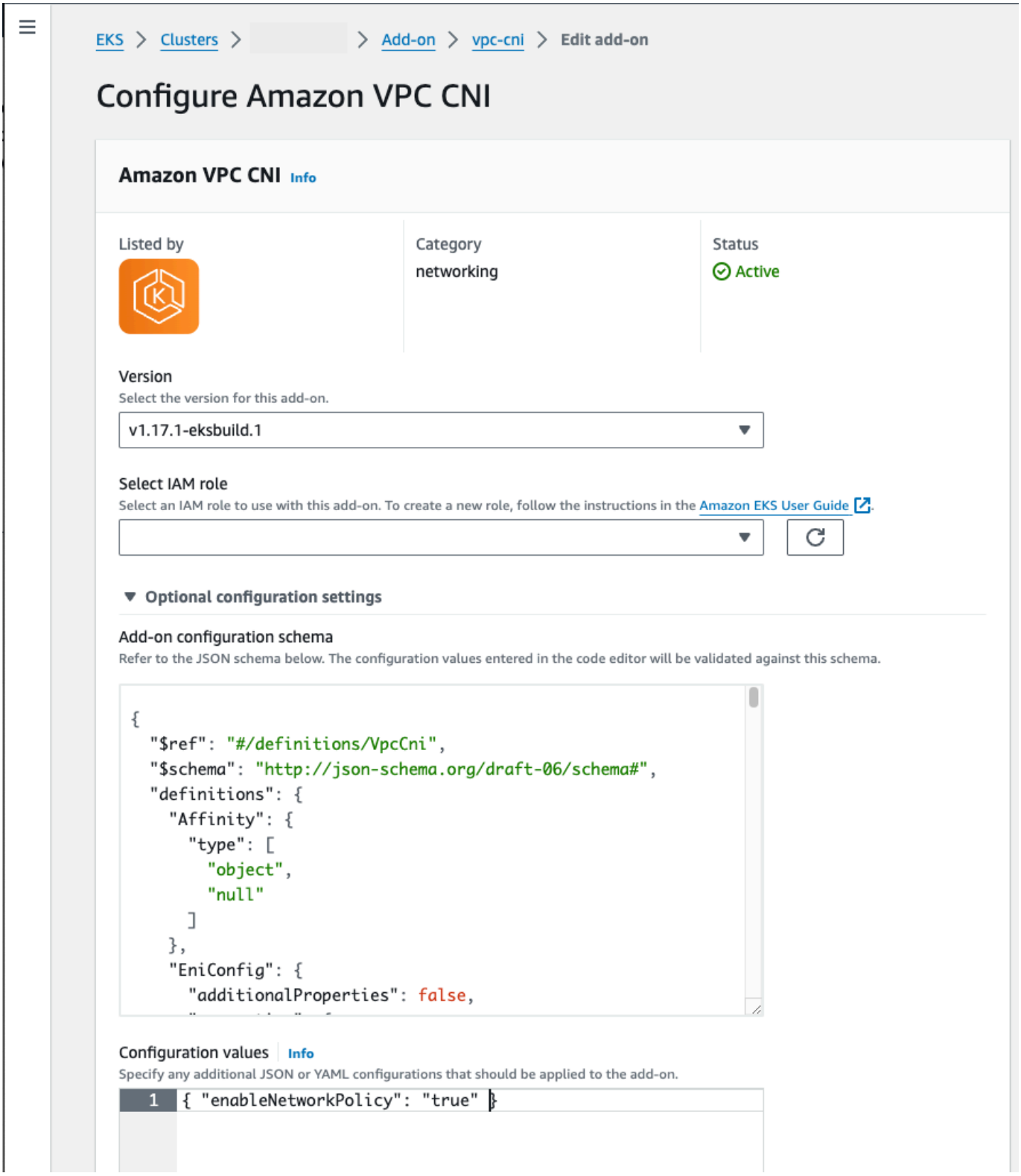
Gehen Sie wie folgt vor, um dies mit der AWS Management Console zu konfigurieren:

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich die Option Cluster aus. Wählen Sie dann den Namen des Clusters aus, für den Sie das Amazon-VPC-CNI-Add-on konfigurieren möchten.
3. Wählen Sie die Registerkarte Add-ons.
4. Wählen Sie das Kästchen oben rechts in der Add-On-Box aus und wählen Sie dann Edit (Bearbeiten).
5. Auf der Seite **`Amazon VPC CNI`** konfigurieren:
   a. Wählen Sie in der Liste Version eine Version vom Typ `v1.14.0-eksbuild.3` oder höher aus.
   b. Erweitern Sie Optionale Konfigurationseinstellungen.
   c. Geben Sie den JSON-Schlüssel `"enableNetworkPolicy":` und den Wert `"true"` in Konfigurationswerte ein. Der resultierende Text muss ein gültiges JSON-Objekt sein. Wenn dieser Schlüssel und dieser Wert die einzigen Daten im Textfeld sind, setzen Sie den Schlüssel

und den Wert in geschweifte Klammern { }. Das folgende Beispiel veranschaulicht, dass die Netzwerkrichtlinie aktiviert ist:

```
{ "enableNetworkPolicy": "true" }
```

Der folgende Screenshot zeigt ein Beispiel für dieses Szenario.


EKS > Clusters > > Add-on > vpc-cni > Edit add-on
Configure Amazon VPC CNI
Amazon VPC CNI Info
Listed by
Category
networking
Status
Active
Version
Select the version for this add-on.
v1.17.1-eksbuild.1
Select IAM role
Select an IAM role to use with this add-on. To create a new role, follow the instructions in the Amazon EKS User Guide.
Optional configuration settings
Add-on configuration schema
Refer to the JSON schema below. The configuration values entered in the code editor will be validated against this schema.
{
"$ref": "#/definitions/VpcCni",
"$schema": "http://json-schema.org/draft-06/schema#",
"definitions": {
"Affinity": {
"type": [
"object",
"null"
]
},
"EniConfig": {
"additionalProperties": false,
Configuration values Info
Specify any additional JSON or YAML configurations that should be applied to the add-on.
1 { "enableNetworkPolicy": "true" }

Selbstverwaltetes Add-On

Helm

Wenn Sie das Amazon-VPC-CNI-Plugin für Kubernetes über `helm` installiert haben, können Sie die Konfiguration aktualisieren, um die Netzwerkrichtlinie zu aktivieren.

1. Führen Sie den folgenden Befehl aus, um die Netzwerkrichtlinie zu aktivieren.

```
helm upgrade --set enableNetworkPolicy=true aws-vpc-cni --namespace kube-system eks/
aws-vpc-cni
```

kubectl

1. Öffnen Sie das `ConfigMap` `amazon-vpc-cni` in Ihrem Editor.

```
kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
```

2. Fügen Sie `data` in `ConfigMap` die folgende Zeile hinzu:

```
enable-network-policy-controller: "true"
```

   Sobald Sie die Zeile hinzugefügt haben, sollte `ConfigMap` wie das folgende Beispiel aussehen.

```
apiVersion: v1
 kind: ConfigMap
 metadata:
  name: amazon-vpc-cni
  namespace: kube-system
 data:
  enable-network-policy-controller: "true"
```

3. Öffnen Sie das `DaemonSet` `aws-node` in Ihrem Editor.

```
kubectl edit daemonset -n kube-system aws-node
```

   a. Ersetzen Sie im Befehlsargument `--enable-network-policy=false` in `args:` im Container `aws-network-policy-agent` im DaemonSet-Manifest `aws-node` von VPC CNI den Wert `false` durch `true`.

```
    - args:
       - --enable-network-policy=true
```

Schritt 4. Nächste Schritte

Nachdem Sie die Konfiguration abgeschlossen haben, überprüfen Sie, ob die `aws-node`-Pods in Ihrem Cluster ausgeführt werden.

```
kubectl get pods -n kube-system | grep 'aws-node\|amazon'
```

Eine Beispielausgabe sieht wie folgt aus.

```
aws-node-gmqp7                                          2/2     Running   1 (24h ago)
 24h
aws-node-prnsh                                          2/2     Running   1 (24h ago)
 24h
```

In den `aws-node`-Pods der Versionen `1.14` und höher befinden sich zwei Container. In früheren Versionen und wenn die Netzwerkrichtlinie deaktiviert ist, gibt es nur einen einzigen Container in den `aws-node`-Pods.

Sie können nun Kubernetes-Netzwerkrichtlinien in Ihrem Cluster bereitstellen.

Um Kubernetes-Netzwerkrichtlinien zu implementieren, können Sie Kubernetes `NetworkPolicy` oder `ClusterNetworkPolicy` Objekte erstellen und diese in Ihrem Cluster bereitstellen. `NetworkPolicy`Objekte sind auf einen Namespace beschränkt, während `ClusterNetworkPolicy` Objekte auf den gesamten Cluster oder mehrere Namespaces beschränkt werden können. Sie implementieren Richtlinien, um Datenverkehr zwischen Pods auf der Grundlage von Bezeichnungsselektoren, Namespaces und IP-Adressbereichen zuzulassen oder zu verweigern. Weitere Informationen zum Erstellen von `NetworkPolicy`-Objekten finden Sie unter Netzwerkrichtlinien in der Kubernetes-Dokumentation.

Die Durchsetzung von Kubernetes-`NetworkPolicy`-Objekten wird mithilfe des Extended Berkeley Packet Filter (eBPF) implementiert. Im Vergleich zu `iptables`-basierten Implementierungen bietet er geringere Latenzzeiten und Leistungsmerkmale, einschließlich einer geringeren CPU-Auslastung und der Vermeidung sequenzieller Suchvorgänge. Darüber hinaus bieten eBPF-Proben Zugriff auf kontextreiche Daten, die bei der Fehlersuche in komplexen Problemen auf Kernel-Ebene helfen und die Beobachtbarkeit verbessern. Amazon EKS unterstützt einen eBPF-basierten Exporter, der die

Proben nutzt, um Richtlinienergebnisse auf jedem Knoten zu protokollieren und die Daten an externe Protokollsammler zu exportieren, um bei der Fehlersuche zu helfen. Weitere Informationen finden Sie in der eBPF-Dokumentation.

Deaktivierung der Kubernetes-Netzwerkrichtlinien für den Netzwerk-Datenverkehr von Amazon EKS Pods

Deaktivierung von Kubernetes-Netzwerkrichtlinien zur Beendigung der Einschränkung des Netzwerk-Datenverkehr von Amazon EKS Pods

1. Auflistung aller Kubernetes-Netzwerkrichtlinien

```
kubectl get netpol -A
```

2. Löschen Sie jede Kubernetes-Netzwerkrichtlinie. Sie müssen alle Netzwerkrichtlinien löschen, bevor Sie Netzwerkrichtlinien deaktivieren können.

```
kubectl delete netpol <policy-name>
```

3. Öffnen Sie das aws-node DaemonSet in Ihrem Editor.

```
kubectl edit daemonset -n kube-system aws-node
```

4. Ersetzen Sie im Befehlsargument `--enable-network-policy=true` in `args:` im Container `aws-network-policy-agent` im DaemonSet-Manifest `aws-node` von VPC CNI den Wert `true` durch `false`.

```
    - args:
       - --enable-network-policy=true
```

Fehlerbehebung bei Kubernetes-Netzwerkrichtlinien für Amazon EKS

Dies ist die Anleitung zur Fehlerbehebung für die Netzwerkrichtlinien-Feature von Amazon VPC CNI.

Dieser Leitfaden behandelt:

- Installationsinformationen, CRD- und RBAC-Berechtigungen the section called “Neue `policyendpoints` CRD und Berechtigungen”
- Protokolle, die bei der Diagnose von Netzwerkrichtlinienproblemen the section called “Netzwerkrichtlinien-Protokolle” untersucht werden müssen

- Ausführung der eBPF-SDK-Tool-Sammlung zur Fehlerbehebung
- Bekannte Probleme und Lösungen the section called “Bekannte Probleme und Lösungen ”

> **Note**
>
> Beachten Sie, dass Netzwerkrichtlinien nur auf Pods angewendet werden, die von Kubernetes-Bereitstellungen erstellt werden. Weitere Einschränkungen der Netzwerkrichtlinien im VPC CNI finden Sie unter the section called “Überlegungen”.

Sie können Fehler bei Netzwerkverbindungen, die Netzwerkrichtlinien verwenden, beheben und untersuchen, indem Sie die Netzwerkrichtlinien-Protokolle lesen und Tools aus dem eBPF-SDK ausführen.

Neue **`policyendpoints`** CRD und Berechtigungen

- CRD: `policyendpoints.networking.k8s.aws`
- Kubernetes API: `apiservice` mit dem Namen `v1.networking.k8s.io`
- Kubernetes-Ressource: `Kind: NetworkPolicy`
- RBAC: `ClusterRole` mit dem Namen `aws-node` (VPC CNI), `ClusterRole` mit dem Namen `eks:network-policy-controller` (Netzwerkrichtlinien-Controller in der EKS-Cluster-Steuerebene)

Für die Netzwerkrichtlinie erstellt das VPC CNI ein neues `CustomResourceDefinition` (CRD) mit dem Namen `policyendpoints.networking.k8s.aws`. Das VPC-CNI muss über Berechtigungen zum Erstellen der CRD und zum Erstellen CustomResources (CR) dieser und der anderen von der VPC CNI () installierten CRD verfügen. `eniconfigs.crd.k8s.amazonaws.com` Beide CRDs sind in der Datei unter verfügbar. `crds.yaml` GitHub Insbesondere muss das VPC CNI über die Verb-Berechtigungen „get“, „list“ und „watch“ für `policyendpoints` verfügen.

Die Kubernetes-Netzwerkrichtlinie ist Teil des `apiservice` mit dem Namen `v1.networking.k8s.io`, und dieser `apiversion: networking.k8s.io/v1` befindet sich in Ihren YAML-Richtliniendateien. Die VPC CNI `DaemonSet` muss über die erforderlichen Berechtigungen verfügen, um diesen Teil der Kubernetes-API nutzen zu können.

Die VPC-CNI-Berechtigungen befinden sich in einem `ClusterRole` mit dem Namen `aws-node`. Beachten Sie, dass `ClusterRole`-Objekte nicht in Namespaces gruppiert sind. Nachfolgend wird der `aws-node` eines Clusters dargestellt:

```
kubectl get clusterrole aws-node -o yaml
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  labels:
    app.kubernetes.io/instance: aws-vpc-cni
    app.kubernetes.io/managed-by: Helm
    app.kubernetes.io/name: aws-node
    app.kubernetes.io/version: v1.19.4
    helm.sh/chart: aws-vpc-cni-1.19.4
    k8s-app: aws-node
  name: aws-node
rules:
- apiGroups:
  - crd.k8s.amazonaws.com
  resources:
  - eniconfigs
  verbs:
  - list
  - watch
  - get
- apiGroups:
  - ""
  resources:
  - namespaces
  verbs:
  - list
  - watch
  - get
- apiGroups:
  - ""
  resources:
  - pods
  verbs:
  - list
  - watch
  - get
- apiGroups:
```

```
  - ""
  resources:
  - nodes
  verbs:
  - list
  - watch
  - get
- apiGroups:
  - ""
  - events.k8s.io
  resources:
  - events
  verbs:
  - create
  - patch
  - list
- apiGroups:
  - networking.k8s.aws
  resources:
  - policyendpoints
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - networking.k8s.aws
  resources:
  - policyendpoints/status
  verbs:
  - get
- apiGroups:
  - vpcresources.k8s.aws
  resources:
  - cninodes
  verbs:
  - get
  - list
  - watch
  - patch
```

Zudem wird in der Steuerebene jedes EKS-Clusters ein neuer Controller ausgeführt. Der Controller verwendet die Berechtigungen des `ClusterRole` mit dem Namen `eks:network-policy-`

`controller`. Nachfolgend wird der `eks:network-policy-controller` eines Clusters dargestellt:

```
kubectl get clusterrole eks:network-policy-controller -o yaml
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  labels:
    app.kubernetes.io/name: amazon-network-policy-controller-k8s
  name: eks:network-policy-controller
rules:
- apiGroups:
  - ""
  resources:
  - namespaces
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - ""
  resources:
  - pods
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - ""
  resources:
  - services
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - networking.k8s.aws
  resources:
  - policyendpoints
  verbs:
  - create
  - delete
```

```
  - get
  - list
  - patch
  - update
  - watch
- apiGroups:
  - networking.k8s.aws
  resources:
  - policyendpoints/finalizers
  verbs:
  - update
- apiGroups:
  - networking.k8s.aws
  resources:
  - policyendpoints/status
  verbs:
  - get
  - patch
  - update
- apiGroups:
  - networking.k8s.io
  resources:
  - networkpolicies
  verbs:
  - get
  - list
  - patch
  - update
  - watch
```

Netzwerkrichtlinien-Protokolle

Jede Entscheidung des VPC CNI, ob Verbindungen durch Netzwerkrichtlinien zugelassen oder verweigert werden, wird in Ablaufprotokollen festgehalten. Die Netzwerkrichtlinien-Protokolle auf jedem Knoten enthalten die Flussprotokolle für jeden Pod, der über eine Netzwerkrichtlinie verfügt. Netzwerkrichtlinien-Protokolle werden unter `/var/log/aws-routed-eni/network-policy-agent.log` gespeichert. Das folgende Beispiel stammt aus einer `network-policy-agent.log`-Datei:

```
{"level":"info","timestamp":"2023-05-30T16:05:32.573Z","logger":"ebpf-
client","msg":"Flow Info: ","Src
IP":"192.168.87.155","Src Port":38971,"Dest IP":"64.6.160","Dest
```

```
Port":53,"Proto":"UDP","Verdict":"ACCEPT"}
```

Netzwerkrichtlinien-Protokolle sind standardmäßig deaktiviert. Um die Netzwerkrichtlinien-Protokolle zu aktivieren, führen Sie die folgenden Schritte aus:

> **Note**
>
> Netzwerkrichtlinien-Protokolle erfordern eine zusätzliche 1 vCPU für den `aws-network-policy-agent`-Container im `aws-node-DaemonSet`-Manifest von VPC CNI.

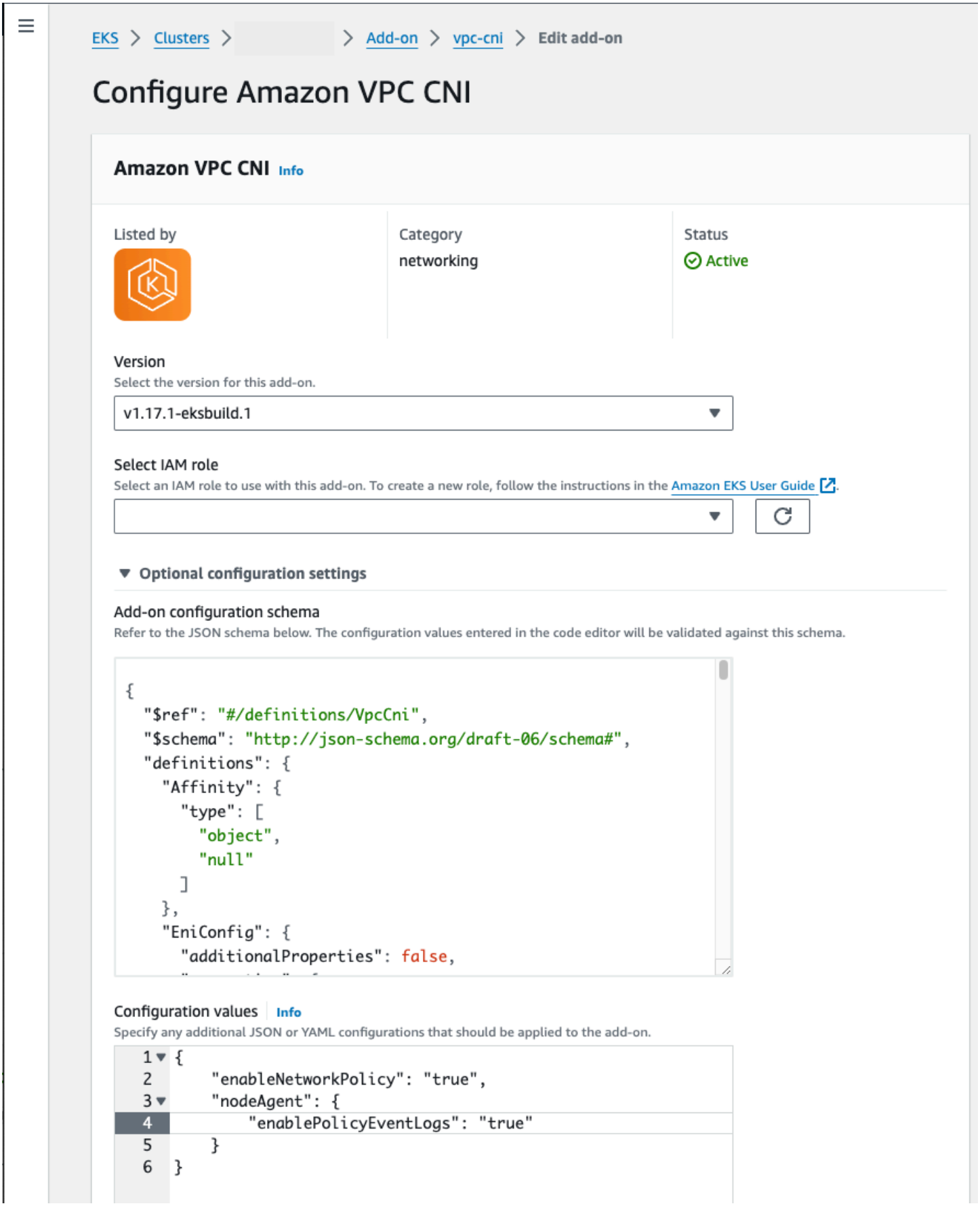
Amazon-EKS-Add-On

AWS-Managementkonsole

a. Öffnen Sie die Amazon-EKS-Konsole.
b. Wählen Sie im linken Navigationsbereich die Option Cluster aus. Wählen Sie dann den Namen des Clusters aus, für den Sie das Amazon-VPC-CNI-Add-on konfigurieren möchten.
c. Wählen Sie die Registerkarte Add-ons.
d. Wählen Sie das Kästchen oben rechts in der Add-On-Box aus und wählen Sie dann Edit (Bearbeiten).
e. Auf der Seite ***Amazon VPC CNI*** konfigurieren:
   i. Wählen Sie `v1.14.0-eksbuild.3` oder eine neuere Version in der Dropdown-Liste aus.
   ii. Erweitern Sie Optionale Konfigurationseinstellungen.
   iii. Geben Sie den JSON-Schlüssel der obersten Ebene `"nodeAgent":` ein, und der Wert ist ein Objekt mit dem Schlüssel `"enablePolicyEventLogs":` und dem Wert `"true"` in Konfigurationswerte. Der resultierende Text muss ein gültiges JSON-Objekt sein. Das folgende Beispiel zeigt, dass Netzwerkrichtlinien und Netzwerkrichtlinien-Protokolle aktiviert sind und die Netzwerkrichtlinien-Protokolle an CloudWatch Logs gesendet werden:

```
{
    "enableNetworkPolicy": "true",
    "nodeAgent": {
        "enablePolicyEventLogs": "true"
    }
}
```

Der folgende Screenshot zeigt ein Beispiel für dieses Szenario.


EKS > Clusters > > Add-on > vpc-cni > Edit add-on
Configure Amazon VPC CNI
Amazon VPC CNI Info
Listed by
Category
networking
Status
Active
Version
Select the version for this add-on.
v1.17.1-eksbuild.1
Select IAM role
Select an IAM role to use with this add-on. To create a new role, follow the instructions in the Amazon EKS User Guide.
Optional configuration settings
Add-on configuration schema
Refer to the JSON schema below. The configuration values entered in the code editor will be validated against this schema.
{
"$ref": "#/definitions/VpcCni",
"$schema": "http://json-schema.org/draft-06/schema#",
"definitions": {
"Affinity": {
"type": [
"object",
"null"
]
},
"EniConfig": {
"additionalProperties": false,
Configuration values Info
Specify any additional JSON or YAML configurations that should be applied to the add-on.
1 {
2 "enableNetworkPolicy": "true",
3 "nodeAgent": {
4 "enablePolicyEventLogs": "true"
5 }
6 }

AWS CLI

a. Führen Sie den folgenden AWS CLI-Befehl aus. Ersetzen Sie `my-cluster` durch den Namen Ihres Clusters und den IAM-Rollen-ARN durch die Rolle, die Sie verwenden.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version v1.14.0-eksbuild.3 \
    --service-account-role-arn arn:aws: iam::123456789012:role/AmazonEKSVPCCNIRole
 \
    --resolve-conflicts PRESERVE --configuration-values '{"nodeAgent":
 {"enablePolicyEventLogs": "true"}}'
```

Selbstverwaltetes Add-On

Helm

Wenn Sie das Amazon-VPC-CNI-Plugin für Kubernetes über `helm` installiert haben, können Sie die Konfiguration aktualisieren, um die Netzwerkrichtlinien-Protokolle zu schreiben.

a. Führen Sie den folgenden Befehl aus, um die Netzwerkrichtlinie zu aktivieren.

```
helm upgrade --set nodeAgent.enablePolicyEventLogs=true aws-vpc-cni --namespace
 kube-system eks/aws-vpc-cni
```

kubectl

Wenn Sie das Amazon-VPC-CNI-Plugin für Kubernetes über `kubectl` installiert haben, können Sie die Konfiguration aktualisieren, um die Netzwerkrichtlinien-Protokolle zu schreiben.

a. Öffnen Sie das `DaemonSet` `aws-node` in Ihrem Editor.

```
kubectl edit daemonset -n kube-system aws-node
```

b. Ersetzen Sie das `false` durch `true` im Befehlsargument `--enable-policy-event-logs=false` im `args:` im `aws-network-policy-agent`-Container im VPC CNI `aws-node` `DaemonSet`-Manifest.

```
    - args:
       - --enable-policy-event-logs=true
```

Netzwerkrichtlinien-Protokolle an Amazon CloudWatch Logs senden

Sie können die Netzwerkrichtlinienprotokolle mithilfe von Diensten wie Amazon CloudWatch Logs überwachen. Sie können die folgenden Methoden verwenden, um die Netzwerkrichtlinienprotokolle an Logs zu CloudWatch senden.

Bei EKS-Clustern befinden sich die Richtlinien-Protokolle unter `/aws/eks/`*`cluster-name`*`/cluster/` und bei selbstverwalteten K8S-Clustern sind die Protokolle unter `/aws/k8s-cluster/cluster/` platziert.

Senden von Netzwerkrichtlinien-Protokollen mit dem Amazon-VPC-CNI-Plugin für Kubernetes

Wenn Sie die Netzwerkrichtlinie aktivieren, wird den `aws-node`-Pods ein zweiter Container für einen Konten-Agent hinzugefügt. Dieser Node-Agent kann die CloudWatch Netzwerkrichtlinien-Protokolle an Logs senden.

> **Note**
>
> Nur die Netzwerkrichtlinien-Protokolle werden vom Knoten-Agent gesendet. Andere von VPC CNI erstellte Protokolle sind nicht enthalten.

Voraussetzungen

- Fügen Sie der IAM-Rolle, die Sie für VPC CNI verwenden, die folgenden Berechtigungen als Abschnitt oder separate Richtlinie hinzu.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups",
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        }
```

```
    ]
}
```

Amazon-EKS-Add-On

AWS-Managementkonsole

a. Öffnen Sie die Amazon-EKS-Konsole.

b. Wählen Sie im linken Navigationsbereich die Option Cluster aus. Wählen Sie dann den Namen des Clusters aus, für den Sie das Amazon-VPC-CNI-Add-on konfigurieren möchten.

c. Wählen Sie die Registerkarte Add-ons.

d. Wählen Sie das Kästchen oben rechts in der Add-On-Box aus und wählen Sie dann Edit (Bearbeiten).

e. Auf der Seite ***Amazon VPC CNI*** konfigurieren:

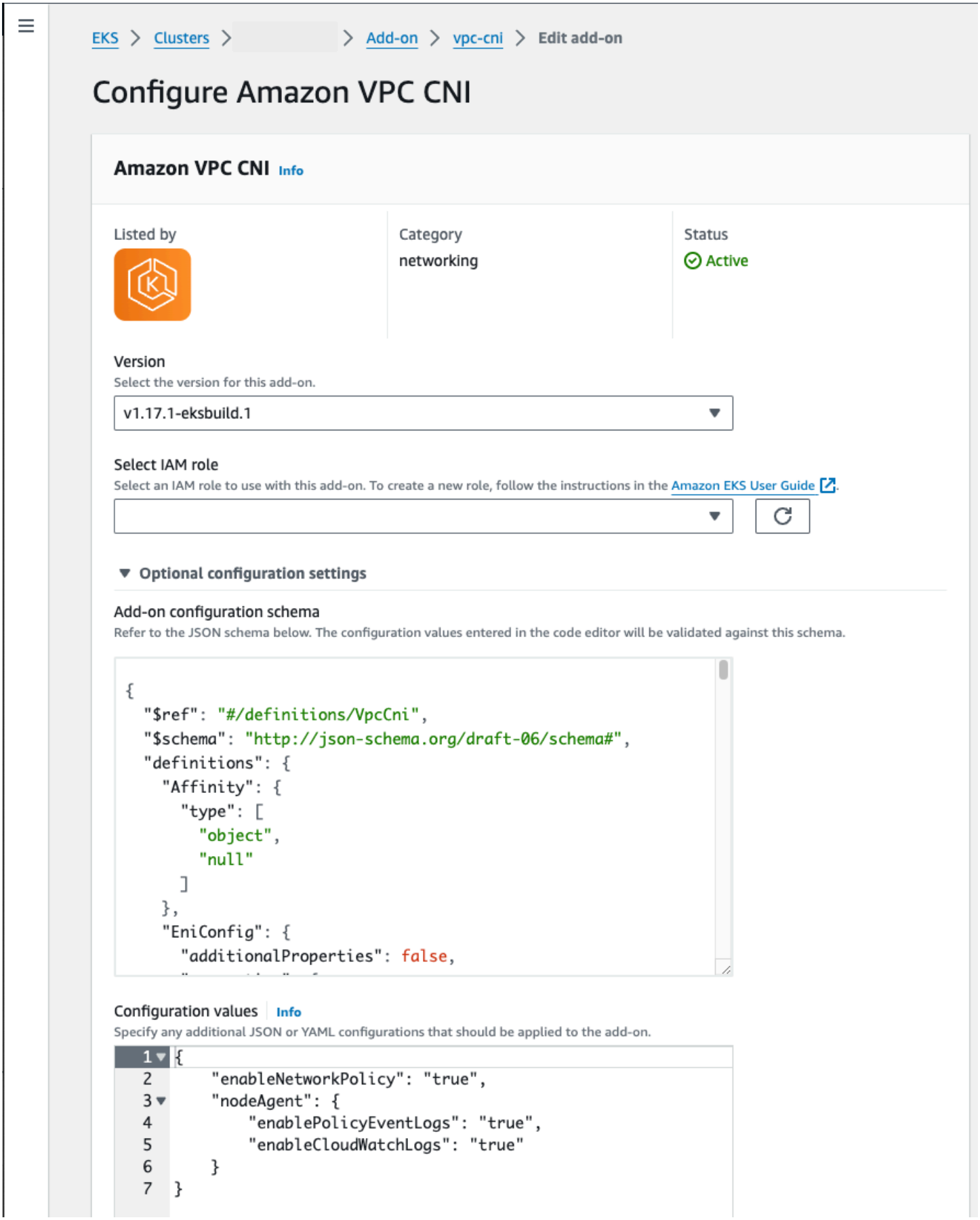
   i. Wählen Sie `v1.14.0-eksbuild.3` oder eine neuere Version in der Dropdown-Liste aus.

   ii. Erweitern Sie Optionale Konfigurationseinstellungen.

   iii. Geben Sie den JSON-Schlüssel der obersten Ebene `"nodeAgent":` ein, und der Wert ist ein Objekt mit dem Schlüssel `"enableCloudWatchLogs":` und dem Wert `"true"` in Konfigurationswerte. Der resultierende Text muss ein gültiges JSON-Objekt sein. Das folgende Beispiel zeigt, dass Netzwerkrichtlinien und Netzwerkrichtlinien-Protokolle aktiviert sind und die Protokolle an CloudWatch Logs gesendet werden:

```
{
    "enableNetworkPolicy": "true",
    "nodeAgent": {
        "enablePolicyEventLogs": "true",
        "enableCloudWatchLogs": "true",
    }
}
```

Der folgende Screenshot zeigt ein Beispiel für dieses Szenario.


EKS > Clusters > > Add-on > vpc-cni > Edit add-on
Configure Amazon VPC CNI
Amazon VPC CNI Info
Listed by
Category
networking
Status
Active
Version
Select the version for this add-on.
v1.17.1-eksbuild.1
Select IAM role
Select an IAM role to use with this add-on. To create a new role, follow the instructions in the Amazon EKS User Guide.
Optional configuration settings
Add-on configuration schema
Refer to the JSON schema below. The configuration values entered in the code editor will be validated against this schema.
{
"$ref": "#/definitions/VpcCni",
"$schema": "http://json-schema.org/draft-06/schema#",
"definitions": {
"Affinity": {
"type": [
"object",
"null"
]
},
"EniConfig": {
"additionalProperties": false,
Configuration values Info
Specify any additional JSON or YAML configurations that should be applied to the add-on.
1 {
2 "enableNetworkPolicy": "true",
3 "nodeAgent": {
4 "enablePolicyEventLogs": "true",
5 "enableCloudWatchLogs": "true"
6 }
7 }

AWS -CLI

a. Führen Sie den folgenden AWS CLI-Befehl aus. Ersetzen Sie `my-cluster` durch den Namen Ihres Clusters und den IAM-Rollen-ARN durch die Rolle, die Sie verwenden.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version v1.14.0-eksbuild.3 \
    --service-account-role-arn arn:aws: iam::123456789012:role/AmazonEKSVPCCNIRole
 \
    --resolve-conflicts PRESERVE --configuration-values '{"nodeAgent":
 {"enablePolicyEventLogs": "true", "enableCloudWatchLogs": "true"}}'
```

Selbstverwaltetes Add-On

Helm

Wenn Sie das Amazon VPC CNI-Plugin für Kubernetes über installiert haben, können Sie die Konfiguration aktualisieren`helm`, um Netzwerkrichtlinienprotokolle an Logs zu senden. CloudWatch

a. Führen Sie den folgenden Befehl aus, um Netzwerkrichtlinien-Protokolle zu aktivieren und sie an Logs zu senden. CloudWatch

```
helm upgrade --set nodeAgent.enablePolicyEventLogs=true --set
 nodeAgent.enableCloudWatchLogs=true aws-vpc-cni --namespace kube-system eks/aws-
vpc-cni
```

kubectl

a. Öffnen Sie das `DaemonSet` `aws-node` in Ihrem Editor.

```
kubectl edit daemonset -n kube-system aws-node
```

b. Ersetzen Sie das `false` durch `true` zwei Befehlsargumenten `--enable-policy-event-logs=false` im `--enable-cloudwatch-logs=false` im `args:` im `aws-network-policy-agent`-Container im VPC CNI `aws-node` `DaemonSet`-Manifest.

```
    - args:
       - --enable-policy-event-logs=true
       - --enable-cloudwatch-logs=true
```

Netzwerkrichtlinien-Protokolle mit einem Fluent Bit-**`DaemonSet`** senden

Wenn Sie Fluent Bit in einem `DaemonSet` zum Senden von Protokollen von Ihren Knoten verwenden, können Sie eine Konfiguration hinzufügen, um die Netzwerkrichtlinien-Protokolle aus den Netzwerkrichtlinien einzubeziehen. Sie können die folgende Beispielkonfiguration verwenden:

```
    [INPUT]
        Name              tail
        Tag               eksnp.*
        Path              /var/log/aws-routed-eni/network-policy-agent*.log
        Parser            json
        DB                /var/log/aws-routed-eni/flb_npagent.db
        Mem_Buf_Limit     5MB
        Skip_Long_Lines   On
        Refresh_Interval  10
```

Enthaltenes eBPF-SDK

Das Amazon-VPC-CNI-Plugin für Kubernetes installiert die eBPF-SDK-Tool-Sammlung auf den Knoten. Sie können die eBPF-SDK-Tools verwenden, um Probleme mit Netzwerkrichtlinien zu identifizieren. Der folgende Befehl listet zum Beispiel die Programme auf, die auf dem Knoten ausgeführt werden.

```
sudo /opt/cni/bin/aws-eks-na-cli ebpf progs
```

Zum Ausführen dieses Befehls können Sie eine beliebige Methode verwenden, um eine Verbindung mit dem Knoten herzustellen.

Bekannte Probleme und Lösungen

In den folgenden Abschnitten werden bekannte Probleme mit dem Netzwerkrichtlinien-Feature von Amazon VPC CNI und deren Lösungen beschrieben.

Netzwerkrichtlinien-Protokolle wurden generiert, obwohl sie auf enable-policy-event-logs „false“ gesetzt sind

Problem: EKS VPC CNI generiert Netzwerkrichtlinien-Protokolle, auch wenn die `enable-policy-event-logs`-Einstellung auf `false` gesetzt ist.

Lösung: Diese `enable-policy-event-logs`-Einstellung deaktiviert nur die Protokolle zur „Entscheidung“ über Richtlinien, jedoch nicht die gesamte Protokollierung des Netzwerkrichtlinien-

Agenten. Dieses Verhalten ist in der aws-network-policy-agent README-Datei unter GitHub dokumentiert. Um die Protokollierung vollständig zu deaktivieren, müssen Sie möglicherweise andere Protokollierungskonfigurationen anpassen.

Probleme bei der Bereinigung der Netzwerk-Richtlinienzuordnung

Problem: Probleme mit dem Netzwerk `policyendpoint` bestehen weiterhin und werden nicht bereinigt, nachdem Pods gelöscht wurden.

Lösung: Dieses Problem wurde durch ein Problem mit dem VPC CNI-Add-on Version 1.19.3-eksbuild.1 verursacht. Aktualisieren Sie auf eine neuere Version des VPC-CNI-Add-Ons, um dieses Problem zu beheben.

Netzwerkrichtlinien werden nicht angewendet

Problem: Das Netzwerkrichtlinien-Feature ist im Amazon-VPC-CNI-Plugin aktiviert, aber die Netzwerkrichtlinien werden nicht korrekt angewendet.

Wenn Sie eine Netzwerkrichtlinie `kind: NetworkPolicy` erstellen und diese keine Auswirkungen auf den Pod hat, überprüfen Sie, ob das Policyendpoint-Objekt im selben Namespace wie der Pod erstellt wurde. Wenn in den Namespaces keine `policyendpoint`-Objekte vorhanden sind, konnte der Netzwerkrichtlinien-Controller (Teil des EKS-Clusters) keine Netzwerkrichtlinien-Regeln für den Netzwerkrichtlinien-Agenten (Teil des VPC CNI) erstellen.

Lösung: Die Lösung besteht darin, die Berechtigungen des VPC CNI (`ClusterRole` : `aws-node`) und des Netzwerkrichtlinien-Controllers (`ClusterRole` : `eks:network-policy-controller`) anzupassen und diese Aktionen in jedem Tool zur Durchsetzung von Richtlinien, wie beispielsweise Kyverno, zuzulassen. Stellen Sie sicher, dass Kyverno-Richtlinien die Erstellung von `policyendpoint`-Objekten nicht blockieren. Informationen zu den erforderlichen Berechtigungen in the section called “Neue `policyendpoints` CRD und Berechtigungen” finden Sie im vorherigen Abschnitt.

Pods kehren nach dem Löschen der Richtlinie im strikten Modus nicht in den Standard-Verweigerungsstatus zurück

Problem: Wenn Netzwerkrichtlinien im strikten Modus aktiviert sind, beginnen Pods mit einer Standard-Verweigerungsrichtlinie. Nachdem die Richtlinien angewendet wurden, wird der Datenverkehr zu den angegebenen Endpunkten zugelassen. Beim Löschen der Richtlinien kehrt der Pod jedoch nicht in den Standard-Verweigerungsstatus zurück, sondern in den Standard-Zulassungsstatus.

Problem: Dieses Problem wurde in der VPC CNI-Version 1.19.3 behoben, welche die Version 1.2.0 des Netzwerkrichtlinien-Agenten enthielt. Nach der Korrektur und bei aktiviertem strengen Modus kehrt der Pod nach dem Entfernen der Richtlinien wie erwartet in den Standard-Verweigerungsstatus zurück.

Sicherheitsgruppen für Startup-Verzögerung von Pods

Problem: Bei Verwendung des Features „Sicherheitsgruppen für Pods“ in EKS kommt es zu einer erhöhten Startup-Verzögerung der Pods.

Lösung: Die Latenz ist auf die Ratenbegrenzung im Resource Controller durch die API-Drosselung auf der `CreateNetworkInterface` API zurückzuführen, die der VPC-Ressourcencontroller verwendet, um Verzweigungen ENIs für die Pods zu erstellen. Überprüfen Sie die API-Limits Ihres Kontos für diesen Vorgang und beantragen Sie gegebenenfalls eine Limit-Erhöhung.

FailedScheduling aufgrund unzureichender vpc.amazonaws.com/pod-eni

Problem: Die Planung von Pods schlägt fehl mit folgendem Fehler: `FailedScheduling 2m53s (x28 over 137m) default-scheduler 0/5 nodes are available: 5 Insufficient vpc.amazonaws.com/pod-eni. preemption: 0/5 nodes are available: 5 No preemption victims found for incoming pod.`

Lösung: Wie beim vorherigen Problem erhöht die Zuweisung von Sicherheitsgruppen zu Pods die Latenz bei der Pod-Planung und kann über den CNI-Schwellenwert für die Zeit zum Hinzufügen jeder ENI hinausgehen, was zu Fehlern beim Starten von Pods führen kann. Dies ist das erwartete Verhalten bei der Verwendung von Sicherheitsgruppen für Pods. Berücksichtigen Sie beim Entwerfen Ihrer Workload-Architektur die Auswirkungen auf die Planung.

IPAM-Konnektivitätsprobleme und Segmentierungsfehler

Problem: Es treten mehrere Fehler auf, darunter Probleme mit der IPAM-Konnektivität, Drosselungsanfragen und Segmentierungsfehler:

- `Checking for IPAM connectivity …`
- `Throttling request took 1.047064274s`
- `Retrying waiting for IPAM-D`
- `panic: runtime error: invalid memory address or nil pointer dereference`

Lösung: Dieses Problem tritt auf, wenn Sie `systemd-udev` auf AL2 023 installieren, da die Datei neu geschrieben wird und eine Richtlinie verletzt wird. Dies kann bei der Aktualisierung auf ein anderes

`releasever` mit einem aktualisierten Paket oder bei der manuellen Aktualisierung des Pakets selbst auftreten. Vermeiden Sie die Installation oder Aktualisierung `systemd-udev` auf AL2 023-Knoten.

Fehler beim Auffinden des Geräts anhand des Namens

Problem: Fehlermeldung: `{"level":"error","ts":"2025-02-05T20:27:18.669Z","caller":"ebpf/bpf_client.go:578","msg":"failed to find device by name eni9ea69618bf0: %!w(netlink.LinkNotFoundError={0xc000115310})"}`

Lösung: Dieses Problem wurde in den neuesten Versionen des Amazon-VPC-CNI-Netzwerkrichtlinien-Agenten (v1.2.0) erkannt und behoben. Aktualisieren Sie auf die neueste Version von VPC CNI, um dieses Problem zu beheben.

CVE-Schwachstellen im Multus-CNI-Image

Problem: Der erweiterte EKS ImageScan CVE-Bericht identifiziert Sicherheitslücken in der Multus CNI-Image-Version v4.1.4-eksbuild.2_thick.

Lösung: Aktualisieren Sie auf die neue Version des Multus-CNI-Images und das neue Netzwerkrichtlinien-Controller-Image, die keine Schwachstellen aufweisen. Der Scanner kann aktualisiert werden, um die in der vorherigen Version gefundenen Schwachstellen zu beheben.

Ablaufinformationen DENY-Urteile in Protokollen

Probleme: Netzwerkrichtlinien-Protokolle zeigen DENY-Urteile an: `{"level":"info","ts":"2024-11-25T13:34:24.808Z","logger":"ebpf-client","caller":"events/events.go:193","msg":"Flow Info: ","Src IP":"","Src Port":9096,"Dest IP":"","Dest Port":56830,"Proto":"TCP","Verdict":"DENY"}`

Lösung: Dieses Problem wurde in der neuen Version des Netzwerkrichtlinien-Controllers behoben. Aktualisieren Sie auf die neueste EKS-Plattformversion, um Protokollierungsprobleme zu beheben.

Pod-to-pod Kommunikationsprobleme nach der Migration von Calico

Problem: Nach dem Upgrade eines EKS-Clusters auf Version 1.30 und dem Wechsel von Calico zu Amazon VPC CNI für Netzwerkrichtlinien schlägt die pod-to-pod Kommunikation fehl, wenn Netzwerkrichtlinien angewendet werden. Die Kommunikation wird wiederhergestellt, wenn die Netzwerkrichtlinien gelöscht werden.

Lösung: Der Netzwerkrichtlinien-Agent im VPC CNI kann nicht so viele Ports wie Calico spezifizieren. Verwenden Sie stattdessen Port-Bereiche in den Netzwerkrichtlinien. Die Höchstzahl der eindeutigen Kombinationen von Ports für jedes Protokoll in jedem `ingress:`- oder `egress:`-Selektor in einer Netzwerkrichtlinie beträgt 24. Verwenden Sie Port-Bereiche, um die Anzahl der eindeutigen Ports zu reduzieren und diese Einschränkung zu vermeiden.

Netzwerkrichtlinien-Agent unterstützt keine eigenständigen Pods

Problem: Netzwerkrichtlinien, die auf eigenständige Pods angewendet werden, können zu inkonsistentem Verhalten führen.

Lösung: Der Netzwerkrichtlinien-Agent unterstützt derzeit nur Pods, die als Teil einer Bereitstellung/eines Replikatsatzes bereitgestellt werden. Wenn Netzwerkrichtlinien auf eigenständige Pods angewendet werden, kann es zu Unstimmigkeiten im Verhalten kommen. Dies ist oben auf dieser Seite, in und in der Ausgabe the section called “Überlegungen” #327 am dokumentiert. aws-network-policy-agent GitHub GitHub Stellen Sie Pods als Teil einer Bereitstellung oder eines Replikatsatzes bereit, um ein konsistentes Verhalten der Netzwerkrichtlinien zu gewährleisten.

Stars-Demo der Netzwerkrichtlinie für Amazon EKS

Diese Demo erstellt einen Front-End-, Back-End- und Client-Service in Ihrem Amazon-EKS-Cluster. Außerdem wird in der Demo eine grafische Benutzeroberfläche für die Verwaltung erstellt, in der die verfügbaren Wege für ein- und ausgehenden Datenverkehr zwischen den einzelnen Services dargestellt werden. Wir empfehlen, die Demo in einem Cluster durchzuführen, in dem keine Produktions-Workloads ausgeführt werden.

Bevor Sie Netzwerkrichtlinien erstellen, können alle Services bidirektional kommunizieren. Nachdem Sie die Netzwerkrichtlinien angewendet haben, können Sie feststellen, dass der Client nur mit dem Front-End-Service kommuniziert und das Back-End nur Datenverkehr vom Front-End akzeptiert.

1. Wenden Sie die Services für Front-End, Back-End, Client und Verwaltungs-Benutzeroberfläche an:

```
kubectl apply -f https://raw.githubusercontent.com/aws-samples/eks-
workshop/2f9d29ed3f82ed6b083649e975a0e574fb8a4058/content/beginner/120_network-
policies/calico/stars_policy_demo/create_resources.files/namespace.yaml
kubectl apply -f https://raw.githubusercontent.com/aws-samples/eks-
workshop/2f9d29ed3f82ed6b083649e975a0e574fb8a4058/content/beginner/120_network-
policies/calico/stars_policy_demo/create_resources.files/management-ui.yaml
kubectl apply -f https://raw.githubusercontent.com/aws-samples/eks-
workshop/2f9d29ed3f82ed6b083649e975a0e574fb8a4058/content/beginner/120_network-
policies/calico/stars_policy_demo/create_resources.files/backend.yaml
```

```
kubectl apply -f https://raw.githubusercontent.com/aws-samples/eks-
workshop/2f9d29ed3f82ed6b083649e975a0e574fb8a4058/content/beginner/120_network-
policies/calico/stars_policy_demo/create_resources.files/frontend.yaml
kubectl apply -f https://raw.githubusercontent.com/aws-samples/eks-
workshop/2f9d29ed3f82ed6b083649e975a0e574fb8a4058/content/beginner/120_network-
policies/calico/stars_policy_demo/create_resources.files/client.yaml
```

2. Zeigen Sie alle Pods im Cluster an.

```
kubectl get pods -A
```

   Eine Beispielausgabe sieht wie folgt aus.

   In Ihrer Ausgabe sollten Pods in den Namespaces aufgeführt sein, die in der folgenden Ausgabe angezeigt werden. Die *NAMEN* Ihrer Pods und die Anzahl der Pods in der Spalte `READY` unterscheiden sich von den Angaben in der folgenden Ausgabe. Fahren Sie erst dann fort, wenn Pods mit ähnlichen Namen angezeigt werden und bei allen `Running` in der Spalte `STATUS` aufgeführt wird.

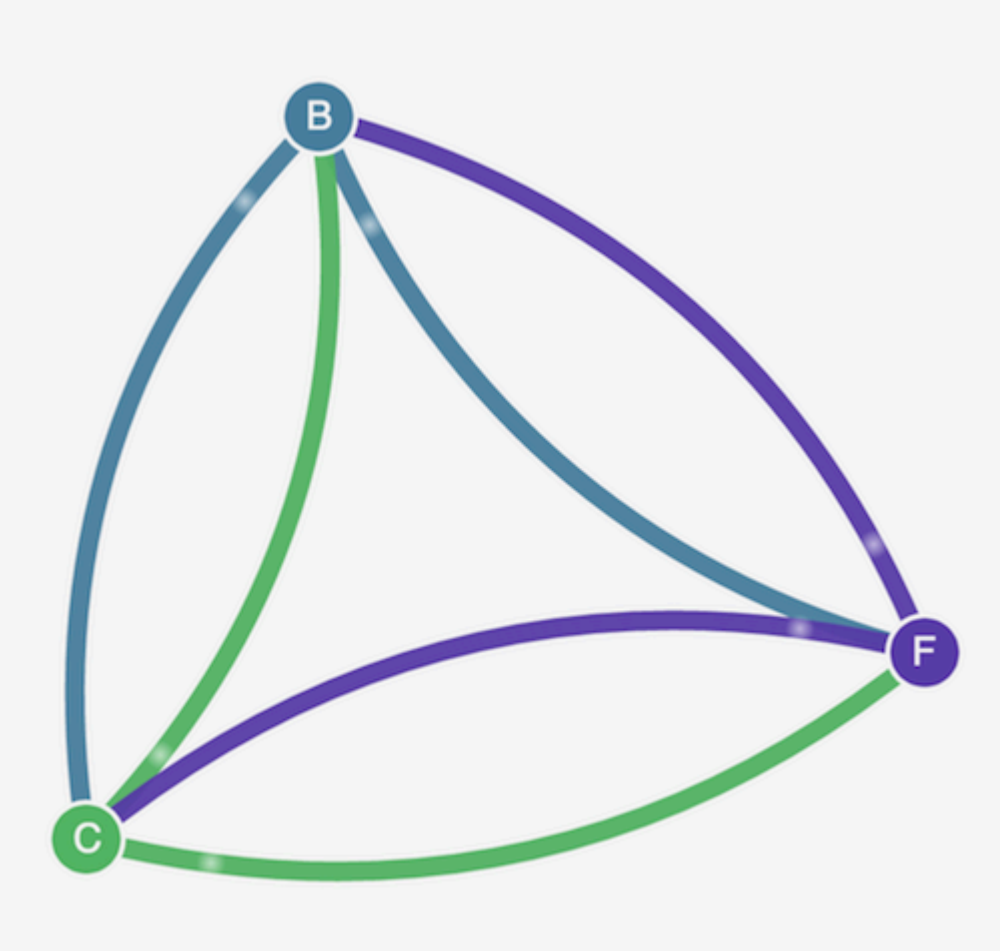
```
NAMESPACE         NAME                                  READY   STATUS
 RESTARTS   AGE
[...]
client            client-xlffc                          1/1     Running   0
     5m19s
[...]
management-ui     management-ui-qrb2g                   1/1     Running   0
     5m24s
stars             backend-sz87q                         1/1     Running   0
     5m23s
stars             frontend-cscnf                        1/1     Running   0
     5m21s
[...]
```

3. Zum Herstellen einer Verbindung mit der Verwaltungsbenutzeroberfläche stellen Sie eine Verbindung mit der `EXTERNAL-IP` des Services her, der auf Ihrem Cluster ausgeführt wird.

```
kubectl get service/management-ui -n management-ui
```

4. Öffnen Sie den Pfad aus dem vorherigen Schritt in einem Browser. Die Verwaltungs-Benutzeroberfläche sollte angezeigt werden. Der C--Knoten ist der Client-Service, der F-Knoten ist der Front-End-Service und der B--Knoten ist der Back-End-Service. Jeder Knoten verfügt

über vollständigen Kommunikationszugriff auf alle anderen Knoten, wie durch die fett markierten, farbigen Zeilen angegeben.



5. Wenden Sie die folgende Netzwerkrichtlinie sowohl in den `stars`- als auch in den `client`-Namespaces an, um die Services voneinander zu isolieren:

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: default-deny
spec:
  podSelector:
    matchLabels: {}
```

Sie können die folgenden Befehle verwenden, um die Richtlinie auf beide Namespaces anzuwenden:

```
kubectl apply -n stars -f https://raw.githubusercontent.com/aws-samples/eks-
workshop/2f9d29ed3f82ed6b083649e975a0e574fb8a4058/content/beginner/120_network-
policies/calico/stars_policy_demo/apply_network_policies.files/default-deny.yaml
kubectl apply -n client -f https://raw.githubusercontent.com/aws-samples/eks-
workshop/2f9d29ed3f82ed6b083649e975a0e574fb8a4058/content/beginner/120_network-
policies/calico/stars_policy_demo/apply_network_policies.files/default-deny.yaml
```

6. Aktualisieren Sie Ihren Browser. Wie Sie feststellen können, kann die Benutzeroberfläche der Verwaltung keinen der Knoten mehr erreichen, sodass diese nicht in der Benutzeroberfläche angezeigt werden.
7. Wenden Sie die folgenden verschiedenen Netzwerkrichtlinien an, um der Verwaltungsbenutzeroberfläche den Zugriff auf die Services zu erlauben: Wenden Sie diese Richtlinie an, um die Benutzeroberfläche zuzulassen:

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: allow-ui
spec:
  podSelector:
    matchLabels: {}
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: management-ui
```

Wenden Sie diese Richtlinie an, um den Client zuzulassen:

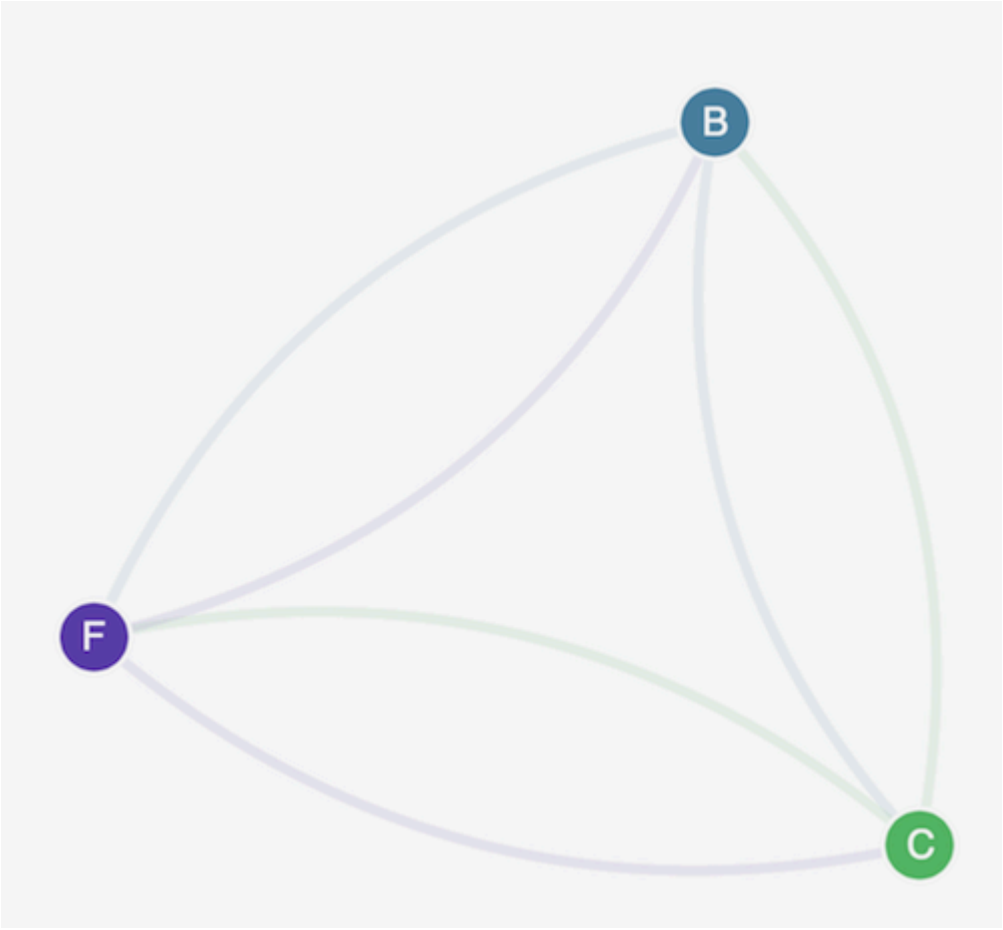
```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: client
  name: allow-ui
spec:
  podSelector:
    matchLabels: {}
  ingress:
    - from:
```

```
        - namespaceSelector:
            matchLabels:
              role: management-ui
```

Sie können die folgenden Befehle verwenden, um beide Richtlinien anzuwenden:

```
kubectl apply -f https://raw.githubusercontent.com/aws-samples/eks-
workshop/2f9d29ed3f82ed6b083649e975a0e574fb8a4058/content/beginner/120_network-
policies/calico/stars_policy_demo/apply_network_policies.files/allow-ui.yaml
kubectl apply -f https://raw.githubusercontent.com/aws-samples/eks-
workshop/2f9d29ed3f82ed6b083649e975a0e574fb8a4058/content/beginner/120_network-
policies/calico/stars_policy_demo/apply_network_policies.files/allow-ui-client.yaml
```

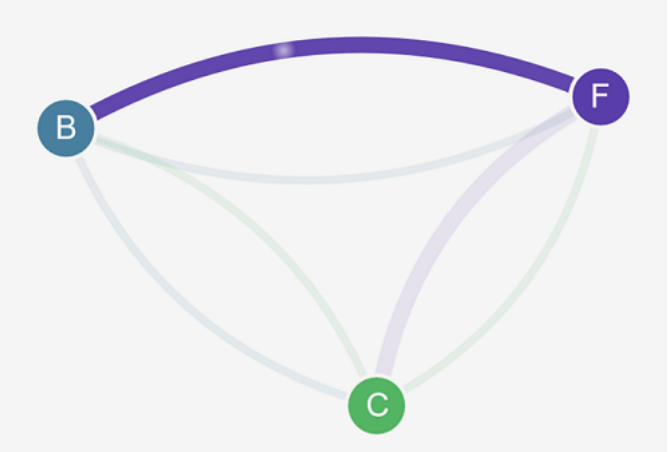
8. Aktualisieren Sie Ihren Browser. Wie Sie feststellen, können die Knoten nun wieder über die Verwaltungs-Benutzeroberfläche erreicht werden, sie können jedoch nicht miteinander kommunizieren.



9. Wenden Sie die folgende Netzwerkrichtlinie an, um Verkehr vom Front-End-Service zum Back-End-Service zuzulassen:

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: backend-policy
spec:
  podSelector:
    matchLabels:
      role: backend
  ingress:
    - from:
        - podSelector:
            matchLabels:
              role: frontend
      ports:
        - protocol: TCP
          port: 6379
```

10. Aktualisieren Sie Ihren Browser. Wie Sie feststellen, kann das Front-End mit dem Back-End kommunizieren.

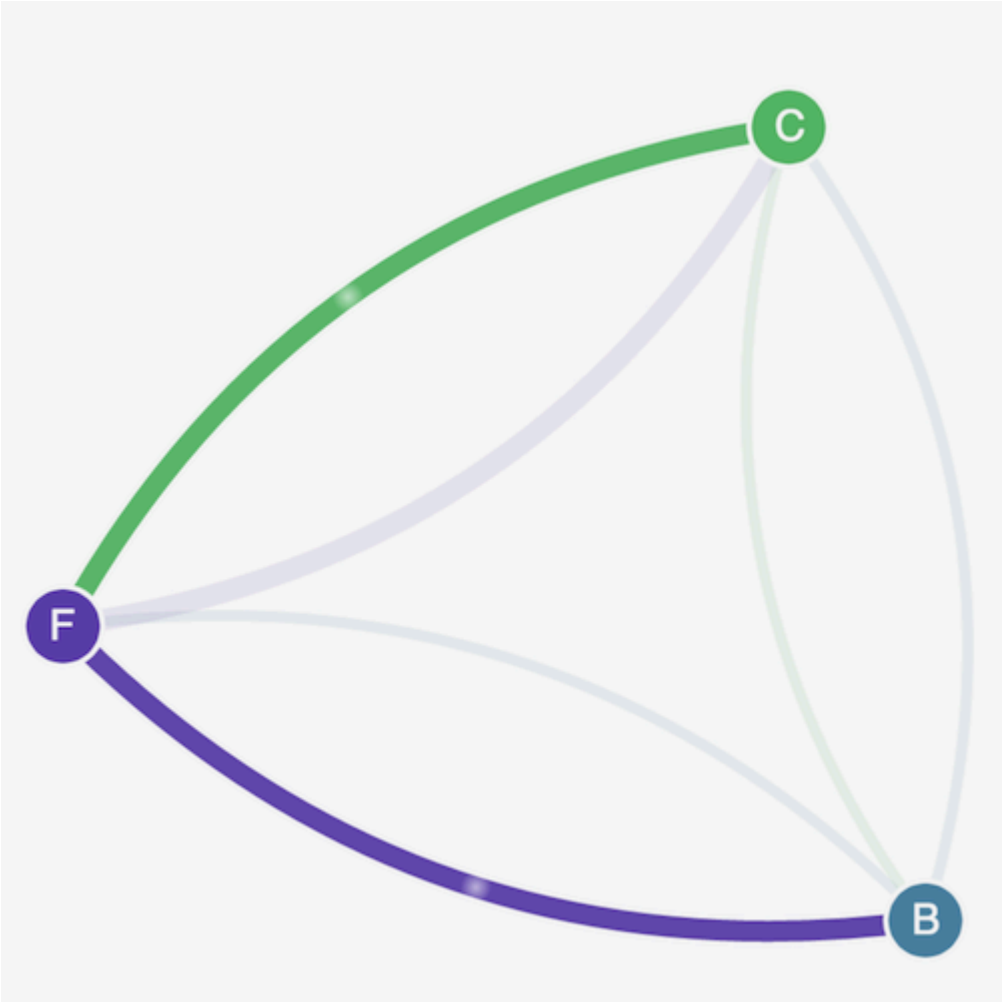


11. Wenden Sie die folgende Netzwerkrichtlinie an, um Datenverkehr vom Client zum Front-End-Service zuzulassen:

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: frontend-policy
spec:
  podSelector:
```

```
    matchLabels:
      role: frontend
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: client
      ports:
        - protocol: TCP
          port: 80
```

12. Aktualisieren Sie Ihren Browser. Wie Sie feststellen, kann der Client mit dem Front-End-Service kommunizieren. Der Front-End-Service kann nach wie vor mit dem Back-End-Service kommunizieren.



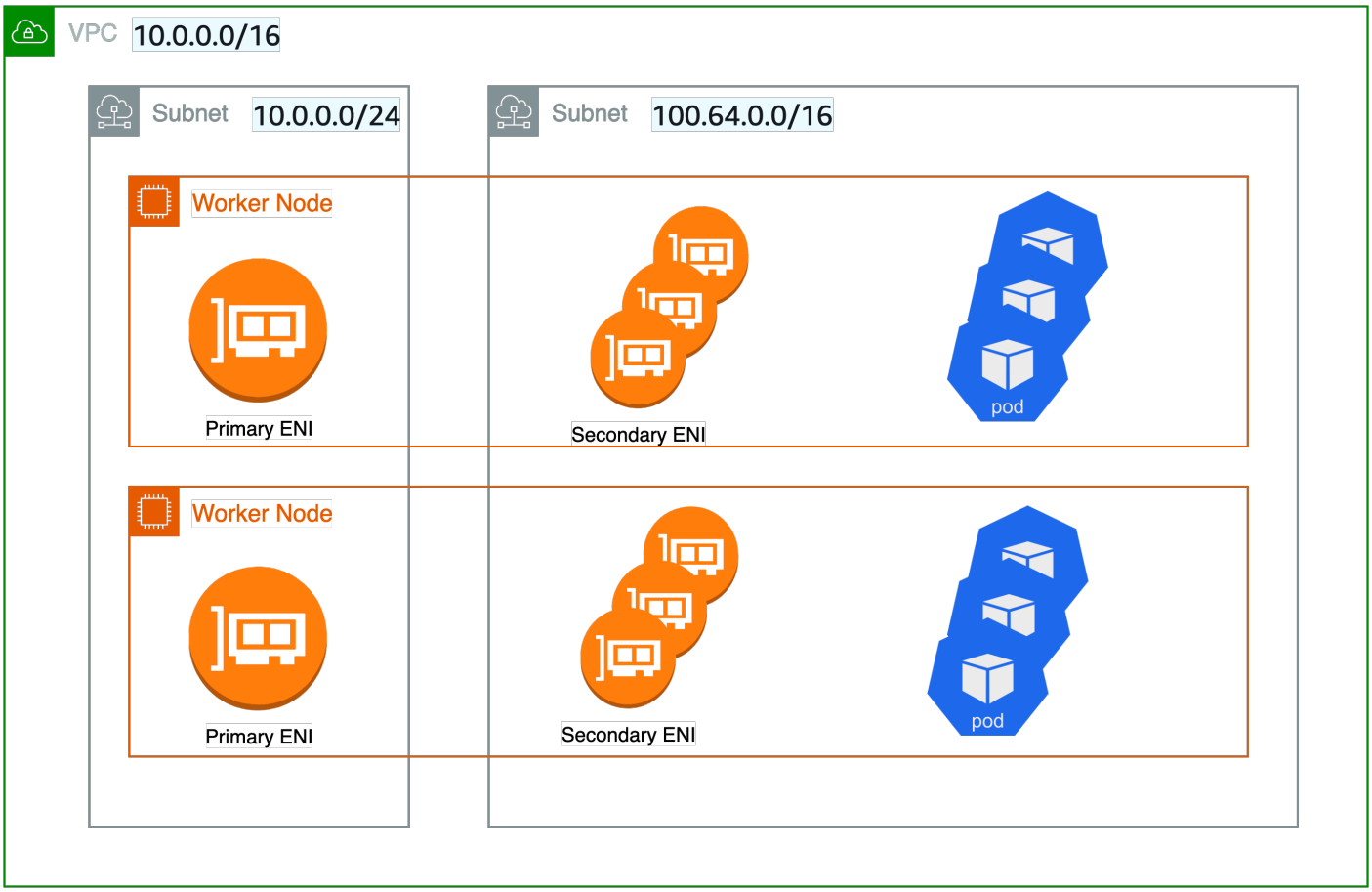
13. (Optional) Wenn Sie die Demo abgeschlossen haben, können Sie die zugehörigen Ressourcen löschen.

```
kubectl delete -f https://raw.githubusercontent.com/aws-samples/eks-
workshop/2f9d29ed3f82ed6b083649e975a0e574fb8a4058/content/beginner/120_network-
policies/calico/stars_policy_demo/create_resources.files/client.yaml
kubectl delete -f https://raw.githubusercontent.com/aws-samples/eks-
workshop/2f9d29ed3f82ed6b083649e975a0e574fb8a4058/content/beginner/120_network-
policies/calico/stars_policy_demo/create_resources.files/frontend.yaml
kubectl delete -f https://raw.githubusercontent.com/aws-samples/eks-
workshop/2f9d29ed3f82ed6b083649e975a0e574fb8a4058/content/beginner/120_network-
policies/calico/stars_policy_demo/create_resources.files/backend.yaml
kubectl delete -f https://raw.githubusercontent.com/aws-samples/eks-
workshop/2f9d29ed3f82ed6b083649e975a0e574fb8a4058/content/beginner/120_network-
policies/calico/stars_policy_demo/create_resources.files/management-ui.yaml
kubectl delete -f https://raw.githubusercontent.com/aws-samples/eks-
workshop/2f9d29ed3f82ed6b083649e975a0e574fb8a4058/content/beginner/120_network-
policies/calico/stars_policy_demo/create_resources.files/namespace.yaml
```

Auch nach dem Löschen der Ressourcen können noch Netzwerkrichtlinien-Endpunkte auf den Knoten vorhanden sein, die die Vernetzung in Ihrem Cluster auf unerwartete Weise stören können. Die einzige sichere Möglichkeit, diese Regeln zu entfernen, besteht darin, die Knoten neu zu starten oder alle Knoten zu beenden und sie zu recyceln. Um alle Knoten zu beenden, legen Sie entweder die Anzahl der gewünschten Auto Scaling-Gruppen auf 0 und anschließend wieder auf die gewünschte Zahl fest oder beenden einfach die Knoten.

Bereitstellung von Pods in alternativen Subnetzen mit benutzerdefiniertem Netzwerk

Gilt für: Linux-IPv4-Fargate-Knoten, Linux-Knoten mit Amazon-EC2-Instances



Standardmäßig erstellt das Amazon-VPC-CNI-Plugin für Kubernetes sekundäre Elastic-Network-Schnittstellen (Netzwerkschnittstellen) für Ihren Amazon-EC2-Knoten im gleichen Subnetz wie die primäre Netzwerkschnittstelle des Knotens. Es verknüpft auch dieselben Sicherheitsgruppen mit der sekundären Netzwerkschnittstelle, die mit der primären Netzwerkschnittstelle verknüpft sind. Aus einem oder mehreren der folgenden Gründe möchten Sie möglicherweise, dass das Plugin sekundäre Netzwerkschnittstellen in einem anderen Subnetz erstellt oder Sie möchten den sekundären Netzwerkschnittstellen oder beiden verschiedene Sicherheitsgruppen zuordnen:

- Es gibt eine begrenzte Anzahl von `IPv4`-Adressen, die in dem Subnetz verfügbar sind, in dem sich die primäre Netzwerkschnittstelle befindet. Dies könnte die Anzahl der Pods einschränken, die Sie in dem Subnetz erstellen können. Durch Verwendung eines anderen Subnetzes für sekundäre Netzwerkschnittstellen können Sie die Anzahl der verfügbaren `IPv4`-Adressen für Pods erhöhen.
- Aus Sicherheitsgründen müssen Ihre Pods unter Umständen andere Sicherheitsgruppen oder Subnetze verwenden als die primäre Netzwerkschnittstelle des Knotens.
- Die Knoten sind in öffentlichen Subnetzen konfiguriert und die Pods sollen in privaten Subnetzen platziert werden. Die einem öffentlichen Subnetz zugeordnete Routing-Tabelle enthält eine Route

zu einem Internet-Gateway. Die einem privaten Subnetz zugeordnete Routing-Tabelle enthält keine Route zu einem Internet-Gateway.

> **Tip**
>
> Sie können Ihrem Amazon-EKS-Cluster auch direkt ein neues oder vorhandenes Subnetz hinzufügen, ohne benutzerdefinierte Netzwerke zu verwenden. Weitere Informationen finden Sie unter the section called “Hinzufügen eines vorhandenen VPC-Subnetzes zu einem Amazon-EKS-Cluster über die Management-Konsole”.

Überlegungen

Nachfolgend finden Sie einige Überlegungen zur Verwendung dieses Features.

- Wenn benutzerdefinierte Netzwerke aktiviert sind, werden Pods keine IP-Adressen zugewiesen, die der primären Netzwerkschnittstelle zugewiesen sind. Pods werden nur IP-Adressen von sekundären Netzwerkschnittstellen zugewiesen.
- Wenn Ihr Cluster die `IPv6`-Familie verwendet, können Sie keine benutzerdefinierten Netzwerke verwenden.
- Wenn Sie vorhaben, benutzerdefinierte Netzwerke zu verwenden, um die `IPv4`-Adresserschöpfung zu verhindern, können Sie stattdessen einen Cluster über die `IPv6`-Familie erstellen. Weitere Informationen finden Sie unter the section called “IPv6”.
- Auch wenn Pods in Subnetzen für sekundäre Netzwerkschnittstellen bereitgestellte andere Subnetz- und Sicherheitsgruppen verwenden können als die primäre Netzwerkschnittstelle des Knotens, müssen sich die Subnetze und Sicherheitsgruppen in derselben VPC befinden wie der Knoten.
- Bei Fargate werden Subnetze über das Fargate-Profil verwaltet. Weitere Informationen finden Sie unter the section called “Profile definieren”.

Anpassung der sekundären Netzwerkschnittstelle in Amazon-EKS-Knoten

Führen Sie die folgenden Schritte aus, bevor Sie mit dem Tutorial beginnen:

- Überlegungen überprüfen

- Vertrautheit mit der Funktionsweise des Amazon-VPC-CNI-Plugins für Kubernetes, das sekundäre Netzwerkschnittstellen erstellt und Pods IP-Adressen zuweist. Weitere Informationen finden Sie unter ENI-Zuweisung auf GitHub.
- Version `2.12.3` oder höher oder Version `1.27.160` oder höher der auf Ihrem Gerät installierten und konfigurierten AWS Befehlszeilenschnittstelle (AWS CLI) oder AWS CloudShell. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Paketmanager wie `yum`apt-get, oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der AWS CLI. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Die AWS CLI-Version, in der installiert ist, AWS CloudShell kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zur Aktualisierung finden Sie im AWS CloudShell Benutzerhandbuch unter AWS CLI in Ihrem Home-Verzeichnis installieren.
- Das `kubectl`-Befehlszeilen-Tool ist auf Ihrem Gerät oder in der AWS CloudShell installiert. Informationen zum Installieren oder Aktualisieren von `kubectl` finden Sie unter the section called “kubectl und eksctl einrichten”.
- Wir empfehlen, dass Sie die Schritte in diesem Thema in einer Bash-Shell ausführen. Wenn Sie keine Bash-Shell verwenden, erfordern einige Skript-Befehle wie Zeilenfortsetzungszeichen und die Art und Weise, wie Variablen gesetzt und verwendet werden, eine Anpassung für Ihre Shell. Darüber hinaus können die Zitier- und Escape-Regeln für Ihre Shell unterschiedlich sein. Weitere Informationen finden Sie unter Verwenden von Anführungszeichen mit Zeichenfolgen in der AWS CLI im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle.

Für dieses Tutorial empfehlen wir die Verwendung der Beispielwerte, außer es gibt eine Annotation, diese zu ersetzen. Sie können alle Beispielwert ersetzen, wenn Sie die Schritte für einen Produktions-Cluster ausführen. Wir empfehlen, alle Schritte auf demselben Terminal auszuführen. Grund dafür ist, dass Variablen während der Schritte festgelegt und verwendet werden und diese in anderen Terminals nicht vorhanden sind.

Die Befehle in diesem Thema werden anhand der Konventionen formatiert, die unter Verwenden der AWS CLI-Beispiele aufgeführt sind. Wenn Sie Befehle über die Befehlszeile für Ressourcen ausführen, die sich in einer anderen AWS Region als der AWS Standardregion befinden, die in dem von Ihnen verwendeten AWS CLI-Profil definiert ist, müssen Sie die Befehle ergänzen `--region us-west-2` und durch Ihre AWS Region `us-west-2` ersetzen.

Wenn Sie benutzerdefinierte Netzwerke in Ihrem Produktions-Cluster bereitstellen möchten, fahren Sie mit the section called “Schritt 2: Konfigurieren Ihrer VPC” fort.

Schritt 1: Erstellen einer Test-VPC und eines Clusters

Mit den folgenden Verfahren können Sie eine Test-VPC und einen Cluster erstellen und benutzerdefinierte Netzwerke für diesen Cluster konfigurieren. Wir empfehlen, den Test-Cluster nicht für Produktions-Workloads zu verwenden, da in diesem Thema verschiedene nicht zugehörige Features nicht behandelt werden, die Sie vielleicht in Ihrem Produktions-Cluster verwenden. Weitere Informationen finden Sie unter the section called “Erstellen eines Clusters”.

1. Führen Sie den folgenden Befehl aus, um die `account_id`-Variable zu definieren.

```
account_id=$(aws sts get-caller-identity --query Account --output text)
```

2. Erstellen Sie eine VPC.

   a. Wenn Sie die Bereitstellung auf einem Testsystem durchführen, erstellen Sie eine VPC mithilfe einer Amazon AWS CloudFormation EKS-Vorlage.

```
aws cloudformation create-stack --stack-name my-eks-custom-networking-vpc \
  --template-url https://s3.us-west-2.amazonaws.com/amazon-eks/
cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml \
  --parameters ParameterKey=VpcBlock,ParameterValue=192.168.0.0/24 \
  ParameterKey=PrivateSubnet01Block,ParameterValue=192.168.0.64/27 \
  ParameterKey=PrivateSubnet02Block,ParameterValue=192.168.0.96/27 \
  ParameterKey=PublicSubnet01Block,ParameterValue=192.168.0.0/27 \
  ParameterKey=PublicSubnet02Block,ParameterValue=192.168.0.32/27
```

   b. Die Erstellung des AWS CloudFormation Stacks dauert einige Minuten. Führen Sie den folgenden Befehl aus, um den Status der Bereitstellung des Stacks zu überprüfen.

```
aws cloudformation describe-stacks --stack-name my-eks-custom-networking-vpc --
query Stacks\[\].StackStatus  --output text
```

   Fahren Sie erst mit dem nächsten Schritt fort, wenn die Ausgabe des Befehls `CREATE_COMPLETE` ist.

   c. Definieren Sie Variablen mit den Werten des privaten Subnetzes, das durch die Vorlage IDs erstellt wurde.

```
subnet_id_1=$(aws cloudformation describe-stack-resources --stack-name my-eks-
custom-networking-vpc \
    --query "StackResources[?
LogicalResourceId=='PrivateSubnet01'].PhysicalResourceId" --output text)
subnet_id_2=$(aws cloudformation describe-stack-resources --stack-name my-eks-
custom-networking-vpc \
    --query "StackResources[?
LogicalResourceId=='PrivateSubnet02'].PhysicalResourceId" --output text)
```

d. Definieren Sie Variablen mit den Availability Zones der im vorherigen Schritt abgerufenen Subnetze.

```
az_1=$(aws ec2 describe-subnets --subnet-ids $subnet_id_1 --query
 'Subnets[*].AvailabilityZone' --output text)
az_2=$(aws ec2 describe-subnets --subnet-ids $subnet_id_2 --query
 'Subnets[*].AvailabilityZone' --output text)
```

3. Erstellen Sie eine Cluster-IAM-Rolle.

a. Führen Sie den folgenden Befehl aus, um eine JSON-Datei für eine IAM-Vertrauensrichtlinie zu erstellen.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

b. Erstellen Sie die Amazon-EKS-Cluster-IAM-Rolle. Falls erforderlich, stellen Sie `eks-cluster-role-trust-policy.json` den Pfad auf Ihrem Computer voran, in den Sie im vorherigen Schritt die Datei geschrieben haben. Der Befehl verknüpft die im vorherigen Schritt erstellte Vertrauensrichtlinie mit der Rolle. Um eine IAM-Rolle zu erstellen, muss dem IAM-Prinzipal, der die Rolle erstellt, die `iam:CreateRole`-Aktion (Berechtigung) zugewiesen werden.

```
aws iam create-role --role-name myCustomNetworkingAmazonEKSClusterRole --assume-
role-policy-document file://"eks-cluster-role-trust-policy.json"
```

c. Fügen Sie der Rolle die von Amazon EKS verwaltete Richtlinie mit dem Namen Amazon EKSCluster Policy hinzu. Um eine IAM-Richtlinie an einen IAM-Prinzipal anzuhängen, muss dem Prinzipal, der die Richtlinie anhängt, eine der folgenden IAM-Aktionen (Berechtigungen) zugewiesen werden: `iam:AttachUserPolicy` oder `iam:AttachRolePolicy`.

```
aws iam attach-role-policy --policy-arn arn:aws: iam::aws:policy/
AmazonEKSClusterPolicy --role-name myCustomNetworkingAmazonEKSClusterRole
```

4. Erstellen Sie einen Amazon-EKS-Cluster und konfigurieren Sie Ihr Gerät für die Kommunikation mit diesem.

a. Erstellen Sie einen Cluster.

```
aws eks create-cluster --name my-custom-networking-cluster \
   --role-arn arn:aws: iam::$account_id:role/
myCustomNetworkingAmazonEKSClusterRole \
   --resources-vpc-config subnetIds="$subnet_id_1","$subnet_id_2"
```

> **Note**
>
> Sie erhalten möglicherweise eine Fehlermeldung, dass eine der Availability Zones in Ihrer Anfrage nicht über genügend Kapazität zum Erstellen eines Amazon-EKS-Clusters verfügt. Wenn dies der Fall ist, enthält die Fehlerausgabe die Availability Zones, die einen neuen Cluster unterstützen können. Versuchen Sie, Ihren Cluster mit mindestens zwei Subnetzen erneut zu erstellen, die sich in den unterstützten Availability Zones für Ihr Konto befinden. Weitere Informationen finden Sie unter the section called “Unzureichende Kapazität”.

b. Die Erstellung des Clusters dauert mehrere Minuten. Führen Sie den folgenden Befehl aus, um den Status der Bereitstellung des Clusters zu überprüfen.

```
aws eks describe-cluster --name my-custom-networking-cluster --query
 cluster.status
```

Fahren Sie erst mit dem nächsten Schritt fort, wenn die Ausgabe des Befehls `"ACTIVE"` ist.

c. Konfigurieren Sie `kubectl` für die Kommunikation mit Ihrem Cluster.

```
aws eks update-kubeconfig --name my-custom-networking-cluster
```

Schritt 2: Konfigurieren Ihrer VPC

Für dieses Tutorial wird die VPC in the section called “Schritt 1: Erstellen einer Test-VPC und eines Clusters” erstellt. Passen Sie für einen Produktions-Cluster die Schritte entsprechend Ihrer VPC an, indem Sie alle Beispielwerte durch eigene Werte ersetzen.

1. Stellen Sie sicher, dass das derzeit installierte Amazon-VPC-CNI-Plugin für Kubernetes die neueste Version ist. Informationen zur Ermittlung der neuesten Version für den Amazon-EKS-Add-On-Typ und zur Aktualisierung Ihrer Version auf diese Version finden Sie unter the section called “Add-On aktualisieren”. Informationen dazu, wie Sie die neueste Version für den selbstverwalteten Add-On-Typ ermitteln und Ihre Version darauf aktualisieren können, finden Sie unter the section called “Amazon VPC CNI”.
2. Rufen Sie die ID Ihrer Cluster-VPC ab und speichern Sie diese zur Verwendung in späteren Schritten in einer Variable.

```
vpc_id=$(aws eks describe-cluster --name my-custom-networking-cluster --query
 "cluster.resourcesVpcConfig.vpcId" --output text)
```

3. Ordnen Sie einen zusätzlichen Classless Inter-Domain Routing (CIDR)-Block der VPC Ihres Clusters zu. Der CIDR-Block kann sich nicht mit vorhandenen zugewiesenen CIDR-Blöcken überschneiden.

   a. Zeigen Sie die aktuellen CIDR-Blöcke an, die Ihrer VPC zugeordnet sind.

```
aws ec2 describe-vpcs --vpc-ids $vpc_id \
    --query 'Vpcs[*].CidrBlockAssociationSet[*].{CIDRBlock: CidrBlock, State:
 CidrBlockState.State}' --out table
```

   Eine Beispielausgabe sieht wie folgt aus.

```
---------------------------------
|          DescribeVpcs          |
+-----------------+--------------+
|    CIDRBlock    |    State     |
+-----------------+--------------+
```

```
|  192.168.0.0/24 |  associated  |
+-----------------+--------------+
```

b. Ordnen Sie Ihrer VPC zusätzliche CIDR-Blöcke zu. Ersetzen Sie den CIDR-Blockwert im folgenden Befehl. Weitere Informationen finden Sie unter Zusätzliche IPv4 CIDR-Blöcke mit Ihrer VPC verknüpfen im Amazon VPC-Benutzerhandbuch.

```
aws ec2 associate-vpc-cidr-block --vpc-id $vpc_id --cidr-block 192.168.1.0/24
```

c. Bestätigen Sie, dass der neue Block zugeordnet ist.

```
aws ec2 describe-vpcs --vpc-ids $vpc_id --query
 'Vpcs[*].CidrBlockAssociationSet[*].{CIDRBlock: CidrBlock, State:
 CidrBlockState.State}' --out table
```

Eine Beispielausgabe sieht wie folgt aus.

```
---------------------------------
|          DescribeVpcs         |
+-----------------+-------------+
|    CIDRBlock    |    State    |
+-----------------+-------------+
|  192.168.0.0/24 |  associated |
|  192.168.1.0/24 |  associated |
+-----------------+-------------+
```

Fahren Sie erst mit dem nächsten Schritt fort, wenn der `State` Ihres neuen CIDR-Blocks `associated` ist.

4. Erstellen Sie so viele Subnetze, wie Sie in jeder Availability Zone verwenden möchten, in denen sich Ihre vorhandenen Subnetze befinden. Geben Sie einen CIDR-Block an, der sich innerhalb des CIDR-Blocks befindet, den Sie in einem vorherigen Schritt Ihrer VPC zugeordnet haben.

a. Erstellen Sie neue Subnetze. Ersetzen Sie die CIDR-Blockwerte im folgenden Befehl. Die Subnetze müssen in einem anderen VPC-CIDR-Block erstellt werden als Ihre vorhandenen Subnetze, jedoch in denselben Availability Zones wie Ihre vorhandenen Subnetze. In diesem Beispiel wird ein Subnetz im neuen CIDR-Block jeder Availability Zone erstellt, in der die aktuellen privaten Subnetze vorhanden sind. Die IDs erstellten Subnetze werden in Variablen gespeichert, um sie in späteren Schritten zu verwenden. Die `Name`-Werte entsprechen den Werten, die den Subnetzen zugewiesen wurden, die in einem vorherigen Schritt mit

der Amazon-EKS-VPC-Vorlage erstellt wurden. Namen sind nicht erforderlich. Sie können verschiedene Namen verwenden.

```
new_subnet_id_1=$(aws ec2 create-subnet --vpc-id $vpc_id --availability-zone $az_1
 --cidr-block 192.168.1.0/27 \
    --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=my-eks-custom-
networking-vpc-PrivateSubnet01},{Key=kubernetes.io/role/internal-elb,Value=1}]' \
    --query Subnet.SubnetId --output text)
new_subnet_id_2=$(aws ec2 create-subnet --vpc-id $vpc_id --availability-zone $az_2
 --cidr-block 192.168.1.32/27 \
    --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=my-eks-custom-
networking-vpc-PrivateSubnet02},{Key=kubernetes.io/role/internal-elb,Value=1}]' \
    --query Subnet.SubnetId --output text)
```

> **Important**
>
> Standardmäßig sind Ihre neuen Subnetze implizit mit der Haupt-Routing-Tabelle Ihrer VPCs verknüpft. Diese Routing-Tabelle ermöglicht die Kommunikation zwischen allen Ressourcen, die in der VPC bereitgestellt werden. Sie erlaubt jedoch keine Kommunikation mit Ressourcen mit IP-Adressen, die sich außerhalb der mit Ihrer VPC verknüpften CIDR-Blöcke befinden. Sie können Ihre eigene Routing-Tabelle Ihren Subnetzen zuordnen, um dieses Verhalten zu ändern. Weitere Informationen finden Sie unter Subnetz-Routing-Tabellen im Amazon-VPC-Benutzerhandbuch.

b. Zeigen Sie die aktuellen Subnetze in Ihrer VPC an.

```
aws ec2 describe-subnets --filters "Name=vpc-id,Values=$vpc_id" \
    --query 'Subnets[*].{SubnetId: SubnetId,AvailabilityZone:
 AvailabilityZone,CidrBlock: CidrBlock}' \
    --output table
```

Eine Beispielausgabe sieht wie folgt aus.

```
-------------------------------------------------------------------
|                          DescribeSubnets                        |
+------------------+--------------------+---------------------------+
| AvailabilityZone |     CidrBlock      |         SubnetId          |
+------------------+--------------------+---------------------------+
|  us-west-2d      |  192.168.0.0/27    |     subnet-example1       |
|  us-west-2a      |  192.168.0.32/27   |     subnet-example2       |
```

```
|  us-west-2a      |  192.168.0.64/27   |     subnet-example3        |
|  us-west-2d      |  192.168.0.96/27   |     subnet-example4        |
|  us-west-2a      |  192.168.1.0/27    |     subnet-example5        |
|  us-west-2d      |  192.168.1.32/27   |     subnet-example6        |
+------------------+--------------------+----------------------------+
```

Sie können sehen, dass die Subnetze im CIDR-Block `192.168.1.0`, den Sie erstellt haben, in denselben Availability Zones liegen wie die Subnetze im CIDR-Block `192.168.0.0`.

Schritt 3: Konfigurieren von Kubernetes-Ressourcen

1. Setzen Sie die `AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG` Umgebungsvariable auf `true` in der `aws-node` DaemonSet.

```
kubectl set env daemonset aws-node -n kube-system
 AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG=true
```

2. Rufen Sie die ID der Cluster-Sicherheitsgruppe ab und speichern Sie diese in einer Variablen zur Verwendung im nächsten Schritt. Amazon EKS erstellt diese Sicherheitsgruppe automatisch, wenn Sie Ihren Cluster erstellen.

```
cluster_security_group_id=$(aws eks describe-cluster --name my-custom-networking-
cluster --query cluster.resourcesVpcConfig.clusterSecurityGroupId --output text)
```

3. Erstellen Sie eine benutzerdefinierte `ENIConfig`-Ressource für jedes Subnetz, in dem Sie Pods bereitstellen möchten.

   a. Erstellen Sie für jede Konfiguration der Netzwerkschnittstelle eine eindeutige Datei.

   Die folgenden Befehle erstellen separate `ENIConfig`-Dateien für die beiden Subnetze, die in einem vorherigen Schritt erstellt wurden. Der Wert für `name` muss eindeutig sein. Der Name entspricht der Availability Zone, in der sich das Subnetz befindet. Die Cluster-Sicherheitsgruppe ist dem `ENIConfig` zugeordnet.

```
cat >$az_1.yaml <<EOF
apiVersion: crd.k8s.amazonaws.com/v1alpha1
kind: ENIConfig
metadata:
  name: $az_1
spec:
  securityGroups:
```

```
    - $cluster_security_group_id
  subnet: $new_subnet_id_1
EOF
```

```
cat >$az_2.yaml <<EOF
apiVersion: crd.k8s.amazonaws.com/v1alpha1
kind: ENIConfig
metadata:
  name: $az_2
spec:
  securityGroups:
    - $cluster_security_group_id
  subnet: $new_subnet_id_2
EOF
```

Für einen Produktions-Cluster können Sie die folgenden Änderungen an den vorherigen Befehlen vornehmen:

- Ersetzen Sie $cluster_security_group_id durch die ID einer vorhandenen Sicherheitsgruppe, die Sie für jeden `ENIConfig` verwenden möchten.
- Wir empfehlen, Ihrer `ENIConfigs` denselben Namen zu geben wie der Availability Zone, in der Sie die `ENIConfig` verwenden, falls möglich. Unter Umständen müssen Sie für Ihre `ENIConfigs` andere Namen verwenden als für die Availability Zones. Wenn Sie beispielsweise mehr als zwei Subnetze in derselben Availability Zone haben und beide mit benutzerdefinierten Netzwerken verwenden möchten, benötigen Sie mehrere `ENIConfigs` für dieselbe Availability Zone. Da jede `ENIConfig` einen eindeutigen Namen erfordert, können Sie nur einer Ihrer `ENIConfigs` den Namen der Availability Zone geben.

  Wenn Ihre `ENIConfig` Namen nicht alle mit Availability Zone-Namen identisch sind, ersetzen Sie $az_1 und $az_2 in den vorherigen Befehlen durch Ihre eigenen Namen und kommentieren Sie Ihre Knoten mit den später in diesem Tutorial. ENIConfig

  > **Note**
  >
  > Wenn Sie keine gültige Sicherheitsgruppe für die Verwendung mit einem Produktionscluster angeben und Folgendes verwenden:

- Version `1.8.0` oder höher des Amazon-VPC-CNI-Plugin für Kubernetes verwenden, dann werden die Sicherheitsgruppen verwendet, die mit der primären Elastic-Network-Schnittstelle des Knotens verknüpft sind.
- eine Version des Amazon-VPC-CNI-Plugins für Kubernetes vor `1.8.0`, dann wird die Standardsicherheitsgruppe für die VPC sekundären Netzwerkschnittstellen zugewiesen.

> **Important**
>
> - `AWS_VPC_K8S_CNI_EXTERNALSNAT=false` ist eine Standardeinstellung in der Konfiguration für das Amazon-VPC-CNI-Plugin für Kubernetes. Wenn Sie die Standardeinstellung verwenden, verwendet Datenverkehr, der für IP-Adressen bestimmt ist, die sich nicht innerhalb eines Ihrer VPC zugeordneten CIDR-Blöcke befinden, die Sicherheitsgruppen und Subnetze der primären Netzwerkschnittstelle Ihres Knotens. Die in Ihrem `ENIConfigs` definierten Subnetze und Sicherheitsgruppen, die zum Erstellen sekundärer Netzwerkschnittstellen verwendet werden, werden für diesen Datenverkehr nicht verwendet. Weitere Informationen zu dieser Einstellung finden Sie unter the section called “Ausgehenden Datenverkehr”.
> - Wenn Sie auch Sicherheitsgruppen für Pods verwenden, wird die in einer `SecurityGroupPolicy` angegebene Sicherheitsgruppe anstelle der Sicherheitsgruppe verwendet, die im `ENIConfigs` angegeben ist. Weitere Informationen finden Sie unter the section called “Sicherheitsgruppen für Pods”.

b. Wenden Sie alle benutzerdefinierten Ressourcendateien, die Sie zuvor erstellt haben, mit den folgenden Befehlen auf Ihren Cluster an:

```
kubectl apply -f $az_1.yaml
kubectl apply -f $az_2.yaml
```

4. Bestätigen Sie, dass Ihre `ENIConfigs` erstellt wurden.

```
kubectl get ENIConfigs
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME         AGE
us-west-2a   117s
```

```
us-west-2d   105s
```

5. Wenn Sie benutzerdefinierte Netzwerke auf einem Produktionscluster aktivieren und Ihrem Namen einen anderen `ENIConfigs` Namen als die Availability Zone geben, für die Sie sie verwenden, fahren Sie mit dem nächsten Schritt fort, um EC2 Amazon-Knoten bereitzustellen.

   Ermöglichen Sie Kubernetes, die Option `ENIConfig` für eine Availability Zone automatisch auf alle neuen EC2 Amazon-Knoten anzuwenden, die in Ihrem Cluster erstellt wurden.

   a. Gehen Sie für den Test-Cluster in diesem Tutorial zum nächsten Schritt.

      Prüfen Sie bei einem Produktionscluster, ob eine Anmerkung mit dem Schlüssel `k8s.amazonaws.com/eniConfig` für die `ENI_CONFIG_ANNOTATION_DEF` Umgebungsvariable in der Containerspezifikation für vorhanden ist. `aws-node` DaemonSet

      ```
      kubectl describe daemonset aws-node -n kube-system | grep
       ENI_CONFIG_ANNOTATION_DEF
      ```

      Wenn eine Ausgabe zurückgegeben wird, ist die Anmerkung vorhanden. Wenn keine Ausgabe zurückgegeben wird, ist die Variable nicht gesetzt. Für einen Produktions-Cluster können Sie entweder diese Einstellung oder die Einstellung im folgenden Schritt verwenden. Wenn Sie diese Einstellung verwenden, überschreibt sie die Einstellung im folgenden Schritt. In diesem Tutorial wird die Einstellung im nächsten Schritt verwendet.

   b. Aktualisieren Sie Ihre `aws-node` DaemonSet , sodass die Option `ENIConfig` für eine Availability Zone automatisch auf alle neuen EC2 Amazon-Knoten angewendet wird, die in Ihrem Cluster erstellt wurden.

      ```
      kubectl set env daemonset aws-node -n kube-system
       ENI_CONFIG_LABEL_DEF=topology.kubernetes.io/zone
      ```

Schritt 4: EC2 Amazon-Knoten bereitstellen

1. Erstellen Sie eine Knoten-IAM-Rolle.

   a. Führen Sie den folgenden Befehl aus, um eine JSON-Datei für eine IAM-Vertrauensrichtlinie zu erstellen.

      ```
      {
        "Version":"2012-10-17",
        "Statement": [
      ```

```
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

b. Erstellen Sie eine IAM-Rolle und speichern Sie den zurückgegebenen Amazon-Ressourcennamen (ARN) in einer Variablen zur Verwendung in einem späteren Schritt.

```
node_role_arn=$(aws iam create-role --role-name myCustomNetworkingNodeRole --
assume-role-policy-document file://"node-role-trust-relationship.json" \
    --query Role.Arn --output text)
```

c. Hängen Sie die drei erforderlichen verwalteten IAM-Richtlinien an die IAM-Rolle an.

```
aws iam attach-role-policy \
  --policy-arn arn:aws: iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name myCustomNetworkingNodeRole
aws iam attach-role-policy \
  --policy-arn arn:aws: iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name myCustomNetworkingNodeRole
aws iam attach-role-policy \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKS_CNI_Policy \
    --role-name myCustomNetworkingNodeRole
```

> **Important**
>
> Der Einfachheit halber wird in diesem Tutorial die Richtlinie AmazonEKS_CNI_Policy dieser Knoten-IAM-Rolle angefügt. In einem Produktions-Cluster empfehlen wir jedoch, die Richtlinie einer separaten IAM-Rolle zuzuordnen, die nur mit dem Amazon-VPC-CNI-Plugin für Kubernetes verwendet wird. Weitere Informationen finden Sie unter the section called “Konfiguration für IRSA”.

2. Erstellen Sie einen der folgenden Typen von Knotengruppen. Informationen zum Bestimmen des Instance-Typs, den Sie bereitstellen möchten, finden Sie unter the section called “Amazon EC2-Instance-Typen”. Füllen Sie für dieses Tutorial die Option Verwaltet, Ohne Startvorlage oder mit Startvorlage ohne angegebene AMI-ID aus. Wenn Sie die Knotengruppe für

Produktions-Workloads verwenden möchten, empfehlen wir Ihnen, sich vor der Bereitstellung der Knotengruppe mit allen Optionen für verwaltete und selbstverwaltete Knotengruppen vertraut zu machen.

- Verwaltet – Stellen Sie Ihre Knotengruppe mit einer der folgenden Optionen bereit:
  - Ohne Startvorlage oder mit Startvorlage ohne angegebene AMI-ID – Führen Sie den folgenden Befehl aus. Verwenden Sie für dieses Tutorial die Beispielwerte: Ersetzen Sie für eine Produktionsknotengruppe alle Beispielwerte durch Ihre eigenen Werte. Der Name der Knotengruppe darf nicht länger als 63 Zeichen sein. Er muss mit einem Buchstaben oder einer Ziffer beginnen, kann danach aber auch Bindestriche und Unterstriche enthalten.

    ```
    aws eks create-nodegroup --cluster-name my-custom-networking-cluster --nodegroup-
    name my-nodegroup \
        --subnets $subnet_id_1 $subnet_id_2 --instance-types t3.medium --node-role
     $node_role_arn
    ```

  - Mit einer Startvorlage mit einer angegebenen AMI-ID

    A. Bestimmen Sie die von Amazon EKS empfohlene maximale Anzahl an Pods für Ihre Knoten. Folgen Sie den Anweisungen in den von Amazon EKS empfohlenen maximalen Pods für jeden EC2 Amazon-Instance-Typ und fügen Sie `--cni-custom-networking-enabled` Schritt 3 in diesem Thema hinzu. Notieren Sie die Ausgabe zur Verwendung im nächsten Schritt.

    B. Geben Sie in Ihrer Startvorlage eine Amazon-EKS-optimierte AMI-ID oder ein benutzerdefiniertes AMI an, das auf dem Amazon-EKS-optimierten AMI basiert. Stellen Sie dann die Knotengruppe mithilfe einer Startvorlage bereit und geben Sie die folgenden Benutzerdaten in der Startvorlage an. Diese Benutzerdaten übergeben Argumente an die `NodeConfig` Spezifikation. Weitere Informationen zu NodeConfig finden Sie in der NodeConfig API-Referenz. Sie können `20` entweder durch den Wert aus dem vorherigen Schritt (empfohlen) oder Ihren eigenen Wert ersetzen.

    ```
    ---
    MIME-Version: 1.0
    Content-Type: multipart/mixed; boundary="BOUNDARY"
    --BOUNDARY
    Content-Type: application/node.eks.aws

    ---
    apiVersion: node.eks.aws/v1alpha1
    kind: NodeConfig
    ```

```
spec:
  cluster:
    name: my-cluster
    ...
    kubelet:
      config:
        maxPods: 20
```

Wenn Sie ein benutzerdefiniertes AMI erstellt haben, das nicht auf dem für Amazon EKS optimierten AMI erstellt wurde, müssen Sie die Konfiguration selbst erstellen.

- Selbstverwaltet

  i. Bestimmen Sie die von Amazon EKS empfohlene maximale Anzahl an Pods für Ihre Knoten. Befolgen Sie die Anweisungen in the section called “Von Amazon EKS empfohlene maximale Pods für jeden Amazon-EC2-Instance-Typ” und fügen Sie `--cni-custom-networking-enabled` zu Schritt 3 in diesem Thema hinzu. Notieren Sie die Ausgabe zur Verwendung im nächsten Schritt.

  ii. Stellen Sie die Knotengruppe mithilfe der Anweisungen in the section called “Amazon Linux” bereit.

> **Note**
>
> Wenn Sie möchten, dass Knoten in einem Produktions-Cluster eine deutlich höhere Anzahl von Pods unterstützen, führen Sie das Skript in the section called “Von Amazon EKS empfohlene maximale Pods für jeden Amazon-EC2-Instance-Typ” erneut aus. Fügen Sie außerdem die Option `--cni-prefix-delegation-enabled` zu dem Befehl hinzu. Beispielsweise wird `110` für einen `m5.large`-Instance-Typ zurückgegeben. Anweisungen, wie Sie diese Funktion aktivieren, finden Sie unter the section called “IP-Adressen erhöhen”. Sie können diese Funktion mit benutzerdefinierten Netzwerken verwenden.

3. Die Erstellung der Knotengruppe dauert mehrere Minuten. Sie können den Status der Erstellung einer verwalteten Knotengruppe mit dem folgenden Befehl überprüfen.

```
aws eks describe-nodegroup --cluster-name my-custom-networking-cluster --nodegroup-name my-nodegroup --query nodegroup.status --output text
```

   Fahren Sie erst mit dem nächsten Schritt fort, wenn die zurückgegebene Ausgabe `ACTIVE` ist.

4. Für das Tutorial können Sie diesen Schritt überspringen.

Wenn Sie Ihre `ENIConfigs` für einen Produktions-Cluster nicht genauso benannt haben wie die Availability Zone, für die Sie diese verwenden, müssen Sie in Ihren Knoten den Namen `ENIConfig` anmerken, die mit dem Knoten verwendet werden soll. Dieser Schritt ist nicht erforderlich, wenn Sie in jeder Availability Zone nur ein Subnetz haben und Ihre `ENIConfigs` denselben Namen haben wie Ihre Availability Zones. Dies liegt daran, dass das Amazon VPC CNI-Plugin für Kubernetes automatisch das korrekte `ENIConfig` dem Knoten zuordnet, wenn Sie dies in einem vorherigen Schritt aktiviert haben.

a. Rufen Sie die Liste der Knoten in Ihrem Cluster ab.

```
kubectl get nodes
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME                                           STATUS   ROLES    AGE     VERSION
ip-192-168-0-126.us-west-2.compute.internal   Ready    <none>   8m49s   v1.22.9-
eks-810597c
ip-192-168-0-92.us-west-2.compute.internal    Ready    <none>   8m34s   v1.22.9-
eks-810597c
```

b. Bestimmen Sie, in welcher Availability Zone sich jeder Knoten befindet. Führen Sie den folgenden Befehl für jeden Knoten aus, der im vorherigen Schritt zurückgegeben wurde, und ersetzen Sie die IP-Adressen basierend auf der vorherigen Ausgabe.

```
aws ec2 describe-instances --filters Name=network-interface.private-dns-
name,Values=ip-192-168-0-126.us-west-2.compute.internal \
--query 'Reservations[].Instances[].{AvailabilityZone: Placement.AvailabilityZone,
 SubnetId: SubnetId}'
```

Eine Beispielausgabe sieht wie folgt aus.

```
[
    {
        "AvailabilityZone": "us-west-2d",
        "SubnetId": "subnet-Example5"
    }
]
```

c. Kommentieren Sie jeden Knoten mit der `ENIConfig`, die Sie für die Subnetz-ID und die Availability Zone erstellt haben. Sie können einen Knoten nur mit einer `ENIConfig`

kommentieren, aber es können mehrere Knoten mit derselben `ENIConfig` kommentiert werden. Ersetzen Sie die Beispielwerte durch Ihre eigenen Werte.

```
kubectl annotate node ip-192-168-0-126.us-west-2.compute.internal
 k8s.amazonaws.com/eniConfig=EniConfigName1
kubectl annotate node ip-192-168-0-92.us-west-2.compute.internal
 k8s.amazonaws.com/eniConfig=EniConfigName2
```

5. Wenn Sie Knoten in einem Produktions-Cluster hatten, in dem Pods ausgeführt wurden, bevor Sie zur Verwendung des benutzerdefinierten Netzwerk-Features gewechselt haben, führen Sie die folgenden Schritte aus:

   a. Stellen Sie sicher, dass es verfügbare Knoten gibt, das benutzerdefinierte Netzwerkfeature verwenden.

   b. Sperren und entleeren Sie die Knoten, um die Pods herunterzufahren. Weitere Informationen finden Sie unter Sicheres Entleeren eines Knotens in der Kubernetes-Dokumentation.

   c. Beenden Sie die Knoten. Wenn sich die Knoten in einer vorhandenen verwalteten Knotengruppe befinden, können Sie die Knotengruppe löschen. Führen Sie den folgenden Befehl aus.

   ```
   aws eks delete-nodegroup --cluster-name my-custom-networking-cluster --nodegroup-
   name my-nodegroup
   ```

   Nur neue Knoten, die mit der Bezeichnung `k8s.amazonaws.com/eniConfig` registriert sind, verwenden das neue benutzerdefinierte Netzwerkfeature.

6. Bestätigen Sie, dass Pos eine IP-Adresse aus einem CIDR-Block zugewiesen wird, der einem der in einem vorherigen Schritt erstellten Subnetze zugeordnet ist.

   ```
   kubectl get pods -A -o wide
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   NAMESPACE     NAME                       READY   STATUS    RESTARTS   AGE     IP
             NODE                                            NOMINATED NODE   READINESS
    GATES
   kube-system   aws-node-2rkn4             1/1     Running   0          7m19s
    192.168.0.92    ip-192-168-0-92.us-west-2.compute.internal    <none>
    <none>
   ```

```
kube-system   aws-node-k96wp             1/1     Running   0          7m15s
 192.168.0.126   ip-192-168-0-126.us-west-2.compute.internal   <none>
 <none>
kube-system   coredns-657694c6f4-smcgr   1/1     Running   0          56m
 192.168.1.23    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   coredns-657694c6f4-stwv9   1/1     Running   0          56m
 192.168.1.28    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   kube-proxy-jgshq           1/1     Running   0          7m19s
 192.168.0.92    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   kube-proxy-wx9vk           1/1     Running   0          7m15s
 192.168.0.126   ip-192-168-0-126.us-west-2.compute.internal   <none>
 <none>
```

Sie können feststellen, dass den coredns-Pods IP-Adressen aus dem `192.168.1.0`-CIDR-Block zugewiesen werden, den Sie Ihrer VPC hinzugefügt haben. Ohne benutzerdefiniertes Netzwerk wären diesen Adressen aus dem CIDR-Block `192.168.0.0` zugewiesen worden, da dies der einzige ursprünglich mit der VPC verbundene CIDR-Block war.

Wenn die Pods eines `spec` `hostNetwork=true` enthalten, wird diesem die primäre IP-Adresse des Knotens zugewiesen. Es wird keine Adresse aus den hinzugefügten Subnetzen zugewiesen. Standardmäßig ist dieser Wert auf `false` festgelegt. Dieser Wert wird für die `kube-proxy` und Amazon-VPC-CNI-Plugin für Kubernetes (`aws-node`)-Pods, die auf Ihrem Cluster ausgeführt werden, auf `true` gesetzt. Aus diesem Grund werden den `kube-proxy`- und `aws-node`-Pods des Plugins in der vorherigen Ausgabe keine Adressen 192.168.1.x zugewiesen. Weitere Informationen zu den `hostNetwork` Einstellungen eines Pods finden Sie unter PodSpec v1 core in der Kubernetes-API-Referenz.

Schritt 5: Löschen von Tutorial-Ressourcen

Nachdem Sie das Tutorial abgeschlossen haben, empfehlen wir, dass Sie die erstellten Ressourcen löschen. Sie können dann die Schritte anpassen, um benutzerdefinierte Netzwerke für einen Produktions-Cluster zu aktivieren.

1. Wenn die von Ihnen erstellte Knotengruppe nur zum Testen gedacht ist, löschen Sie sie.

```
aws eks delete-nodegroup --cluster-name my-custom-networking-cluster --nodegroup-name
 my-nodegroup
```

2. Selbst wenn die AWS CLI-Ausgabe besagt, dass der Cluster gelöscht wurde, ist der Löschvorgang möglicherweise noch nicht abgeschlossen. Der Löschvorgang dauert einige Minuten. Bestätigen Sie, dass der Vorgang abgeschlossen ist, indem Sie den folgenden Befehl ausführen.

   ```
   aws eks describe-nodegroup --cluster-name my-custom-networking-cluster --nodegroup-
   name my-nodegroup --query nodegroup.status --output text
   ```

   Fahren Sie erst fort, wenn die zurückgegebene Ausgabe der folgenden Ausgabe entspricht.

   ```
   An error occurred (ResourceNotFoundException) when calling the DescribeNodegroup
    operation: No node group found for name: my-nodegroup.
   ```

3. Wenn die von Ihnen erstellte Knotengruppe nur zum Testen gedacht ist, löschen Sie den Knoten IAM-Rolle.

   a. Trennen Sie die Richtlinien von der Rolle.

   ```
   aws iam detach-role-policy --role-name myCustomNetworkingNodeRole --policy-arn
    arn:aws: iam::aws:policy/AmazonEKSWorkerNodePolicy
   aws iam detach-role-policy --role-name myCustomNetworkingNodeRole --policy-arn
    arn:aws: iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
   aws iam detach-role-policy --role-name myCustomNetworkingNodeRole --policy-arn
    arn:aws: iam::aws:policy/AmazonEKS_CNI_Policy
   ```

   b. Löschen Sie die Rolle.

   ```
   aws iam delete-role --role-name myCustomNetworkingNodeRole
   ```

4. Löschen Sie den Cluster.

   ```
   aws eks delete-cluster --name my-custom-networking-cluster
   ```

   Bestätigen Sie, dass der Cluster gelöscht ist, indem Sie den folgenden Befehl ausführen.

   ```
   aws eks describe-cluster --name my-custom-networking-cluster --query cluster.status
    --output text
   ```

Wird eine Ausgabe ähnlich der folgenden zurückgegeben, wurde der Cluster erfolgreich gelöscht.

```
An error occurred (ResourceNotFoundException) when calling the DescribeCluster
 operation: No cluster found for name: my-custom-networking-cluster.
```

5. Löschen Sie die Cluster-IAM-Rolle.
   a. Trennen Sie die Richtlinien von der Rolle.

   ```
   aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSClusterRole --
   policy-arn arn:aws: iam::aws:policy/AmazonEKSClusterPolicy
   ```

   b. Löschen Sie die Rolle.

   ```
   aws iam delete-role --role-name myCustomNetworkingAmazonEKSClusterRole
   ```

6. Löschen Sie die Subnetze, die Sie in einem vorherigen Schritt erstellt haben.

```
aws ec2 delete-subnet --subnet-id $new_subnet_id_1
aws ec2 delete-subnet --subnet-id $new_subnet_id_2
```

7. Löschen Sie die VPC, die Sie erstellt haben.

```
aws cloudformation delete-stack --stack-name my-eks-custom-networking-vpc
```

Zuweisung weiterer IP-Adressen mit Präfixen zu Amazon-EKS-Knoten

Gilt für: Linux- und Windows-Knoten mit Amazon-EC2-Instances

Gilt für: Öffentliche und private Subnetze

Jede Amazon-EC2-Instance unterstützt eine maximale Anzahl von elastischen Netzwerkschnittstellen und eine maximale Anzahl von IP-Adressen, die jeder Netzwerkschnittstelle zugewiesen werden können. Jeder Knoten benötigt eine IP-Adresse für jede Netzwerkschnittstelle. Alle anderen verfügbaren IP-Adressen können `Pods` zugewiesen werden. Jeder `Pod` benötigt seine eigene IP-Adresse. Daher verfügen Sie möglicherweise über Knoten, die über verfügbare Datenverarbeitungs- und Speicherressourcen verfügen, aber keine weiteren `Pods` aufnehmen können, da der Knoten nicht mehr über IP-Adressen verfügt, die `Pods` zugewiesen werden können.

Sie können die Anzahl der IP-Adressen erhöhen, die Knoten Pods zuweisen können, indem Sie IP-Präfixe zuweisen, anstatt Ihren Knoten einzelne sekundäre IP-Adressen zuzuweisen. Jedes Präfix enthält mehrere IP-Adressen. Wenn Sie Ihren Cluster nicht für die IP-Präfixzuweisung konfigurieren, muss Ihr Cluster mehr Amazon-EC2-Anwendungsprogrammierschnittstellen (API)-Aufrufe durchführen, um die für die Pod-Konnektivität erforderlichen Netzwerkschnittstellen und IP-Adressen zu konfigurieren. Mit zunehmender Cluster-Größe kann die Häufigkeit dieser API-Aufrufe zu längeren Pod- und Instance-Startzeiten führen. Dies führt zu Skalierungsverzögerungen, um die Nachfrage nach großen und spitzen Workloads zu decken, und erhöht den Kosten- und Verwaltungsaufwand, da Sie zusätzliche Cluster und VPCs bereitstellen müssen, um die Skalierungsanforderungen zu erfüllen. Weitere Informationen finden Sie unter Kubernetes-Skalierbarkeitsschwellenwerte auf GitHub.

Kompatibilität mit den Amazon-VPC-CNI-Plugin für Kubernetes-Features

Sie können IP-Präfixe mit den folgenden Feature verwenden:

- IPv4-Quellennetzwerk-Adressübersetzung – Weitere Informationen finden Sie unter the section called “Ausgehenden Datenverkehr”.
- IPv6-Adressen für Cluster, Pods und Services – Weitere Informationen finden Sie unter the section called “IPv6”.
- Einschränkung des Datenverkehrs mithilfe von Kubernetes-Netzwerkrichtlinien – Weitere Informationen finden Sie unter the section called “Kubernetes-Richtlinien”.

Die folgende Liste enthält Informationen zu den geltenden Einstellungen für das Amazon-VPC-CNI-Plugin. Weitere Informationen zu den einzelnen Einstellungen finden Sie unter amazon-vpc-cni-k8s auf GitHub.

- `WARM_IP_TARGET`
- `MINIMUM_IP_TARGET`
- `WARM_PREFIX_TARGET`

Überlegungen

Beachten Sie bei der Verwendung von diesem Feature Folgendes:

- Jeder Amazon-EC2-Instance-Typ unterstützt eine maximale Anzahl von Pods. Wenn Ihre verwaltete Knotengruppe aus mehreren Instance-Typen besteht, wird die kleinste maximale Anzahl von Pods für eine Instance im Cluster auf alle Knoten im Cluster angewendet.
- Standardmäßig ist die maximale Anzahl von `Pods`, die Sie auf einem Knoten ausführen können, 110, aber Sie können diese Anzahl ändern. Wenn Sie die Anzahl ändern und eine verwaltete Knotengruppe haben, führt die nächste AMI- oder Startvorlagenaktualisierung Ihrer Knotengruppe dazu, dass neue Knoten mit dem geänderten Wert erstellt werden.
- Beim Übergang von der Zuweisung von IP-Adressen zur Zuweisung von IP-Präfixen empfehlen wir, dass Sie neue Knotengruppen erstellen, um die Anzahl der verfügbaren IP-Adressen zu erhöhen, anstatt die vorhandenen Knoten fortlaufend zu ersetzen. Die Ausführung von Pods in einem Knoten, dem sowohl IP-Adressen als auch Präfixe zugewiesen sind, kann zu Inkonsistenzen bei der angekündigten IP-Adresskapazität führen, was sich auf die zukünftigen Workloads auf dem Knoten auswirkt. Die empfohlene Vorgehensweise für die Umstellung finden Sie im Abschnitt Prefix-Delegierungsmodus für Linux im Leitfaden für bewährte Methoden für Amazon EKS.
- Der Geltungsbereich der Sicherheitsgruppe befindet sich auf Knotenebene. Weitere Informationen finden Sie unter Sicherheitsgruppe.
- Einer Netzwerkschnittstelle zugewiesene IP-Präfixe unterstützen eine hohe Pod-Dichte pro Knoten und haben die beste Startzeit.
- IP-Präfixe und IP-Adressen sind mit standardmäßigen elastischen Amazon-EC2-Netzwerkschnittstellen verknüpft. Pods, die bestimmte Sicherheitsgruppen erfordern, wird die primäre IP-Adresse einer Zweigstellen-Netzwerkschnittstelle zugewiesen. Sie können Pods, die IP-Adressen erhalten, oder IP-Adressen von IP-Präfixen mit Pods kombinieren, die Zweigstellen-Netzwerkschnittstellen auf demselben Knoten erhalten.
- Nur für Cluster mit Linux-Knoten.
  - Nachdem Sie das Add-On so konfiguriert haben, dass Netzwerkschnittstellen Präfixe zugewiesen werden, können Sie Ihr Add-On für Amazon-VPC-CNI-Plugin für Kubernetes nicht auf eine niedrigere Version als `1.9.0` (oder `1.10.1`) herabstufen, ohne alle Knoten in allen Knotengruppen in Ihrem Cluster zu entfernen.
  - Wenn Sie auch Sicherheitsgruppen für Pods mit `POD_SECURITY_GROUP_ENFORCING_MODE=standard` und `AWS_VPC_K8S_CNI_EXTERNALSNAT=false` verwenden, werden bei der Kommunikation Ihrer Pods mit Endpunkten außerhalb Ihrer VPC die Sicherheitsgruppen des Knotens verwendet und nicht die Sicherheitsgruppen, die Sie Ihren Pods zugewiesen haben.

Wenn Sie auch Sicherheitsgruppen für Pods mit `POD_SECURITY_GROUP_ENFORCING_MODE=strict` verwenden, werden die `Pod's`-Sicherheitsgruppen verwendet, wenn Ihr `Pods` mit Endpunkten außerhalb Ihres VPC kommunizieren.

Anzahl der verfügbaren IP-Adressen für Ihren Amazon-EKS-Knoten erhöhen

Sie können die Anzahl der IP-Adressen erhöhen, die Knoten Pods zuweisen können, indem Sie IP-Präfixe zuweisen, anstatt Ihren Knoten einzelne sekundäre IP-Adressen zuzuweisen.

Voraussetzungen

- Sie benötigen einen vorhandenen Cluster. Informationen zum Bereitstellen finden Sie unter the section called “Erstellen eines Clusters”.
- Die Subnetze, in denen sich Ihre Amazon-EKS-Knoten befinden, müssen über ausreichend zusammenhängende `/28` (für `IPv4`-Cluster) oder `/80` (für `IPv6`-Cluster) CIDR-Blöcke (Classless Inter-Domain Routing) verfügen. Sie können nur Linux-Knoten in einem `IPv6`-Cluster haben. Die Verwendung von IP-Präfixen kann fehlschlagen, wenn IP-Adressen im Subnetz-CIDR verstreut sind. Wir empfehlen Folgendes:
  - Verwendung einer Subnetz-CIDR-Reservierung, sodass selbst dann, wenn IP-Adressen innerhalb des reservierten Bereichs noch verwendet werden, die IP-Adressen nach ihrer Veröffentlichung nicht neu zugewiesen werden. Dadurch wird sichergestellt, dass Präfixe für die Zuweisung ohne Segmentierung verfügbar sind.
  - Verwenden Sie neue Subnetze, die speziell für die Ausführung der Workloads verwendet werden, denen IP-Präfixe zugewiesen sind. Sowohl Windows- als auch Linux-Workloads können bei der Zuordnung von IP-Präfixen im selben Subnetz ausgeführt werden.
- Um Ihren Knoten IP-Präfixe zuzuweisen, müssen Ihre Knoten AWS Nitro-basiert sein. Instances, die nicht Nitro-basiert sind, weisen weiterhin einzelne sekundäre IP-Adressen zu, weisen jedoch eine deutlich geringere Anzahl von IP-Adressen auf, die Pods zugewiesen werden können als Nitro-basierte Instances.
- Nur für Cluster mit Linux-Knoten – Wenn Ihr Cluster für die `IPv4`-Familie konfiguriert ist, muss Version `1.9.0` oder höher des Amazon-VPC-CNI-Plugins für das Kubernetes-Add-On installiert sein. Sie können Ihre aktuelle Version mit dem folgenden Befehl überprüfen.

```
kubectl describe daemonset aws-node --namespace kube-system | grep Image | cut -d "/"
 -f 2
```

Wenn Ihr Cluster für die IPv6-Familie konfiguriert ist, müssen Sie die Version 1.10.1 des Add-ons installiert haben. Wenn Ihre Plugin-Version älter ist als die erforderlichen Versionen, müssen Sie sie aktualisieren. Weitere Informationen finden Sie in den Aktualisierungsabschnitten von Assign IPs to Pods with the Amazon VPC CNI.

- Nur für Cluster mit Windows-Knoten
  - Sie müssen den Windows-Support für Ihren Cluster aktiviert haben. Weitere Informationen finden Sie unter the section called “Windows-Support aktivieren”.

Knoten IP-Adress-Präfixe zuweisen

Konfigurieren Sie Ihren Cluster, um Knoten IP-Adresspräfixe zuzuweisen. Führen Sie den Vorgang durch, der dem Betriebssystem Ihres Knotens entspricht.

Linux

1. Aktivieren Sie den Parameter, um Netzwerkschnittstellen für das Amazon VPC CNI Präfixe zuzuweisen. DaemonSet Wenn Sie einen Cluster bereitstellen, wird die Version 1.10.1 oder eine neuere Version des Amazon-VPC-CNI-Plugins für Kubernetes-Add-On mit diesem bereitgestellt. Wenn Sie den Cluster mit der IPv6-Familie erstellt haben, wurde für diese Einstellung standardmäßig true festgelegt. Wenn Sie den Cluster mit der IPv4-Familie erstellt haben, wurde für diese Einstellung standardmäßig false festgelegt.

```
kubectl set env daemonset aws-node -n kube-system ENABLE_PREFIX_DELEGATION=true
```

> **Important**
>
> Auch wenn Ihr Subnetz über verfügbare IP-Adressen verfügt, wird in den Protokollen des Amazon-VPC-CNI-Plugins für Kubernetes der folgende Fehler angezeigt, wenn im Subnetz keine zusammenhängenden /28-Blöcke verfügbar sind:
>
> ```
> InsufficientCidrBlocks: The specified subnet does not have enough free cidr
>  blocks to satisfy the request
> ```
>
> Dies kann aufgrund der Fragmentierung vorhandener sekundärer IP-Adressen auftreten, die über ein Subnetz verteilt sind. Um diesen Fehler zu beheben, erstellen Sie entweder ein neues Subnetz und starten Sie Pods dort, oder verwenden Sie eine CIDR-Reservierung für das EC2 Amazon-Subnetz, um Speicherplatz innerhalb eines Subnetzes

für die Verwendung mit Präfixzuweisung zu reservieren. Weitere Informationen erhalten Sie unter Subnetz-CIDR-Reservierungen im Amazon-VPC-Benutzerhandbuch.

2. Wenn Sie planen, eine verwaltete Knotengruppe ohne Startvorlage bereitzustellen oder mit einer Startvorlage, in der Sie keine AMI-ID angegeben haben, und Sie eine Version des Amazon-VPC-CNI-Plugins für Kubernetes verwenden, die mindestens der in den Voraussetzungen gelisteten Version entspricht, fahren Sie mit dem nächsten Schritt fort. Verwaltete Knotengruppen berechnen automatisch die maximale Anzahl von Pods für Sie.

   Wenn Sie eine selbstverwaltete Knotengruppe oder eine verwaltete Knotengruppe mit einer Startvorlage bereitstellen, in der Sie eine AMI-ID angegeben haben, müssen Sie die von Amazon EKS empfohlene Anzahl der maximalen Pods für Ihre Knoten ermitteln. Folgen Sie den Anweisungen in den von Amazon EKS empfohlenen maximalen Pods für jeden EC2 Amazon-Instance-Typ und fügen Sie Schritt 3 `--cni-prefix-delegation-enabled` hinzu. Notieren Sie sich die Ausgabe zur Verwendung in einem späteren Schritt.

   > **Important**
   >
   > Verwaltete Knotengruppen erzwingen eine maximale Anzahl für den Wert von `maxPods`. Für Instances mit weniger als 30 V ist CPUs die maximale Anzahl 110 und für alle anderen Instances ist die maximale Anzahl 250. Diese maximale Anzahl wird angewendet, unabhängig davon, ob die Präfixdelegierung aktiviert ist oder nicht.

3. Wenn Sie einen für `IPv6` konfigurierten Cluster verwenden, fahren Sie mit dem nächsten Schritt fort.

   Geben Sie die Parameter in einer der folgenden Optionen an. Um zu ermitteln, welche Option für Sie die richtige ist und welchen Wert Sie dafür angeben müssen, finden Sie unter WARM_PREFIX_TARGET, WARM_IP_TARGET und MINIMUM_IP_TARGET on. GitHub

   Sie können die Beispielwerte durch einen Wert größer als Null ersetzen.

   - `WARM_PREFIX_TARGET`

     ```
     kubectl set env ds aws-node -n kube-system WARM_PREFIX_TARGET=1
     ```

   - `WARM_IP_TARGET` oder `MINIMUM_IP_TARGET` – Wenn einer der Werte festgelegt ist, überschreibt er alle Werte, die für `WARM_PREFIX_TARGET` festgelegt wurden.

```
kubectl set env ds aws-node -n kube-system WARM_IP_TARGET=5
```

```
kubectl set env ds aws-node -n kube-system MINIMUM_IP_TARGET=2
```

4. Erstellen Sie einen der folgenden Typen von Knotengruppen mit mindestens einem Amazon EC2 Nitro Amazon Linux 2023 Instance-Typ. Eine Liste der Nitro-Instance-Typen finden Sie unter Instances built on the Nitro System im EC2 Amazon-Benutzerhandbuch. Diese Funktion wird unter Windows nicht unterstützt. Für die Optionen, die *110* enthalten, ersetzen Sie es entweder durch den Wert aus Schritt 3 (empfohlen) oder Ihren eigenen Wert.

   - Selbstverwaltet – Stellen Sie die Knotengruppe gemäß den Anweisungen unter Erstellen selbstverwalteter Amazon-Linux-Knoten bereit. Bevor Sie den CloudFormation Stack erstellen, öffnen Sie die Vorlagendatei und passen Sie sie wie folgt `NodeLaunchTemplate` an `UserData`

     ```
     ...
                 apiVersion: node.eks.aws/v1alpha1
                 kind: NodeConfig
                 spec:
                   cluster:
                     name: ${ClusterName}
                     apiServerEndpoint: ${ApiServerEndpoint}
                     certificateAuthority: ${CertificateAuthorityData}
                     cidr: ${ServiceCidr}
                   kubelet:
                     config:
                       maxPods: 110
     ...
     ```

     Wenn Sie `eksctl` zum Erstellen der Knotengruppe verwenden, können Sie den folgenden Befehl verwenden.

     ```
     eksctl create nodegroup --cluster my-cluster --managed=false --max-pods-per-node
      110
     ```

   - Verwaltet – Stellen Sie Ihre Knotengruppe mit einer der folgenden Optionen bereit:
     - Ohne Startvorlage oder mit einer Startvorlage ohne angegebene AMI-ID – Führen Sie das Verfahren unter Verwaltete Knotengruppe für Ihren Cluster erstellen durch. Verwaltete Knotengruppen berechnen automatisch den von Amazon EKS empfohlenen `max-pods`-Wert für Sie.

- Mit einer Startvorlage mit einer angegebenen AMI-ID – Geben Sie in Ihrer Startvorlage eine Amazon EKS-optimierte AMI-ID oder ein benutzerdefiniertes AMI an, das auf dem Amazon EKS-optimierten AMI basiert, stellen Sie dann die Knotengruppe mithilfe einer Startvorlage bereit und geben Sie die folgenden Benutzerdaten an in der Startvorlage. Diese Benutzerdaten übergeben ein `NodeConfig` Objekt, das vom `nodeadm` Tool auf dem Knoten gelesen werden soll. Weitere Informationen zu `nodeadm` finden Sie in der nodeadm-Dokumentation.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="//"

--//
Content-Type: application/node.eks.aws

---
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
 cluster:
   apiServerEndpoint: [.replaceable]`my-cluster`
   certificateAuthority: [.replaceable]`LS0t...`
   cidr: [.replaceable]`10.100.0.0/16`
   name: [.replaceable]`my-cluster
 kubelet:
   config:
     maxPods: [.replaceable]`110`
--//--
```

Wenn Sie `eksctl` zum Erstellen der Knotengruppe verwenden, können Sie den folgenden Befehl verwenden.

```
eksctl create nodegroup --cluster my-cluster --max-pods-per-node 110
```

Wenn Sie ein benutzerdefiniertes AMI erstellt haben, das nicht auf dem für Amazon EKS optimierten AMI erstellt wurde, müssen Sie die Konfiguration selbst erstellen.

> **Note**
>
> Wenn Sie Pods auch IP-Adressen aus einem anderen Subnetz als dem der Instance zuweisen möchten, müssen Sie die Funktion in diesem Schritt aktivieren. Weitere Informationen finden Sie unter the section called “Benutzerdefinierte Netzwerke”.

Windows

1. Aktivieren Sie die Zuweisung von IP-Präfixen.
   a. Öffnen Sie die `amazon-vpc-cni` `ConfigMap` zum Bearbeiten.

   ```
   kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
   ```

   b. Fügen Sie dem Abschnitt `data` die folgende Zeile hinzu.

   ```
     enable-windows-prefix-delegation: "true"
   ```

   c. Speichern Sie die Datei und schließen Sie den Editor.
   d. Vergewissern Sie sich, dass die Anmerkung der `ConfigMap` hinzugefügt wurde.

   ```
   kubectl get configmap -n kube-system amazon-vpc-cni -o "jsonpath={.data.enable-
   windows-prefix-delegation}"
   ```

   Wenn die zurückgegebene Ausgabe nicht `true` ist, ist möglicherweise ein Fehler aufgetreten. Versuchen Sie erneut, den Schritt abzuschließen.

   > **Important**
   >
   > Auch wenn Ihr Subnetz über verfügbare IP-Adressen verfügt, wird in den Protokollen des Amazon-VPC-CNI-Plugins für Kubernetes der folgende Fehler angezeigt, wenn im Subnetz keine zusammenhängenden `/28`-Blöcke verfügbar sind:
   >
   > ```
   > InsufficientCidrBlocks: The specified subnet does not have enough free cidr
   >  blocks to satisfy the request
   > ```
   >
   > Dies kann aufgrund der Fragmentierung vorhandener sekundärer IP-Adressen auftreten, die über ein Subnetz verteilt sind. Um diesen Fehler zu beheben, erstellen

Sie entweder ein neues Subnetz und starten Sie Pods dort, oder verwenden Sie eine CIDR-Reservierung für das EC2 Amazon-Subnetz, um Speicherplatz innerhalb eines Subnetzes für die Verwendung mit Präfixzuweisung zu reservieren. Weitere Informationen erhalten Sie unter Subnetz-CIDR-Reservierungen im Amazon-VPC-Benutzerhandbuch.

2. (Optional) Geben Sie eine zusätzliche Konfiguration an, um das Verhalten Ihres Clusters vor der Skalierung und der dynamischen Skalierung zu steuern. Weitere Informationen finden Sie unter Konfigurationsoptionen mit aktiviertem Präfix-Delegierungsmodus unter Windows. GitHub

   a. Öffnen Sie die `amazon-vpc-cni` `ConfigMap` zum Bearbeiten.

   ```
   kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
   ```

   b. Ersetzen Sie die Beispielwerte durch einen Wert größer als Null und fügen Sie die benötigten Einträge zum `data` Abschnitt von hinzu`ConfigMap`. Wenn Sie für `warm-ip-target` oder `minimum-ip-target` einen Wert festlegen, überschreibt der Wert jeden Wert, der für `warm-prefix-target` festgelegt wurde.

   ```
   warm-prefix-target: "1"
   warm-ip-target: "5"
   minimum-ip-target: "2"
   ```

   c. Speichern Sie die Datei und schließen Sie den Editor.

3. Erstellen Sie Windows-Knotengruppen mit mindestens einem Amazon EC2 Nitro-Instance-Typ. Eine Liste der Nitro-Instance-Typen finden Sie unter Instances built on the Nitro System im EC2 Amazon-Benutzerhandbuch. Standardmäßig beträgt die maximale Anzahl von Pods, die Sie auf einem Knoten bereitstellen können, 110. Wenn Sie diese Zahl erhöhen oder verringern möchten, geben Sie in den Benutzerdaten für die Bootstrap-Konfiguration Folgendes an. Ersetzen Sie *`max-pods-quantity`* durch Ihren Wert für die maximale Anzahl von Pods.

   ```
   -KubeletExtraArgs '--max-pods=max-pods-quantity'
   ```

   Wenn Sie verwaltete Knotengruppen bereitstellen, muss diese Konfiguration zur Startvorlage hinzugefügt werden. Weitere Informationen finden Sie unter the section called “Startvorlagen”. Weitere Informationen zu den Konfigurationsparametern für das Windows-Bootstrap-Skript finden Sie unter the section called “Bootstrap-Skript-Konfigurationsparameter”.

Bestimmung der maximalen Anzahl von Pods und verfügbaren IP-Adressen

1. Nachdem Ihre Knoten bereitgestellt wurden, zeigen Sie die Knoten in Ihrem Cluster an.

   ```
   kubectl get nodes
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   NAME                                           STATUS     ROLES    AGE   VERSION
   ip-192-168-22-103.region-code.compute.internal   Ready      <none>   19m   v1.XX.X-
   eks-6b7464
   ip-192-168-97-94.region-code.compute.internal    Ready      <none>   19m   v1.XX.X-
   eks-6b7464
   ```

2. Beschreiben Sie einen der Knoten, um den Wert von `max-pods` für den Knoten und die Anzahl der verfügbaren IP-Adressen zu bestimmen. Ersetzen Sie *`192.168.30.193`* durch die `IPv4`-Adresse im Namen eines Ihrer Knoten, der in der Ausgabe des vorherigen Schritts zurückgegeben wurde.

   ```
   kubectl describe node ip-192-168-30-193.region-code.compute.internal | grep 'pods\|
   PrivateIPv4Address'
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   pods:                                  110
   vpc.amazonaws.com/PrivateIPv4Address:  144
   ```

   In der vorherigen Ausgabe `110` ist dies die maximale Anzahl von Pods, die Kubernetes auf dem Knoten bereitstellen wird, obwohl *`144`* IP-Adressen verfügbar sind.

Einzelnen Pods Sicherheitsgruppen zuweisen

Gilt für: Linux-Knoten mit EC2 Amazon-Instances

Gilt für: Private Subnetze

Sicherheitsgruppen für Pods integrieren EC2 Amazon-Sicherheitsgruppen in Kubernetes Pods. Sie können EC2 Amazon-Sicherheitsgruppen verwenden, um Regeln zu definieren, die eingehenden und ausgehenden Netzwerkverkehr zu und von Pods zulassen, die Sie auf Knoten bereitstellen, die

auf vielen EC2 Amazon-Instance-Typen und Fargate laufen. Eine ausführliche Erläuterung dieser Funktion finden Sie im Blogbeitrag Einführung in Sicherheitsgruppen für Pods.

Kompatibilität mit den Amazon-VPC-CNI-Plugin für Kubernetes-Features

Sie können Sicherheitsgruppen für Pods mit den folgenden Features verwenden:

- IPv4 Übersetzung der Quellnetzwerkadresse — Weitere Informationen finden Sie unter. the section called “Ausgehenden Datenverkehr”
- IPv6 Adressen für Cluster, Pods und Dienste — Weitere Informationen finden Sie unterthe section called “IPv6”.
- Einschränkung des Datenverkehrs mithilfe von Kubernetes-Netzwerkrichtlinien – Weitere Informationen finden Sie unter the section called “Kubernetes-Richtlinien”.

Überlegungen

Berücksichtigen Sie vor der Bereitstellung von Sicherheitsgruppen für Pods die folgenden Einschränkungen und Bedingungen:

- Sicherheitsgruppen für Pods können nicht mit Windows-Knoten oder dem EKS-Automodus verwendet werden.
- Sicherheitsgruppen für Pods können mit Clustern verwendet werden, die für die `IPv6` Familie konfiguriert sind und EC2 Amazon-Knoten enthalten. Verwenden Sie dazu Version 1.16.0 oder höher des Amazon VPC CNI-Plug-ins. Sie können Sicherheitsgruppen für Pods mit für die `IPv6`-Familie konfigurierten Clustern verwenden, die nur Fargate-Knoten enthalten. Hierfür muss mindestens die Version 1.7.7 des Amazon-VPC-CNI-Plugins verwendet werden. Weitere Informationen finden Sie unter the section called “IPv6”.
- Sicherheitsgruppen für Pods werden von den meisten Nitro-basierten EC2 Amazon-Instance-Familien unterstützt, allerdings nicht von allen Generationen einer Familie. Beispielsweise werden die Instance-Familien und -generationen `m5`, `c5`, `r5`, `m6g`, `c6g` und `r6g` unterstützt. Es werden keine Instance-Typen in der `t`-Familie unterstützt. Eine vollständige Liste der unterstützten Instance-Typen finden Sie in der Datei limits.go unter. GitHub Ihre Knoten müssen von einem der aufgelisteten Instance-Typen sein, die in der Datei mit `IsTrunkingCompatible: true` gekennzeichnet sind.
- Wenn Sie benutzerdefinierte Netzwerk- und Sicherheitsgruppen für Pods zusammen verwenden, wird die durch Sicherheitsgruppen für Pods angegebene Sicherheitsgruppe anstelle der im `ENIConfig` angegebenen Sicherheitsgruppe verwendet.

- Wenn Sie Version `1.10.2` oder älter des Amazon-VPC-CNI-Plugins verwenden und die Einstellung `terminationGracePeriodSeconds` in Ihre Pod-Spezifikation aufnehmen, darf der Wert der Einstellung nicht Null sein.
- Wenn Sie Version `1.10` oder neuer des Amazon-VPC-CNI-Plugins verwenden oder Version `1.11` mit `POD_SECURITY_GROUP_ENFORCING_MODE` = `strict`, welches die Standardeinstellung ist, dann werden Kubernetes-Services vom Typ `NodePort` und `LoadBalancer`, die Instance-Ziele verwenden, für die `externalTrafficPolicy` auf `Local` eingestellt ist, nicht für Pods unterstützt, denen Sie Sicherheitsgruppen zuweisen. Weitere Informationen zur Verwendung eines Load Balancers mit Instance-Zielen finden Sie unter the section called “Netzwerk-Load-Balancer”.
- Wenn Sie Version `1.10` des Amazon-VPC-CNI-Plugins oder höher verwenden oder Version `1.11` mit `POD_SECURITY_GROUP_ENFORCING_MODE` = `strict`, welches die Standardeinstellung ist, ist die Quell-NAT für von Pods mit zugewiesenen Sicherheitsgruppen ausgehenden Datenverkehr deaktiviert, damit Sicherheitsgruppenregeln für ausgehenden Datenverkehr angewendet werden können. Um auf das Internet zuzugreifen, müssen Pods mit zugewiesenen Sicherheitsgruppen auf Knoten gestartet werden, die in einem privaten Subnetz bereitgestellt werden, das mit einem NAT-Gateway oder einer NAT-Instance konfiguriert ist. Pods mit zugewiesenen Sicherheitsgruppen, welche in öffentlichen Subnetzen bereitgestellt werden, können nicht auf das Internet zugreifen.

  Wenn Sie Version `1.11` oder höher des Plug-Ins mit `POD_SECURITY_GROUP_ENFORCING_MODE` = `standard` verwenden, wird der für außerhalb der VPC bestimmte Pod-Datenverkehr in die IP-Adresse der primären Netzwerkschnittstelle der Instance übersetzt. Für diesen Datenverkehr werden die Regeln in den Sicherheitsgruppen für die primäre Netzwerkschnittstelle anstelle der Regeln in den Pods-Sicherheitsgruppen verwendet.
- Um die Calico-Netzwerkrichtlinie mit Pods zu verwenden, denen Sicherheitsgruppen zugewiesen sind, müssen Sie Version `1.11.0` oder höher des Amazon-VPC-CNI-Plugins verwenden und `POD_SECURITY_GROUP_ENFORCING_MODE` = `standard` festlegen. Andernfalls unterliegt der Verkehrsfluss zu und von Pods mit zugehörigen Sicherheitsgruppen nicht der Durchsetzung der Calico-Netzwerkrichtlinien und ist nur auf die Durchsetzung von EC2 Amazon-Sicherheitsgruppen beschränkt. Informationen zum Aktualisieren Ihrer Version von Amazon VPC CNI finden Sie unter the section called “Amazon VPC CNI”.
- Pods, die auf EC2 Amazon-Knoten ausgeführt werden und Sicherheitsgruppen in Clustern verwenden, die diese verwenden, NodeLocal DNSCachewerden nur mit Version `1.11.0` oder höher des Amazon VPC CNI-Plug-ins und mit `POD_SECURITY_GROUP_ENFORCING_MODE` = unterstützt. `standard` Informationen zum Aktualisieren Ihrer Version des Amazon-VPC-CNI-Plug-ins finden Sie unter the section called “Amazon VPC CNI”.

- Sicherheitsgruppen für Pods können bei Pods mit hoher Fluktuation zu einer höheren Pod-Startup-Latenz führen. Dies ist auf eine Ratenbegrenzung im Ressourcencontroller zurückzuführen.
- Der Geltungsbereich der EC2 Sicherheitsgruppe befindet sich auf Pod-Ebene. Weitere Informationen finden Sie unter Sicherheitsgruppe.

  Wenn Sie `POD_SECURITY_GROUP_ENFORCING_MODE=standard` und `AWS_VPC_K8S_CNI_EXTERNALSNAT=false` festlegen, verwendet der für Endpunkte außerhalb der VPC bestimmte Datenverkehr die Sicherheitsgruppen des Knotens und nicht die Sicherheitsgruppen des Pods.

Konfiguration des Amazon-VPC-CNI-Plugins für Kubernetes für Sicherheitsgruppen für Amazon-EKS-Pods

Wenn Sie Pods mit Amazon-EC2-Instances verwenden, müssen Sie das Amazon-VPC-CNI-Plugin für Kubernetes für Sicherheitsgruppen konfigurieren.

Wenn Sie ausschließlich Fargate-Pods verwenden und über keine Amazon-EC2-Knoten in Ihrem Cluster verfügen, lesen Sie the section called “SecurityGroupPolicy”.

1. Überprüfen Sie die aktuelle Version Ihres Amazon-VPC-CNI-Plugins für Kubernetes mit dem folgenden Befehl:

   ```
   kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni: |
    cut -d : -f 3
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   v1.7.6
   ```

   Wenn Ihre Amazon-VPC-CNI-Plugin für Kubernetes-Version älter als `1.7.7` ist, aktualisieren Sie das Plugin auf Version `1.7.7` oder höher. Weitere Informationen finden Sie unter the section called “Amazon VPC CNI”.
2. Fügen Sie die verwaltete IAM-Richtlinie AmazonEKSVPCResourceController der Cluster-Rolle hinzu, die Ihrem Amazon-EKS-Cluster zugeordnet ist. Die Richtlinie ermöglicht es der Rolle, Netzwerkschnittstellen, ihre privaten IP-Adressen sowie deren An- und Abkopplung an und von Netzwerk-Instances zu verwalten.

a. Rufen Sie den Namen Ihrer Cluster-IAM-Rolle ab und speichern Sie ihn in einer Variablen. Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters.

```
cluster_role=$(aws eks describe-cluster --name my-cluster --query cluster.roleArn
 --output text | cut -d / -f 2)
```

b. Fügen Sie der Rolle die -Richtlinie an.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSVPCResourceController --role-name $cluster_role
```

3. Aktivieren Sie das Amazon-VPC-CNI-Add-On, um Netzwerkschnittstellen für Pods zu verwalten, indem Sie die Variable `ENABLE_POD_ENI` im `aws-node` DaemonSet auf `true` setzen. Wenn diese Einstellung auf `true` festgelegt wurde, erstellt das Add-On für jeden Knoten im Cluster eine benutzerdefinierte `cninode`-Ressource. Der VPC-Ressourcen-Controller erstellt und verknüpft eine spezielle Netzwerkschnittstelle, die als Trunk-Netzwerkschnittstelle mit der Beschreibung `aws-k8s-trunk-eni` bezeichnet wird.

```
kubectl set env daemonset aws-node -n kube-system ENABLE_POD_ENI=true
```

> **Note**
>
> Die Trunk-Netzwerkschnittstelle ist in der maximalen Anzahl von Netzwerkschnittstellen enthalten, die vom Instance-Typ unterstützt werden. Eine Liste der maximalen Anzahl von Netzwerkschnittstellen, die von jedem Instance-Typ unterstützt werden, finden Sie unter IP-Adressen pro Netzwerkschnittstelle pro Instance-Typ im Amazon-EC2-Benutzerhandbuch. Wenn an Ihren Knoten bereits die maximale Anzahl an Standardnetzwerkschnittstellen angeschlossen ist, reserviert der VPC-Ressourcencontroller einen Speicherplatz. Sie müssen Ihre ausgeführten Pods so weit herunterskalieren, dass der Controller eine Standard-Netzwerkschnittstelle trennen und löschen, die Trunk-Netzwerkschnittstelle erstellen und an die Instance anhängen kann.

4. Mit dem folgenden Befehl können Sie ermitteln, welche Ihrer Knoten über eine benutzerdefinierte Ressource vom Typ `CNINode` verfügen. Wenn `No resources found` zurückgegeben wird, warten Sie einige Sekunden und versuchen Sie es dann erneut. Der vorherige Schritt erfordert einen Neustart des Amazon-VPC-CNI-Plugins für Kubernetes-Pods, was einige Sekunden in Anspruch nimmt.

```
kubectl get cninode -A
     NAME FEATURES
     ip-192-168-64-141.us-west-2.compute.internal [{"name":"SecurityGroupsForPods"}]
     ip-192-168-7-203.us-west-2.compute.internal [{"name":"SecurityGroupsForPods"}]
```

Wenn Sie VPC-CNI-Versionen verwenden, die älter als `1.15` sind, wurden anstelle der benutzerdefinierten `CNINode`-Ressource Knotenbezeichnungen verwendet. Mit dem folgenden Befehl können Sie ermitteln, bei welchen Ihrer Knoten die Knoten-Bezeichnung `aws-k8s-trunk-eni` auf `true` festgelegt ist. Wenn `No resources found` zurückgegeben wird, warten Sie einige Sekunden und versuchen Sie es dann erneut. Der vorherige Schritt erfordert einen Neustart des Amazon-VPC-CNI-Plugins für Kubernetes-Pods, was einige Sekunden in Anspruch nimmt.

```
kubectl get nodes -o wide -l vpc.amazonaws.com/has-trunk-attached=true
```

Nachdem die Trunk-Netzwerkschnittstelle erstellt wurde, werden Pods sekundäre IP-Adressen von der Trunk- oder Standardnetzwerkschnittstelle zugewiesen. Die Trunk-Schnittstelle wird automatisch gelöscht, wenn der Knoten gelöscht wird.

Wenn Sie in einem späteren Schritt eine Sicherheitsgruppe für einen Pod bereitstellen, erstellt der VPC-Ressourcen-Controller eine spezielle Netzwerkschnittstelle, die als Zweigstellennetzwerkschnittstelle bezeichnet wird, mit einer Beschreibung von `aws-k8s-branch-eni` und ordnet ihr die Sicherheitsgruppen zu. Neben den an den Knoten angeschlossenen Standard- und Amtsnetzschnittstellen werden Nebenstellennetzschnittstellen erstellt.

Wenn Sie Aktivitäts- oder Bereitschaftsprüfungen verwenden, müssen Sie auch frühe TCP Demux deaktivieren, damit das `kubelet` über TCP eine Verbindung zu Pods auf Zweig-Netzwerkschnittstellen herstellen kann. Führen Sie den folgenden Befehl aus, um das frühe TCP Demux zu deaktivieren:

```
kubectl patch daemonset aws-node -n kube-system \
  -p '{"spec": {"template": {"spec": {"initContainers": [{"env":
[{"name":"DISABLE_TCP_EARLY_DEMUX","value":"true"}],"name":"aws-vpc-cni-init"}]}}}}'
```

> **Note**
>
> Wenn Sie `1.11.0` oder höher des Amazon-VPC-CNI-Plugins für das Kubernetes-Add-On verwenden und `POD_SECURITY_GROUP_ENFORCING_MODE` = `standard` festlegen, wie im nächsten Schritt beschrieben, müssen Sie den vorherigen Befehl nicht ausführen.

5. Wenn Ihr Cluster `NodeLocal DNSCache` verwendet oder Sie die Calico-Netzwerkrichtlinie mit Ihren Pods verwenden möchten, die über eigene Sicherheitsgruppen verfügen, oder wenn Sie Kubernetes-Services vom Typ `NodePort` und `LoadBalancer` haben, die Instance-Ziele mit einem auf `Local` festgelegten `externalTrafficPolicy` für Pods verwenden, denen Sie Sicherheitsgruppen zuweisen möchten, müssen Sie Version `1.11.0` oder höher des Amazon-VPC-CNI-Plugins für das Kubernetes-Add-On verwenden und die folgende Einstellung aktivieren:

```
kubectl set env daemonset aws-node -n kube-system
 POD_SECURITY_GROUP_ENFORCING_MODE=standard
```

   WICHTIG: Pod-Sicherheitsgruppenregeln werden nicht auf den Datenverkehr zwischen Pods oder zwischen Pods und Services wie **`kubelet`** oder **`nodeLocalDNS`** angewendet, die sich auf demselben Knoten befinden. Pods, die verschiedene Sicherheitsgruppen auf demselben Knoten verwenden, können nicht kommunizieren, da sie in verschiedenen Subnetzen konfiguriert sind und das Routing zwischen diesen Subnetzen deaktiviert ist. Für ausgehenden Datenverkehr von Pods an Adressen außerhalb der VPC wird die Netzwerkadresse in die IP-Adresse der primären Netzwerkschnittstelle der Instance übersetzt (es sei denn, Sie haben auch `AWS_VPC_K8S_CNI_EXTERNALSNAT=true` festgelegt). Für diesen Datenverkehr werden die Regeln in den Sicherheitsgruppen für die primäre Netzwerkschnittstelle anstelle der Regeln in den Pods-Sicherheitsgruppen verwendet. ** Damit diese Einstellung auf vorhandene Pods angewendet wird, müssen Sie die Pods oder die Knoten, auf denen die Pods ausgeführt werden, neu starten.

6. Informationen zur Verwendung einer Sicherheitsgruppenrichtlinie für Ihren Pod finden Sie unter the section called “SecurityGroupPolicy”.

Sicherheitsgruppenrichtlinie für einen Amazon-EKS-Pod verwenden

Um Sicherheitsgruppen für Pods zu verwenden, benötigen Sie eine bereits vorhandene Sicherheitsgruppe. In den folgenden Schritten wird gezeigt, wie Sie die Sicherheitsgruppenrichtlinie für ein Pod verwenden. Wenn nicht anders angegeben, führen Sie alle Schritte vom selben Terminal

aus, da Variablen in den folgenden Schritten verwendet werden, die nicht über Terminals hinweg bestehen bleiben.

Wenn Sie einen Pod mit EC2 Amazon-Instances haben, müssen Sie das Plugin konfigurieren, bevor Sie dieses Verfahren verwenden. Weitere Informationen finden Sie unter the section called “Konfigurieren”.

1. Erstellen Sie einen Kubernetes-Namespace, in dem -Ressourcen bereitgestellt werden sollen. Sie können es *my-namespace* durch den Namen eines Namespaces ersetzen, den Sie verwenden möchten.

   ```
   kubectl create namespace my-namespace
   ```

2. Stellen Sie ein Amazon EKS `SecurityGroupPolicy` in Ihrem Cluster bereit.

   a. Kopieren Sie den folgenden Inhalt auf Ihr Gerät. Sie können es durch *podSelector* ersetzen, `serviceAccountSelector` wenn Sie Pods lieber anhand der Dienstkontobezeichnungen auswählen möchten. Sie müssen den einen oder anderen Selektor angeben. Ein leeres `podSelector` (Beispiel: `podSelector: {}`) wählt alle Pods im Namespace aus. Sie können *my-role* den Namen Ihrer Rolle ändern. Ein leeres `serviceAccountSelector` wählt alle Servicekonten im Namespace aus. Sie können ihn *my-security-group-policy* durch einen Namen für sich `SecurityGroupPolicy` und *my-namespace* durch den Namespace ersetzen, `SecurityGroupPolicy` in dem Sie die Datei erstellen möchten.

   Sie müssen es *my_pod_security_group_id* durch die ID einer vorhandenen Sicherheitsgruppe ersetzen. Wenn Sie nicht über eine vorhandene Sicherheitsgruppe verfügen, müssen Sie eine erstellen. Weitere Informationen finden Sie unter EC2 Amazon-Sicherheitsgruppen für Linux-Instances im EC2 Amazon-Benutzerhandbuch. Sie können eine Sicherheitsgruppe von 1 bis 5 angeben IDs. Falls Sie mehr als eine ID angeben, gilt die Kombination aller Regeln in allen Sicherheitsgruppen für die ausgewählten Pods.

   ```
   cat >my-security-group-policy.yaml <<EOF
   apiVersion: vpcresources.k8s.aws/v1beta1
   kind: SecurityGroupPolicy
   metadata:
     name: my-security-group-policy
     namespace: my-namespace
   spec:
     podSelector:
       matchLabels:
   ```

```
      role: my-role
  securityGroups:
    groupIds:
      - my_pod_security_group_id
EOF
```

> **Important**
>
> Die Sicherheitsgruppe oder -gruppen, die Sie für Ihre Pods angeben, müssen die folgenden Kriterien erfüllen:
>
> - Sie müssen vorhanden sein. Wenn sie nicht vorhanden sind, bleibt Ihr Pod im Erstellungsprozess hängen, wenn Sie einen Pod bereitstellen, der dem Selektor entspricht. Wenn Sie den Pod beschreiben, wird eine Fehlermeldung ähnlich der folgenden angezeigt: `An error occurred (InvalidSecurityGroupID.NotFound) when calling the CreateNetworkInterface operation: The securityGroup ID 'sg-05b1d815d1EXAMPLE' does not exist.`
> - Diese müssen eingehende Kommunikation von der auf Ihre Knoten angewendeten Sicherheitsgruppe (für `kubelet`) über alle Ports erlauben, für die Sie Tests konfiguriert haben.
> - Diese müssen ausgehende Kommunikation über die TCP- und UDP-Ports 53 zu einer Sicherheitsgruppe zulassen, die dem Pods (oder den Knoten, auf denen die Pods ausgeführt werden) zugeordnet ist, auf denen CoreDNS ausgeführt wird. Die Sicherheitsgruppe für Ihre CoreDNS-Pods muss eingehenden TCP- und UDP-Port-53-Datenverkehr von der von Ihnen angegebenen Sicherheitsgruppe zulassen.
> - Sie müssen über die erforderlichen eingehenden und ausgehenden Regeln verfügen, um mit anderen Pods kommunizieren zu können, mit denen sie kommunizieren müssen.
> - Sie müssen über Regeln verfügen, die es den Pods ermöglichen, mit der Kubernetes-Steuerebene zu kommunizieren, wenn Sie die Sicherheitsgruppe mit Fargate verwenden. Am einfachsten erreichen Sie dies, indem Sie die Cluster-Sicherheitsgruppe als eine der Sicherheitsgruppen angeben.
>
> Sicherheitsgruppenrichtlinien gelten nur für neu geplante Pods. Sie haben keinen Einfluss auf ausgeführte Pods.

b. Die Richtlinie bereitstellen.

```
kubectl apply -f my-security-group-policy.yaml
```

3. Stellen Sie eine Beispielanwendung mit einem Label bereit, die dem *my-role*-Wert für *podSelector* entspricht, den Sie in einem vorherigen Schritt angegeben haben.

   a. Kopieren Sie den folgenden Inhalt auf Ihr Gerät. Ersetzen Sie die Beispielwerte durch Ihre eigenen, und führen Sie dann den geänderten Befehl aus. Wenn Sie ersetzen*my-role*, stellen Sie sicher, dass es sich um denselben Wert handelt, den Sie in einem vorherigen Schritt für den Selektor angegeben haben.

```
cat >sample-application.yaml <<EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-deployment
  namespace: my-namespace
  labels:
    app: my-app
spec:
  replicas: 4
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
        role: my-role
    spec:
      terminationGracePeriodSeconds: 120
      containers:
      - name: nginx
        image: public.ecr.aws/nginx/nginx:1.23
        ports:
        - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: my-app
  namespace: my-namespace
  labels:
```

```
    app: my-app
spec:
  selector:
    app: my-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
EOF
```

b. Stellen Sie die Anwendung mit dem folgenden Befehl bereit. Wenn Sie die Anwendung bereitstellen, stimmt das Amazon-VPC-CNI-Plugin für Kubernetes mit dem `role`-Label überein und die Sicherheitsgruppen, die Sie im vorherigen Schritt angegeben haben, werden auf den Pod angewendet.

```
kubectl apply -f sample-application.yaml
```

4. Zeigen Sie die Pods an, die mit der Beispielanwendung bereitgestellt wurden. Für den Rest dieses Themas wird dieses Terminal als `TerminalA` bezeichnet.

```
kubectl get pods -n my-namespace -o wide
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME                              READY   STATUS    RESTARTS   AGE     IP
  NODE                                              NOMINATED NODE   READINESS GATES
my-deployment-5df6f7687b-4fbjm    1/1     Running   0          7m51s   192.168.53.48
  ip-192-168-33-28.region-code.compute.internal   <none>           <none>
my-deployment-5df6f7687b-j9fl4    1/1     Running   0          7m51s   192.168.70.145
  ip-192-168-92-33.region-code.compute.internal   <none>           <none>
my-deployment-5df6f7687b-rjxcz    1/1     Running   0          7m51s   192.168.73.207
  ip-192-168-92-33.region-code.compute.internal   <none>           <none>
my-deployment-5df6f7687b-zmb42    1/1     Running   0          7m51s   192.168.63.27
  ip-192-168-33-28.region-code.compute.internal   <none>           <none>
```

> **Note**
>
> Probieren Sie diese Tipps aus, wenn Pods hängen bleiben.
>
> - Wenn Pods im `Waiting`-Stauts hängen bleiben, führen Sie `kubectl describe pod` *`my-deployment-xxxxxxxxxx-xxxxx`* `-n` *`my-namespace`* aus. Wenn

Sie `Insufficient permissions: Unable to create Elastic Network Interface.` sehen, bestätigen Sie, dass Sie die IAM-Richtlinie in einem vorherigen Schritt zur IAM-Cluster-Rolle hinzugefügt haben.

- Wenn Pods im `Pending`-Status hängen bleiben, überprüfen Sie, ob der Instance-Typ Ihres Knotens in limits.go aufgeführt ist, und dass die maximale Anzahl der vom Instance-Typ unterstützten Zweigstellen-Netzwerkschnittstellen multipliziert mit der Anzahl der Knoten in Ihrer Knotengruppe noch nicht erreicht wurde. Eine `m5.large`-Instance unterstützt beispielsweise neun Zweigstellen-Netzwerkschnittstellen. Wenn Ihre Knotengruppe fünf Knoten hat, können maximal 45 Zweigstellen-Netzwerkschnittstellen für die Knotengruppe erstellt werden. Der 46. Pod, den Sie bereitstellen möchten, bleibt im `Pending`-Status, bis ein anderer Pod mit zugehörigen Sicherheitsgruppen gelöscht wird.

Wenn Sie `kubectl describe pod` *`my-deployment-xxxxxxxxxx-xxxxx`* `-n` *`my-namespace`* ausführen und eine Meldung ähnlich der folgenden Meldung angezeigt wird, kann sie problemlos ignoriert werden. Diese Meldung kann angezeigt werden, wenn das Amazon-VPC-CNI-Plugin für Kubernetes versucht, das Host-Netzwerk einzurichten und die Erstellung der Netzwerkschnittstelle fehlschlägt. Das Plug-In protokolliert dieses Ereignis, bis die Netzwerkschnittstelle erstellt wird.

```
Failed to create Pod sandbox: rpc error: code = Unknown desc = failed to set up
 sandbox container "e24268322e55c8185721f52df6493684f6c2c3bf4fd59c9c121fd4cdc894579f"
 network for Pod "my-deployment-5df6f7687b-4fbjm": networkPlugin
cni failed to set up Pod "my-deployment-5df6f7687b-4fbjm-c89wx_my-namespace" network:
 add cmd: failed to assign an IP address to container
```

Die maximale Anzahl von Pods, die auf dem Instance-Typ ausgeführt werden können, darf nicht überschritten werden. Eine Liste der maximalen Anzahl von Pods, die Sie auf jedem Instance-Typ ausführen können, finden Sie unter eni-max-pods.txt on. GitHub Wenn Sie einen Pod mit zugeordneten Sicherheitsgruppen oder den Knoten löschen, auf dem der Pod ausgeführt wird, löscht der VPC-Ressourcen-Controller die Zweigstellen-Netzwerkschnittstelle. Wenn Sie einen Cluster mit Pods löschen, indem Sie Pods für Sicherheitsgruppen verwenden, löscht der Controller die Netzwerkschnittstellen der Zweigstelle nicht, sodass Sie diese selbst löschen müssen. Informationen zum Löschen von Netzwerkschnittstellen finden Sie unter Löschen einer Netzwerkschnittstelle im EC2 Amazon-Benutzerhandbuch.

5. Setzen Sie in einem separaten Terminal eine Shell in einen der Pods ein. Für den Rest dieses Themas wird dieses Terminal als `TerminalB` bezeichnet. *`5df6f7687b-4fbjm`*Ersetzen Sie es durch die ID eines der Pods, die in Ihrer Ausgabe aus dem vorherigen Schritt zurückgegeben wurden.

```
kubectl exec -it -n my-namespace my-deployment-5df6f7687b-4fbjm -- /bin/bash
```

6. Überprüfen Sie in der Shell von `TerminalB`, dass die Beispielanwendung funktioniert.

```
curl my-app
```

   Eine Beispielausgabe sieht wie folgt aus.

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
[...]
```

   Sie haben diese Ausgabe erhalten, weil alle Pods, auf denen die Anwendung ausgeführt wird, der Sicherheitsgruppe zugeordnet sind, die Sie erstellt haben. Diese Gruppe enthält eine Regel, die den gesamten Datenverkehr zwischen allen Pods zulässt, die der Sicherheitsgruppe zugeordnet ist. DNS-Datenverkehr wird von dieser Sicherheitsgruppe an die Cluster-Sicherheitsgruppe übertragen, die mit Ihren Knoten verknüpft ist. Auf den Knoten werden die CoreDNS-Pods ausgeführt, bei denen Ihre Pods eine Namenssuche durchgeführt haben.

7. Entfernen Sie von `TerminalA` aus die Sicherheitsgruppenregeln, die die DNS-Kommunikation mit der Cluster-Sicherheitsgruppe ermöglichen, aus Ihrer Sicherheitsgruppe. Wenn Sie die DNS-Regeln nicht in einem vorherigen Schritt zur Clustersicherheitsgruppe hinzugefügt haben, *`$my_cluster_security_group_id`* ersetzen Sie sie durch die ID der Sicherheitsgruppe, in der Sie die Regeln erstellt haben.

```
aws ec2 revoke-security-group-ingress --group-id $my_cluster_security_group_id --
security-group-rule-ids $my_tcp_rule_id
aws ec2 revoke-security-group-ingress --group-id $my_cluster_security_group_id --
security-group-rule-ids $my_udp_rule_id
```

8. Versuchen Sie über `TerminalB` noch einmal, auf die Anwendung zuzugreifen.

```
curl my-app
```

Eine Beispielausgabe sieht wie folgt aus.

```
curl: (6) Could not resolve host: my-app
```

Der Versuch schlägt fehl, da der Pod nicht mehr auf die CoreDNS-Pods zugreifen kann, denen die Cluster-Sicherheitsgruppe zugeordnet ist. Die Cluster-Sicherheitsgruppe verfügt nicht mehr über die Sicherheitsgruppenregeln, die eine DNS-Kommunikation von der Sicherheitsgruppe, die Ihrem Pod zugeordnet ist, erlauben.

Wenn Sie versuchen, über die IP-Adressen auf die Anwendung zuzugreifen, die für einen der Pods in einem vorherigen Schritt zurückgegeben wurden, erhalten Sie dennoch eine Antwort, da zwischen Pods, denen die Sicherheitsgruppe zugeordnet ist, alle Ports erlaubt sind, und eine Namenssuche daher nicht erforderlich ist.

9. Wenn Sie mit dem Experimentieren fertig sind, können Sie die von Ihnen erstellte Sicherheitsgruppen-Richtlinie, Anwendung und Sicherheitsgruppe entfernen. Führen Sie die folgenden Befehle über `TerminalA` aus.

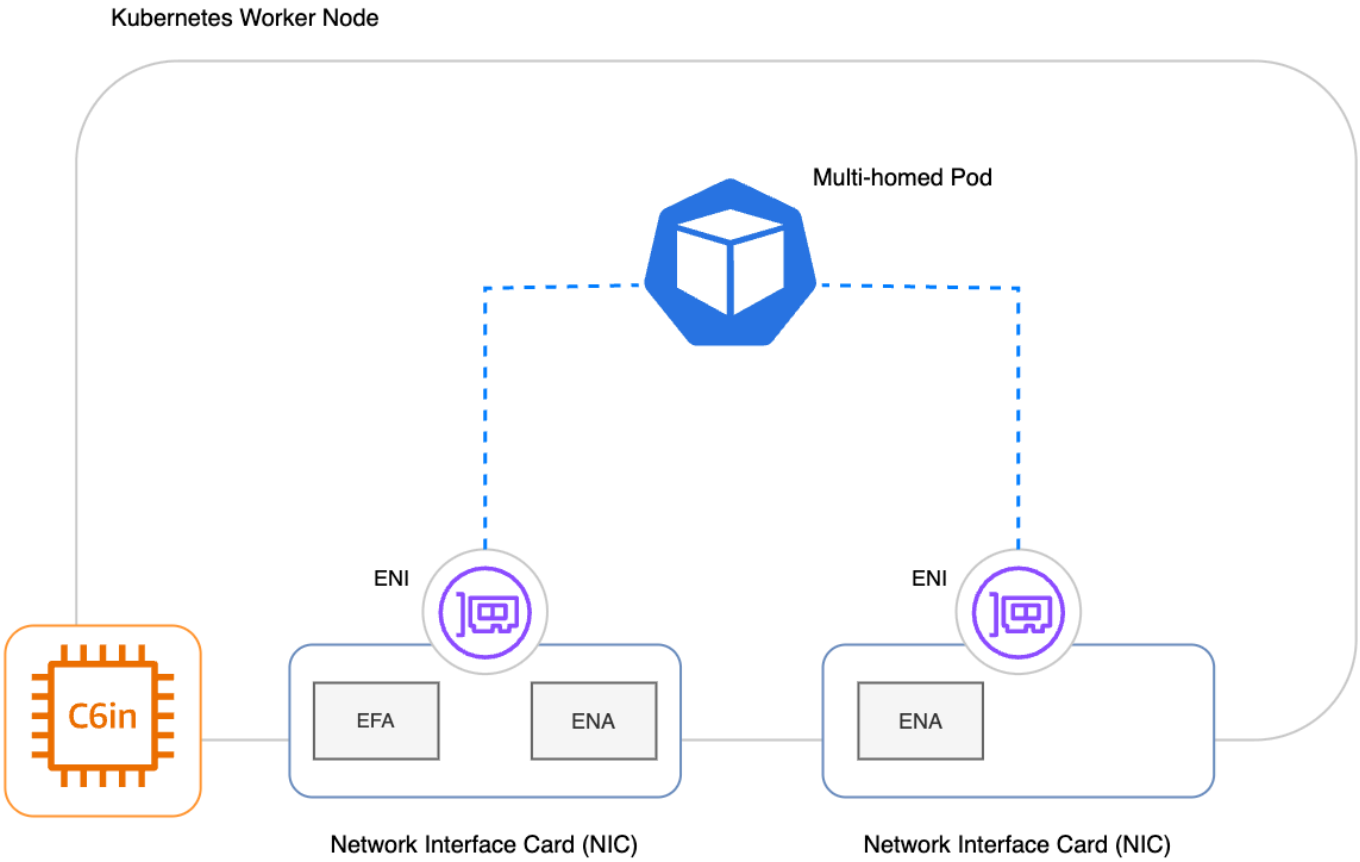
```
kubectl delete namespace my-namespace
aws ec2 revoke-security-group-ingress --group-id $my_pod_security_group_id --
security-group-rule-ids $my_inbound_self_rule_id
wait
sleep 45s
aws ec2 delete-security-group --group-id $my_pod_security_group_id
```

Anfügen mehrerer Netzwerkschnittstellen an Pods

Standardmäßig weist das Amazon-VPC-CNI-Plugin jedem Pod eine IP-Adresse zu. Diese IP-Adresse ist an eine elastische Netzwerkschnittstelle angefügt, die den gesamten ein- und ausgehenden Datenverkehr für den Pod verarbeitet. Um die Bandbreite und die Leistung pro Sekunde zu erhöhen, können Sie das Multi-NIC-Feature des VPC CNI verwenden, um einen Multi-Homed-Pod zu konfigurieren. Ein Multi-Homed-Pod ist ein einzelner Kubernetes-Pod, der mehrere Netzwerkschnittstellen (und mehrere IP-Adressen) verwendet. Durch den Betrieb eines Multi-Homed-Pods können Sie den Anwendungsdatenverkehr mithilfe gleichzeitiger Verbindungen auf mehrere

Netzwerkschnittstellen verteilen. Dies ist besonders nützlich für Anwendungsfälle in den Bereichen Künstliche Intelligenz (KI), Machine Learning (ML) und High Performance Computing (HPC).

Das folgende Diagramm zeigt einen mehrfach vernetzten Pod, der auf einem Worker-Knoten ausgeführt wird und bei dem mehrere Netzwerkschnittstellenkarten () NICs verwendet werden.



Hintergrund

Bei Amazon EC2 ist eine elastic network interface eine logische Netzwerkkomponente in einer VPC, die eine virtuelle Netzwerkkarte darstellt. Bei vielen EC2 Instance-Typen teilen sich die Netzwerkschnittstellen hardwaremäßig eine einzige Netzwerkschnittstellenkarte (NIC). Diese einzelne Netzwerkkarte verfügt über eine maximale Bandbreite und eine maximale Paketrate pro Sekunde.

Wenn das Multi-NIC-Feature aktiviert ist, weist die VPC CNI IP-Adressen nicht wie standardmäßig vorgesehen in großen Mengen zu. Stattdessen weist das VPC CNI bei Bedarf einer Netzwerkschnittstelle auf jeder Netzwerkkarte eine IP-Adresse zu, wenn ein neuer Pod gestartet wird. Dieses Verhalten verringert die Geschwindigkeit, mit der IP-Adressen erschöpft werden, was

durch die Verwendung von Multi-Homed-Pods erhöht wird. Da die VPC CNI IP-Adressen bei Bedarf zuweist, kann der Start von Pods auf Instances mit aktiviertem Multi-NIC-Feature länger dauern.

Überlegungen

- Stellen Sie sicher, dass in Ihrem Kubernetes-Cluster die VPC-CNI-Version `1.20.0` oder höher ausgeführt wird. Das Multi-NIC-Feature ist nur in der Version `1.20.0` VPC CNI oder höher verfügbar.
- Aktivieren Sie die `ENABLE_MULTI_NIC`-Umgebungsvariable im VPC-CNI-Plugin. Sie können den folgenden Befehl ausführen, um die Variable festzulegen und eine Bereitstellung von zu starten DaemonSet.
  - `kubectl set env daemonset aws-node -n kube-system ENABLE_MULTI_NIC=true`
- Stellen Sie sicher, dass Sie Worker-Knoten mit mehreren Netzwerkschnittstellenkarten (NICs) erstellen. Eine Liste der EC2 Instances mit mehreren Netzwerkschnittstellenkarten finden Sie unter Netzwerkkarten im EC2 Amazon-Benutzerhandbuch.
- Wenn das Multi-NIC-Feature aktiviert ist, weist die VPC CNI IP-Adressen nicht wie standardmäßig vorgesehen in großen Mengen zu. Da die VPC CNI IP-Adressen bei Bedarf zuweist, kann der Start von Pods auf Instances mit aktiviertem Multi-NIC-Feature länger dauern. Weitere Informationen finden Sie im vorhergehenden Abschnitt the section called “Hintergrund”.
- Bei aktiviertem Multi-NIC-Feature verfügen Pods standardmäßig nicht über mehrere Netzwerkschnittstellen. Sie müssen jede Workload so konfigurieren, dass sie Multi-NIC verwendet. Fügen Sie die `k8s.amazonaws.com/nicConfig: multi-nic-attachment`-Annotationen zu den Workloads hinzu, die über mehrere Netzwerkschnittstellen verfügen sollen.

Überlegungen zu **`IPv6`**

- Benutzerdefinierte IAM-Richtlinie – Erstellen und verwenden Sie für `IPv6`-Cluster die folgende benutzerdefinierte IAM-Richtlinie für das VPC CNI. Diese Richtlinie gilt speziell für Multi-NICs. Weitere allgemeine Informationen zur Verwendung des VPC CNI mit `IPv6`-Clustern finden Sie unter the section called “IPv6”.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AmazonEKSCNIPolicyIPv6MultiNIC",
            "Effect": "Allow",
```

```
            "Action": [
                "ec2:CreateNetworkInterface",
                "ec2:DescribeInstances",
                "ec2:AssignIpv6Addresses",
                "ec2:DetachNetworkInterface",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeTags",
                "ec2:ModifyNetworkInterfaceAttribute",
                "ec2:DeleteNetworkInterface",
                "ec2:DescribeInstanceTypes",
                "ec2:UnassignIpv6Addresses",
                "ec2:AttachNetworkInterface",
                "ec2:DescribeSubnets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AmazonEKSCNIPolicyENITagIPv6MultiNIC",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*:*:network-interface/*"
        }
    ]
}
```

- IPv6 Übergangsmechanismus nicht verfügbar – Bei Verwendung des Multi-NIC-Feature weist die VPC CNI den Pods in einem IPv6-Cluster keine IPv4-Adresse zu. Andernfalls weist die VPC CNI jedem Pod eine hostlokale IPv4-Adresse zu, damit ein Pod mit externen IPv4-Ressourcen in einer anderen Amazon VPC oder im Internet kommunizieren kann.

Usage

Nachdem das Multi-NIC-Feature in der VPC CNI aktiviert wurde und die aws-node-Pods neu gestartet wurden, können Sie jede Workload so konfigurieren, dass sie Multi-Homed ist. Nachfolgend finden Sie ein Beispiel für eine YAML-Konfiguration mit der erforderlichen Annotation:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: orders-deployment
  namespace: ecommerce
  labels:
```

```
    app: orders
spec:
  replicas: 3
  selector:
    matchLabels:
      app: orders
  template:
    metadata:
      annotations:
         k8s.amazonaws.com/nicConfig: multi-nic-attachment
      labels:
        app: orders
    spec:
...
```

Häufig gestellte Fragen

1. Was ist eine Netzwerkkarte (NIC)?

Eine Netzwerkschnittstellenkarte (NIC), auch einfach Netzwerkkarte genannt, ist ein physisches Gerät, das die Netzwerkkonnektivität für die zugrunde liegende Cloud-Computing-Hardware ermöglicht. Bei modernen EC2 Servern bezieht sich dies auf die Nitro-Netzwerkkarte. Eine Elastic Network Interface (ENI) ist eine virtuelle Darstellung dieser zugrunde liegenden Netzwerkkarte.

Einige EC2 Instance-Typen verfügen über mehrere, NICs um die Bandbreite und die Leistung der Paketrate zu erhöhen. In solchen Fällen können Sie den zusätzlichen Netzwerkkarten sekundäre ENIs Karten zuweisen. Beispielsweise kann ENI Nr. 1 als Schnittstelle für die an den Netzwerkkartenindex 0 angefügten Netzwerkkarte dienen, während ENI Nr. 2 als Schnittstelle für die an einen separaten Netzwerkkartenindex angefügten Netzwerkkarte dienen kann.

2. Was ist ein Multi-Homed-Pod?

Ein Multi-Homed-Pod ist ein einzelner Kubernetes-Pod mit mehreren Netzwerkschnittstellen (und damit mehreren IP-Adressen). Jede Pod-Netzwerkschnittstelle ist mit einem Elastic Network Interface (ENI) verknüpft. Dabei ENIs handelt es sich um logische Repräsentationen von separaten NICs auf dem zugrunde liegenden Worker-Knoten. Durch mehrere Netzwerkschnittstellen verfügt ein Multi-Homed-Pod über zusätzliche Datenübertragungskapazität, was wiederum die Datenübertragungsrate erhöht.

> **Important**
>
> Das VPC CNI kann mehrfach vernetzte Pods nur für Instance-Typen konfigurieren, die mehrere haben. NICs

3. Warum sollte ich dieses Feature nutzen?

Wenn Sie die Netzwerkleistung Ihrer Kubernetes-basierten Workloads skalieren müssen, können Sie die Multi-NIC-Funktion verwenden, um mehrfach vernetzte Pods auszuführen, die eine Schnittstelle zu allen NICs Basisgeräten herstellen, an die ein ENA-Gerät angeschlossen ist. Durch die Nutzung zusätzlicher Netzwerkkarten erhöhen Sie die Bandbreitenkapazität und die Paketratenleistung Ihrer Anwendungen, indem Sie den Anwendungsdatenverkehr auf mehrere gleichzeitige Verbindungen verteilen. Dies ist besonders nützlich für Anwendungsfälle in den Bereichen Künstliche Intelligenz (KI), Machine Learning (ML) und High Performance Computing (HPC).

4. Wie verwende ich dieses Feature?

1. Zunächst müssen Sie sicherstellen, dass Ihr Kubernetes-Cluster VPC CNI Version 1.20 oder höher verwendet. Die Schritte zum Aktualisieren der VPC CNI als EKS-Add-On finden Sie unter the section called “Aktualisierung (EKS-Add-On)”.
2. Anschließend müssen Sie den Multi-NIC-Support in der VPC CNI mithilfe der `ENABLE_MULTI_NIC`-Umgebungsvariablen aktivieren.
3. Danach müssen Sie sicherstellen, dass Sie Knoten mit mehreren Netzwerkkarten erstellen und verbinden. Eine Liste der EC2 Instance-Typen mit mehreren Netzwerkkarten finden Sie unter Netzwerkkarten im EC2 Amazon-Benutzerhandbuch.
4. Abschließend konfigurieren Sie jede Workload so, dass sie entweder mehrere Netzwerkschnittstellen (Multi-Homed-Pods) oder eine einzelne Netzwerkschnittstelle verwendet.

5. Wie konfiguriere ich meine Workloads so, dass sie mehrere NICs auf einem unterstützten Worker-Node verwenden?

Um Multi-Homed-Pods zu verwenden, müssen Sie die folgende Annotation hinzufügen: `k8s.amazonaws.com/nicConfig: multi-nic-attachment`. Dadurch wird eine ENI von jeder NIC in der zugrunde liegenden Instance an den Pod angehängt (Eins-zu-Viele-Zuordnung zwischen einem Pod und dem NICs).

Wenn diese Annotation fehlt, geht das VPC CNI davon aus, dass Ihr Pod nur eine Netzwerkschnittstelle benötigt, und weist ihm eine IP von einem ENI auf einer beliebigen verfügbaren Netzwerkkarte zu.

6. Welche Netzwerkschnittstellen-Adapter werden von diesem Feature unterstützt?

Sie können jeden Netzwerkschnittstellen-Adapter verwenden, wenn Sie mindestens einen ENA für den IP-Datenverkehr an die zugrunde liegende Netzwerkkarte angefügt haben. Weitere Informationen zu ENA finden Sie unter Elastic Network Adapter (ENA) im EC2 Amazon-Benutzerhandbuch.

Unterstützte Netzwerkgeräte-Konfigurationen:

- ENA-Schnittstellen bieten alle herkömmlichen IP-Netzwerk- und Routing-Funktionen, die zur Unterstützung von IP-Netzwerken für eine VPC erforderlich sind. Weitere Informationen finden Sie unter Enhanced Networking with ENA auf Ihren EC2 Instances aktivieren.
- EFA (EFA mit ENA)-Schnittstellen bieten sowohl das ENA-Gerät für IP-Netzwerke als auch das EFA-Gerät für die Kommunikation mit niedriger Latenz und hohem Durchsatz.

> **Important**
>
> Wenn an eine Netzwerkkarte nur-EFA-Adapter angeschlossen ist, wird diese vom VPC CNI bei der Bereitstellung der Netzwerkkonnektivität für einen Multi-Homed-Pod übersprungen. Wenn Sie jedoch einen reinen EFA-Adapter mit einem ENA-Adapter auf einer Netzwerkkarte kombinieren, wird der VPC-CNI auch ENIs auf diesem Gerät verwaltet. Informationen zur Verwendung von Nur-EFA-Schnittstellen mit EKS-Clustern finden Sie unter the section called “Trainings-Cluster mit EFA einrichten”.

7. Kann ich überprüfen, ob ein Knoten in meinem Cluster ENA-Support bietet?

Ja, Sie können die AWS CLI oder EC2 API verwenden, um Netzwerkinformationen über eine EC2 Instance in Ihrem Cluster abzurufen. Dadurch erhalten Sie detaillierte Informationen darüber, ob die Instance ENA-Support bietet. Im folgenden Beispiel ersetzen Sie es `<your-instance-id>` durch die EC2 Instanz-ID eines Knotens.

AWS CLI-Beispiel:

```
aws ec2 describe-instances --instance-ids <your-instance-id> --query
 "Reservations[].Instances[].EnaSupport"
```

Beispielausgabe:

```
[ true ]
```

8. Kann ich die verschiedenen IP-Adressen sehen, die einem Pod zugeordnet sind?

Nein, nicht so einfach. Sie können jedoch `nsenter` vom Knoten aus verwenden, um gängige Netzwerk-Tools wie `ip route show` auszuführen und die zusätzlichen IP-Adressen und Schnittstellen anzuzeigen.

9. Kann ich die Anzahl der Netzwerkschnittstellen für meine Pods steuern?

Nein. Wenn Ihr Workload so konfiguriert ist, dass mehrere NICs auf einer unterstützten Instance verwendet werden, erhält ein einzelner Pod automatisch eine IP-Adresse von jeder Netzwerkkarte auf der Instance. Alternativ verfügen Single-Homed-Pods über eine Netzwerkschnittstelle, die an eine Netzwerkkarte der Instance angefügt ist.

> **Important**
>
> Netzwerkkarten, an die ausschließlich ein nur-EFA-Gerät angeschlossen ist, werden vom VPC CNI übersprungen.

10. Kann ich meine Pods so konfigurieren, dass sie eine bestimmte Netzwerkkarte verwenden?

Nein, dies wird nicht unterstützt. Wenn ein Pod über die entsprechende Annotation verfügt, konfiguriert die VPC CNI ihn automatisch so, dass er jede Netzwerkkarte mit einem ENA-Adapter auf dem Worker-Knoten verwendet.

11. Ist dieses Feature mit den anderen VPC CNI-Netzwerk-Features kompatibel?

Ja, das Multi-NIC-Feature in VPC CNI ist sowohl mit benutzerdefinierten Netzwerken als auch mit der erweiterten Subnetzermittlung kompatibel. Allerdings verwenden die Multi-Homed-Pods keine benutzerdefinierten Subnetze oder Sicherheitsgruppen. Stattdessen weist die VPC CNI den Multi-Homed-Pods IP-Adressen und Netzwerkschnittstellen zu, die dieselbe Subnetz- und Sicherheitsgruppen-Konfiguration wie der Knoten aufweisen. Weitere Informationen zu benutzerdefinierten Netzwerken finden Sie unter the section called “Benutzerdefinierte Netzwerke”.

Das Multi-NIC-Feature in VPC CNI ist nicht mit Sicherheitsgruppen für Pods kompatibel und kann nicht mit diesen kombiniert werden.

12. Kann ich mit diesem Feature Netzwerkrichtlinien verwenden?

Ja, Sie können Kubernetes-Netzwerkrichtlinien mit mehreren Netzwerkkarten verwenden. Kubernetes-Netzwerkrichtlinien beschränken den Netzwerk-Datenverkehr zu und von Ihren Pods. Weitere Informationen zur Anwendung von Netzwerkrichtlinien mit dem VPC CNI finden Sie unter the section called “Kubernetes-Richtlinien”.

13. Ist der Support für mehrere Netzwerkkarten in EKS Auto Mode aktiviert?

Multi-NIC wird für Cluster im EKS Auto Mode nicht unterstützt.

Abweichende CNI-Plugins für Amazon-EKS-Cluster

Das Amazon-VPC-CNI-Plugin für Kubernetes ist das einzige CNI-Plugin, das von Amazon EKS mit Amazon-EC2-Knoten unterstützt wird. Amazon EKS unterstützt die Kernfunktionen von Cilium und Calico für Amazon EKS Hybrid Nodes. Amazon EKS führt Upstream-Kubernetes aus, sodass Sie alternative kompatible CNI-Plugins in Amazon-EC2-Knoten in Ihrem Cluster installieren können. Wenn Sie über Fargate-Knoten in Ihrem Cluster verfügen, ist das Amazon VPC-CNI-Plugin für Kubernetes bereits auf Ihren Fargate-Knoten installiert. Es ist das einzige CNI-Plugin, das Sie mit Fargate-Knoten verwenden können. Ein Versuch, ein alternatives CNI-Plugin auf Fargate-Knoten zu installieren, schlägt fehl.

Wenn Sie beabsichtigen, ein alternatives CNI-Plugin auf Amazon-EC2-Knoten zu verwenden, empfehlen wir Ihnen, sich kommerziellen Support für das Plugin zu holen oder über das interne Fachwissen zu verfügen, um Fehler zu beheben und Korrekturen zum CNI-Plugin-Projekt beizutragen.

Amazon EKS pflegt Beziehungen zu einem Netzwerk von Partnern, die Support für alternative kompatible CNI-Plugins bieten. Einzelheiten zu den Versionen, Qualifizierungen und durchgeführten Tests finden Sie in der folgenden Partnerdokumentation.

Partner	Produkt	Dokumentation
Tigera	Calico	Installationsanleitungen
Isovalent	Cilium	Installationsanleitungen
Juniper	Cloudnatives Contrail Networking (CN2)	Installationsanleitungen

Partner	Produkt	Dokumentation
VMware	Antrea	Installationsanleitungen

Amazon EKS ist bestrebt, Ihnen eine große Auswahl an Optionen zu bieten, um alle Anwendungsfälle abzudecken.

Alternative kompatible Plugins für Netzwerkrichtlinien

Calico ist eine weit verbreitete Lösung für Container-Netzwerke und -Sicherheit. Die Verwendung von Calico in EKS ermöglicht eine vollständig konforme Durchsetzung von Netzwerkrichtlinien für Ihre EKS-Cluster. Darüber hinaus können Sie das Calico-Netzwerk nutzen, das IP-Adressen aus Ihrer zugrunde liegenden VPC speichert. Calico Cloud erweitert die Feature von Calico Open Source und bietet erweiterte Sicherheits- und Beobachtbarkeitsfunktionen.

Der Datenverkehr zu und von Pods mit zugehörigen Sicherheitsgruppen unterliegt nicht der Durchsetzung von Calico-Netzwerkrichtlinien und ist nur auf die Durchsetzung von Amazon VPC-Sicherheitsgruppen beschränkt.

Wenn Sie die Durchsetzung von Calico-Netzwerkrichtlinien verwenden, empfehlen wir Ihnen, die Umgebungsvariable `ANNOTATE_POD_IP` auf `true` festzulegen, um ein bekanntes Problem mit Kubernetes zu vermeiden. Um dieses Feature nutzen zu können, müssen Sie der `aws-node`-ClusterRole die `patch`-Berechtigung für Pods hinzufügen. Beachten Sie, dass das Hinzufügen von Patch-Berechtigungen zum `aws-node` DaemonSet den Sicherheitsbereich für das Plugin erweitert. Weitere Informationen finden Sie unter ANNOTATE_POD_IP im VPC-CNI-Repo auf GitHub.

Überlegungen zu Amazon EKS Auto Mode

Amazon EKS Auto Mode unterstützt keine alternativen CNI-Plugins oder Netzwerkrichtlinien-Plugins. Weitere Informationen finden Sie unter EKS Auto Mode.

Anfügen mehrerer Netzwerkschnittstellen an Pods mit Multus

Multus CNI ist ein Container Network Interface (CNI)-Plugin für Amazon EKS, welches das Anfügen mehrerer Netzwerkschnittstellen an einen Pod ermöglicht. Weitere Informationen finden Sie in der Multus-CNI-Dokumentation auf GitHub.

In Amazon EKS hat jeder Pod eine Netzwerkschnittstelle, die vom Amazon-VPC-CNI-Plug-In zugewiesen wird. Mit Multus können Sie einen mehrfach vernetzten Pod mit mehreren Schnittstellen

erstellen. Dies wird durch Multus erreicht, der als "meta-plugin" fungiert; ein CNI-Plug-In, das mehrere andere CNI-Plug-Ins aufrufen kann. Die AWS-Unterstützung für Multus wird mit dem Amazon-VPC-CNI-Plug-In als Standard-Delegierungs-Plug-In konfiguriert.

- Amazon EKS wird keine CNI-Plugins für Single Root I/O Virtualization (SR-IOV) und Data Plane Development Kit (DPDK) erstellen und veröffentlichen. Sie können jedoch eine Paketbeschleunigung erreichen, indem Sie sich über ein von Multus verwaltetes Hostgerät und `ipvlan`-Plugins direkt mit Amazon EC2 Elastic Network Adapters (ENA) verbinden.
- Amazon EKS unterstützt Multus, das einen generischen Prozess bereitstellt, der eine einfache Verkettung zusätzlicher CNI-Plugins ermöglicht. Multus und der Verkettungsprozess werden unterstützt, aber AWS bietet keinen Support für alle kompatiblen CNI-Plugins, die verkettet werden können, oder Probleme, die in diesen CNI-Plugins auftreten können, die nichts mit der Verkettungskonfiguration zu tun haben.
- Amazon EKS bietet Support und Lebenszyklus-Verwaltung für das Multus-Plugin, ist jedoch nicht für IP-Adressen oder zusätzliche Verwaltung im Zusammenhang mit den zusätzlichen Netzwerkschnittstellen verantwortlich. Die IP-Adresse und Verwaltung der Standard-Netzwerkschnittstelle, die das Amazon-VPC-CNI-Plugin verwendet, bleibt unverändert.
- Nur das Amazon-VPC-CNI-Plugin wird offiziell als Standard-Delegierungs-Plugin unterstützt. Sie müssen das veröffentlichte Multus-Installationsmanifest ändern, um das Standard-Delegierten-Plugin auf ein alternatives CNI umzukonfigurieren, wenn Sie das Amazon VPC-CNI-Plugin nicht für das primäre Netzwerk verwenden möchten.
- Multus wird nur unterstützt, wenn das Amazon-VPC-CNI als primäres CNI verwendet wird. Wir unterstützen das Amazon-VPC-CNI nicht, wenn es für Schnittstellen höherer Ordnung verwendet wird, sekundär oder anderweitig.
- Um zu verhindern, dass das Amazon-VPC-CNI-Plugin versucht, zusätzliche Netzwerkschnittstellen zu verwalten, die Pods zugewiesen sind, können Sie den Netzwerkschnittstellen den folgenden Tag hinzufügen.

  key

  : `node.k8s.amazonaws.com/no_manage`

  Wert

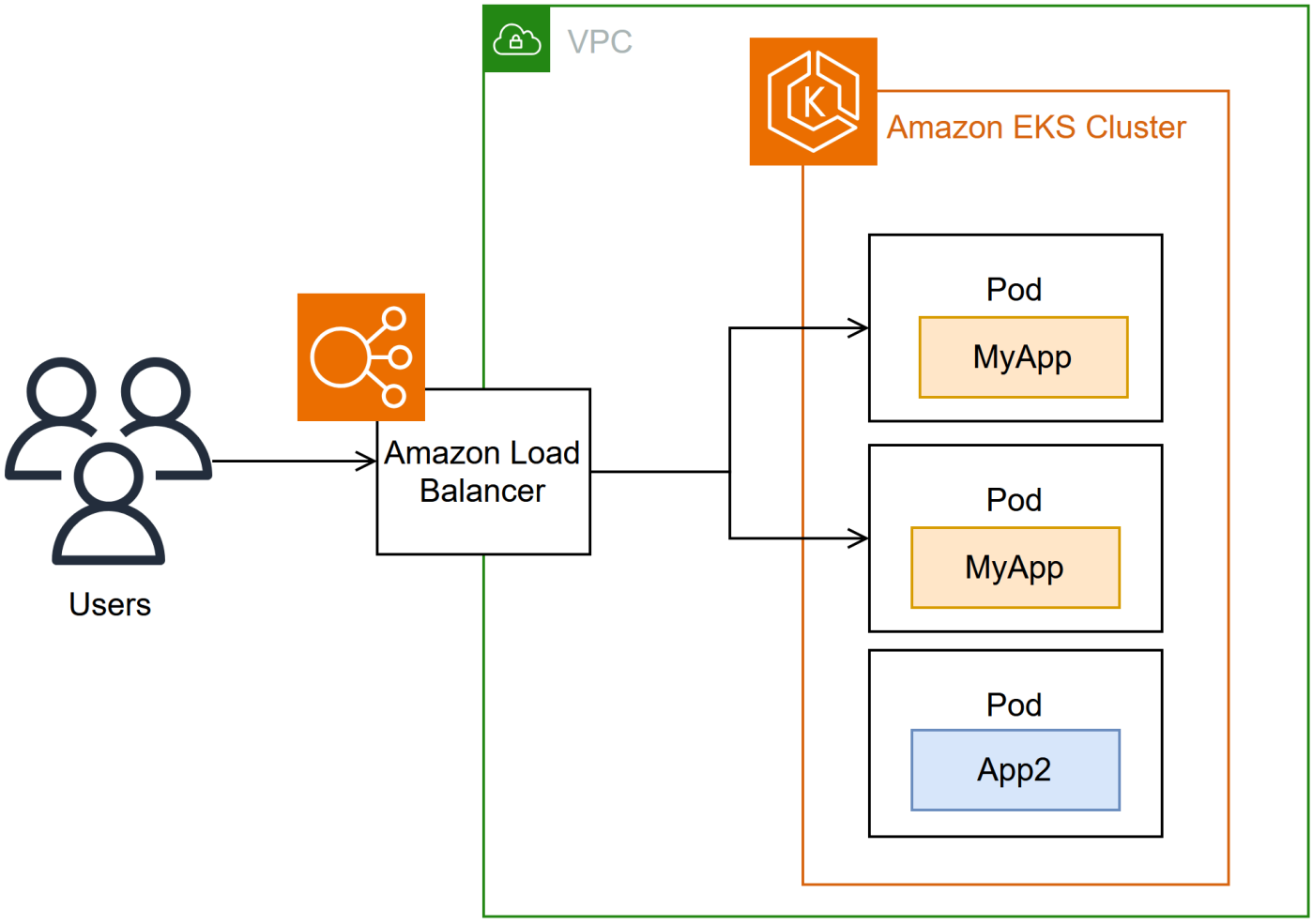
  : `true`
- Multus ist mit Netzwerkrichtlinien kompatibel, die Richtlinie muss jedoch um Ports und IP-Adressen erweitert werden, die Teil von zusätzlichen Netzwerkschnittstellen sein können, die an Pods angeschlossen sind.

Eine Einführung in die Implementierung finden Sie im Multus-Einrichtungsanleitung auf GitHub.

Internetverkehr mit AWS Load Balancer Controller weiterleiten

> **Tip**
>
> Melden Sie sich für bevorstehende Amazon EKS-Workshops an.

Der Load AWS Balancer Controller verwaltet AWS Elastic Load Balancers für einen Kubernetes-Cluster. Mit dem Controller können Sie Ihre Cluster-Anwendungen im Internet verfügbar machen. Der Controller stellt AWS Load Balancer bereit, die auf Cluster-Service- oder Ingress-Ressourcen verweisen. Mit anderen Worten: Der Controller erstellt eine einzelne IP-Adresse oder einen DNS-Namen, der auf mehrere Pods in Ihrem Cluster verweist.



Der Controller überwacht Kubernetes-Ingress- oder Service-Ressourcen. Als Reaktion darauf erstellt es die entsprechenden AWS Elastic Load Balancing Balancing-Ressourcen. Sie können das spezifische Verhalten der Load Balancer konfigurieren, indem Sie Annotationen zu den Kubernetes-Ressourcen hinzufügen. Sie können beispielsweise mithilfe von Anmerkungen AWS Sicherheitsgruppen an Load Balancer anhängen.

Der Controller stellt die folgenden Ressourcen bereit:

Kubernetes **`Ingress`**

Der LBC erstellt einen AWS Application Load Balancer (ALB), wenn Sie einen Kubernetes-`Ingress` erstellen. Überprüfen Sie die Annotationen, die Sie auf eine Ingress-Ressource anwenden können.

Kubernetes-Service vom Typ **`LoadBalancer`**

Der LBC erstellt einen AWS Network Load Balancer (NLB), wenn Sie einen Kubernetes-Service vom Typ `LoadBalancer` erstellen. Überprüfen Sie die Annotationen, die Sie auf eine Service-Ressource anwenden können.

In der Vergangenheit wurde der Kubernetes-Netzwerk-Load-Balancer für Instance-Ziele verwendet, während der LBC für IP-Ziele eingesetzt wurde. Mit der Load AWS Balancer Controller-Version `2.3.0` oder höher können Sie NLBs mit beiden Zieltypen erstellen. Weitere Informationen zu NLB-Zieltypen finden Sie unter Zieltyp im Benutzerhandbuch für Network Load Balancer.

Der Controller ist ein Open-Source-Projekt, auf dem verwaltet wird. GitHub

Bevor Sie den Controller bereitstellen, empfehlen wir Ihnen, die Voraussetzungen und Überlegungen in Weiterleiten von Anwendungen und HTTP-Datenverkehr mit Application Load Balancers und the section called “Netzwerk-Load-Balancer” zu überprüfen. In diesen Themen stellen Sie eine Beispiel-App bereit, die einen AWS Load Balancer enthält.

Kubernetes-API **`Gateway`**

Mit der Load AWS Balancer Controller-Version `2.14.0` oder höher erstellt der LBC einen AWS Application Load Balancer (ALB), wenn Sie ein Kubernetes erstellen. `Gateway` Kubernetes Gateway standardisiert mehr Konfigurationen als Ingress, für das benutzerdefinierte Anmerkungen für viele gängige Optionen erforderlich waren. Überprüfen Sie die Konfiguration,

die Sie auf eine Gateway-Ressource anwenden können. Weitere Informationen zur `Gateway` API finden Sie unter Gateway-API in der Kubernetes-Dokumentation.

Installation des Controllers

Sie können eines der folgenden Verfahren verwenden, um den Load AWS Balancer Controller zu installieren:

- Wenn Sie Amazon EKS noch nicht kennen, empfehlen wir Ihnen, Helm für die Installation zu verwenden, da dies die Installation des Load AWS Balancer Controllers vereinfacht. Weitere Informationen finden Sie unter the section called “Installation mit Helm”.
- Für erweiterte Konfigurationen, wie z. B. Cluster mit eingeschränktem Netzwerkzugriff auf öffentliche Container-Registries, verwenden Sie Kubernetes-Manifeste Weitere Informationen finden Sie unter the section called “Installation mit Manifesten”.

Migration von veralteten Controller-Versionen

- Wenn Sie veraltete Versionen des Load AWS Balancer Controllers installiert haben, finden Sie weitere Informationen unter. the section called “Von veralteter Version migrieren”
- Veraltete Versionen können nicht aktualisiert werden. Sie müssen entfernt und eine aktuelle Version des Load AWS Balancer Controllers installiert werden.
- Zu den veralteten Versionen gehören:
  - AWS ALB Ingress Controller für Kubernetes („Ingress Controller“), ein Vorgänger des Load Balancer Controllers. AWS
  - Beliebige `0.1.`*`x`* Version des AWS Load Balancer Controllers

Veraltete Cloud-Provider

Kubernetes beinhaltet einen veralteten Cloud-Provider für AWS. Der ältere Cloud-Anbieter ist in der Lage, AWS Load Balancer bereitzustellen, ähnlich wie der AWS Load Balancer Controller. Der veraltete Cloud-Provider erstellt klassische Load Balancer. Wenn Sie den Load AWS Balancer Controller nicht installieren, verwendet Kubernetes standardmäßig den alten Cloud-Anbieter. Sie sollten den Load AWS Balancer Controller installieren und vermeiden, den alten Cloud-Anbieter zu verwenden.

Important

In den Versionen 2.5 und neuer wird der Load AWS Balancer Controller mit dem zum Standardcontroller für Kubernetes-Dienstressourcen *`type: LoadBalancer`* und erstellt für jeden Dienst einen AWS Network Load Balancer (NLB). Dies geschieht durch einen mutierenden Webhook für Services, der das Feld `spec.loadBalancerClass` auf `service.k8s.aws/nlb` für neue Services von `type: LoadBalancer` setzt. Sie können dieses Feature deaktivieren und wieder den veralteten Cloud Provider als Standard-Controller verwenden, indem Sie den Wert für den Helm-Chart `enableServiceMutatorWebhook` auf `false` setzen. Der Cluster stellt keine neuen Classic Load Balancer für Ihre Services bereit, es sei denn, Sie schalten dieses Feature aus. Bestehende Classic Load Balancer werden weiterhin funktionieren.

Installieren Sie den AWS Load Balancer Controller mit Helm

Tip

Melden Sie sich für bevorstehende Amazon EKS-Workshops an.

Tip

Mit Amazon EKS Auto Mode ist es nicht erforderlich, Netzwerk-Add-Ons zu installieren oder zu aktualisieren. Der Automatikmodus umfasst Funktionen für Pod-Netzwerke und Load Balancing.
Weitere Informationen finden Sie unter EKS Auto Mode.

In diesem Thema wird beschrieben, wie Sie den Load AWS Balancer Controller mit Helm, einem Paketmanager für Kubernetes, und installieren. `eksctl` Der Controller wird mit den Standardoptionen installiert. Weitere Informationen zum Controller, einschließlich Einzelheiten zur Konfiguration mit Anmerkungen, finden Sie in der Load AWS Balancer Controller-Dokumentation unter. GitHub

Ersetzen Sie in den folgenden Schritten die Beispielwerte durch Ihre eigenen Werte.

Voraussetzungen

Bevor Sie mit diesem Tutorial beginnen, müssen Sie die folgenden Schritte ausführen:

- Erstellen Sie einen Amazon-EKS-Cluster. Informationen zum Erstellen finden Sie unter Erste Schritte.
- Installieren Sie Helm auf Ihrem lokalen Rechner.
- Stellen Sie sicher, dass Ihr Amazon-VPC-CNI-Plugin für Kubernetes, `kube-proxy` und CoreDNS-Add-Ons die in den Servicekonto-Token aufgeführten Mindestversionen aufweist.
- Erfahren Sie mehr über die Konzepte von AWS Elastic Load Balancing. Weitere Informationen finden Sie im Elastic Load Balancing-Benutzerhandbuch.
- Erfahren Sie mehr über den Kubernetes-Service und die Ingress-Ressourcen.

Überlegungen

Bevor Sie mit den Konfigurationsschritten auf dieser Seite fortfahren, beachten Sie Folgendes:

- Die IAM-Richtlinie und die Rolle (`AmazonEKSLoadBalancerControllerRole`) können für mehrere EKS-Cluster im selben AWS Konto wiederverwendet werden.
- Wenn Sie den Controller im selben Cluster installieren, in dem die Rolle (`AmazonEKSLoadBalancerControllerRole`) ursprünglich erstellt wurde, fahren Sie mit Schritt 2: Installieren des Load Balancer Controllers fort, nachdem Sie überprüft haben, ob die Rolle vorhanden ist.
- Wenn Sie IAM-Rollen für Servicekonten (IRSA) verwenden, muss IRSA für jeden Cluster eingerichtet werden, und die OpenID Connect (OIDC)-Anbieter-ARN in der Vertrauensrichtlinie der Rolle ist für jeden EKS-Cluster spezifisch. Wenn Sie den Controller auf einem neuen Cluster mit einem vorhandenen `AmazonEKSLoadBalancerControllerRole` installieren, aktualisieren Sie außerdem die Vertrauensrichtlinie der Rolle, um den OIDC-Anbieter des neuen Clusters einzuschließen, und erstellen Sie ein neues Servicekonto mit der entsprechenden Rollen-Annotation. Informationen zum Feststellen, ob Sie bereits über einen OIDC-Anbieter verfügen oder einen erstellen müssen, finden Sie unter the section called “IAM-OIDC-Anbieter”.

Schritt 1: IAM-Rolle mithilfe von **`eksctl`** erstellen

Die folgenden Schritte beziehen sich auf die Release-Version des Load AWS Balancer Controller v2.14.1. Weitere Informationen zu allen Versionen finden Sie auf der Load AWS Balancer Controller-Release-Seite unter GitHub.

1. Laden Sie eine IAM-Richtlinie für den Load AWS Balancer Controller herunter, mit der er in AWS APIs Ihrem Namen Anrufe tätigen kann.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-
controller/v2.14.1/docs/install/iam_policy.json
```

   - Wenn es sich bei Ihrer Partition um eine nicht standardmäßige AWS Partition handelt, wie z. B. eine Regierung oder eine Region China, lesen Sie die Richtlinien auf GitHub und laden Sie die entsprechende Richtlinie für Ihre Region herunter.

2. Erstellen Sie eine IAM-Richtlinie mit der im vorherigen Schritt heruntergeladenen Richtlinie.

```
aws iam create-policy \
    --policy-name AWSLoadBalancerControllerIAMPolicy \
    --policy-document file://iam_policy.json
```

> **Note**
>
> Wenn Sie sich die Richtlinie in ansehen AWS-Managementkonsole, zeigt die Konsole Warnungen für den ELB-Dienst an, aber nicht für den ELB v2-Dienst. Dies liegt daran, dass einige der Maßnahmen in der Richtlinie für ELB v2, aber nicht für ELB gelten. Sie können diese Warnungen für ELB ignorieren.

3. Ersetzen Sie die Werte für den Cluster-Namen, den Regions-Code und die Konto-ID.

```
eksctl create iamserviceaccount \
    --cluster=<cluster-name> \
    --namespace=kube-system \
    --name=aws-load-balancer-controller \
    --attach-policy-arn=arn:aws: iam::<AWS_ACCOUNT_ID>:policy/
AWSLoadBalancerControllerIAMPolicy \
    --override-existing-serviceaccounts \
    --region <aws-region-code> \
    --approve
```

Schritt 2: AWS Load Balancer Controller installieren

1. Fügen Sie das `eks-charts` Helm-Chart-Repository hinzu. AWS unterhält dieses Repository auf GitHub.

```
helm repo add eks https://aws.github.io/eks-charts
```

2. Aktualisieren Sie Ihr lokales Repository, um sicherzustellen, dass Sie über die neuesten Charts verfügen.

```
helm repo update eks
```

3. Installieren Sie den AWS Load Balancer Controller.

   Wenn Sie den Controller auf EC2 Amazon-Knoten bereitstellen, die eingeschränkten Zugriff auf den Amazon EC2 Instance Metadata Service (IMDS) haben, oder wenn Sie ihn auf Fargate- oder Amazon EKS-Hybridknoten bereitstellen, fügen Sie dem folgenden `helm` Befehl die folgenden Flags hinzu:

   - `--set region=`*`region-code`*
   - `--set vpcId=`*`vpc-xxxxxxxx`*

   Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters. Im folgenden Befehl ist `aws-load-balancer-controller` das Kubernetes-Servicekonto, das Sie in einem vorherigen Schritt erstellt haben.

   Weitere Informationen zur Konfiguration des Helmdiagramms finden Sie unter values.yaml on. GitHub

```
helm install aws-load-balancer-controller eks/aws-load-balancer-controller \
  -n kube-system \
  --set clusterName=my-cluster \
  --set serviceAccount.create=false \
  --set serviceAccount.name=aws-load-balancer-controller \
  --version 1.14.0
```

> **Important**
>
> Das bereitgestellte Diagramm erhält keine automatischen Sicherheitsaktualisierungen. Sie müssen manuell auf ein neueres Diagramm aktualisieren, wenn es verfügbar wird. Wechseln Sie beim Upgrade *`install`* zum vorherigen `upgrade` Befehl.

Der `helm install` Befehl installiert automatisch die benutzerdefinierten Ressourcendefinitionen (CRDs) für den Controller. Der `helm upgrade`-Befehl installiert dies nicht. Wenn Sie das verwendenhelm upgrade,, müssen Sie das manuell installieren CRDs. Führen Sie den folgenden Befehl aus, um das zu installieren CRDs:

```
wget https://raw.githubusercontent.com/aws/eks-charts/master/stable/aws-load-balancer-controller/crds/crds.yaml
kubectl apply -f crds.yaml
```

Schritt 3: Überprüfen, ob der Controller installiert ist

1. Stellen Sie sicher, dass der Controller installiert ist.

   ```
   kubectl get deployment -n kube-system aws-load-balancer-controller
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
   aws-load-balancer-controller   2/2     2            2           84s
   ```

   Sie erhalten die vorherige Ausgabe, wenn Sie mit Helm bereitgestellt haben. Wenn Sie mit dem Kubernetes-Manifest bereitgestellt haben, haben Sie nur ein Replikat.

2. Bevor Sie den Controller zur Bereitstellung von AWS Ressourcen verwenden können, muss Ihr Cluster bestimmte Anforderungen erfüllen. Weitere Informationen erhalten Sie unter the section called “Application Load Balancing” und the section called “Netzwerk-Load-Balancer”.

Installieren Sie den AWS Load Balancer Controller mit Manifesten

> **Tip**
>
> Mit Amazon EKS Auto Mode ist es nicht erforderlich, Netzwerk-Add-Ons zu installieren oder zu aktualisieren. Der Automatikmodus umfasst Funktionen für Pod-Netzwerke und Load Balancing.
> Weitere Informationen finden Sie unter EKS Auto Mode.

In diesem Thema wird beschrieben, wie Sie den Controller durch Herunterladen und Anwenden von Kubernetes-Manifesten installieren. Sie können die vollständige Dokumentation für den Controller auf GitHub anzeigen.

Ersetzen Sie in den folgenden Schritten die Beispielwerte durch Ihre eigenen Werte.

Voraussetzungen

Bevor Sie mit diesem Tutorial beginnen, müssen Sie die folgenden Schritte ausführen:

- Erstellen Sie einen Amazon-EKS-Cluster. Informationen zum Erstellen finden Sie unter Erste Schritte.
- Installieren Sie Helm auf Ihrem lokalen Rechner.
- Stellen Sie sicher, dass Ihr Amazon-VPC-CNI-Plugin für Kubernetes, `kube-proxy` und CoreDNS-Add-Ons die in den Servicekonto-Token aufgeführten Mindestversionen aufweist.
- Erfahren Sie mehr über die Konzepte von AWS Elastic Load Balancing. Weitere Informationen finden Sie im Elastic Load Balancing-Benutzerhandbuch.
- Erfahren Sie mehr über den Kubernetes-Service und die Ingress-Ressourcen.

Überlegungen

Bevor Sie mit den Konfigurationsschritten auf dieser Seite fortfahren, beachten Sie Folgendes:

- Die IAM-Richtlinie und die Rolle (`AmazonEKSLoadBalancerControllerRole`) können für mehrere EKS-Cluster im selben AWS Konto wiederverwendet werden.
- Wenn Sie den Controller auf demselben Cluster installieren, auf dem die Rolle (`AmazonEKSLoadBalancerControllerRole`) ursprünglich erstellt wurde, fahren Sie fort mit Schritt 2: Installation von cert-manager nachdem Sie überprüft haben, dass die Rolle vorhanden ist.
- Wenn Sie IAM-Rollen für Servicekonten (IRSA) verwenden, muss IRSA für jeden Cluster eingerichtet werden, und die OpenID Connect (OIDC)-Anbieter-ARN in der Vertrauensrichtlinie der Rolle ist für jeden EKS-Cluster spezifisch. Wenn Sie den Controller auf einem neuen Cluster mit einem vorhandenen `AmazonEKSLoadBalancerControllerRole` installieren, aktualisieren Sie außerdem die Vertrauensrichtlinie der Rolle, um den OIDC-Anbieter des neuen Clusters einzuschließen, und erstellen Sie ein neues Servicekonto mit der entsprechenden Rollen-Annotation. Informationen zum Feststellen, ob Sie bereits über einen OIDC-Anbieter verfügen oder einen erstellen müssen, finden Sie unter the section called “IAM-OIDC-Anbieter”.

Schritt 1: IAM konfigurieren

Die folgenden Schritte beziehen sich auf die Release-Version des Load AWS Balancer Controller v2.14.1. Weitere Informationen zu allen Versionen finden Sie auf der Load AWS Balancer Controller-Release-Seite unter GitHub.

1. Laden Sie eine IAM-Richtlinie für den Load AWS Balancer Controller herunter, mit der er in AWS APIs Ihrem Namen Anrufe tätigen kann.

   Example

   AWS

   ```
   curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.14.1/docs/install/iam_policy.json
   ```

   AWS GovCloud (US)

   ```
   curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.14.1/docs/install/iam_policy_us-gov.json
   ```

   ```
   mv iam_policy_us-gov.json iam_policy.json
   ```

2. Erstellen Sie eine IAM-Richtlinie mit der im vorherigen Schritt heruntergeladenen Richtlinie.

```
aws iam create-policy \
    --policy-name AWSLoadBalancerControllerIAMPolicy \
    --policy-document file://iam_policy.json
```

> **Note**
>
> Wenn Sie sich die Richtlinie in ansehen AWS-Managementkonsole, zeigt die Konsole Warnungen für den ELB-Dienst an, aber nicht für den ELB v2-Dienst. Dies liegt daran, dass einige der Maßnahmen in der Richtlinie für ELB v2, aber nicht für ELB gelten. Sie können diese Warnungen für ELB ignorieren.

Example

eksctl

a. Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters, *111122223333* durch Ihre Konto-ID und führen Sie dann den Befehl aus.

```
eksctl create iamserviceaccount \
  --cluster=my-cluster \
  --namespace=kube-system \
  --name=aws-load-balancer-controller \
  --role-name AmazonEKSLoadBalancerControllerRole \
  --attach-policy-arn=arn:aws: iam::111122223333:policy/
AWSLoadBalancerControllerIAMPolicy \
  --approve
```

AWS CLI and kubectl

a. Rufen Sie die OIDC-Anbieter-ID Ihres Clusters ab und speichern Sie sie in einer Variablen.

```
oidc_id=$(aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text | cut -d '/' -f 5)
```

b. Stellen Sie fest, ob in Ihrem Konto bereits ein IAM-OIDC-Anbieter mit der ID Ihres Clusters vorhanden ist. Sie müssen OIDC sowohl für den Cluster als auch für IAM konfigurieren.

```
aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
```

Wenn eine Ausgabe zurückgegeben wird, verfügen Sie bereits über einen IAM-OIDC-Anbieter für Ihren Cluster. Wenn keine Ausgabe erfolgt, müssen Sie einen IAM-OIDC-Anbieter für Ihr Cluster erstellen. Weitere Informationen finden Sie unter the section called “IAM-OIDC-Anbieter”.

c. Kopieren Sie den folgenden Inhalt auf Ihr Gerät. Ersetzen Sie *111122223333* durch Ihre Konto-ID. *region-code*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet. Ersetzen Sie *EXAMPLED539D4633E53DE1B71EXAMPLE* mit den im vorherigen Schritt zurückgegebenen Ausgaben.

```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/oidc.eks.us-
east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.us-east-1.amazonaws.com/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
                    "oidc.eks.us-east-1.amazonaws.com/id/
EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-system:aws-
load-balancer-controller"
                }
            }
        }
    ]
}
```

d. Erstellen Sie die IAM-Rolle.

```
aws iam create-role \
  --role-name AmazonEKSLoadBalancerControllerRole \
  --assume-role-policy-document file://"load-balancer-role-trust-policy.json"
```

e. Hängen Sie die erforderliche von Amazon EKS verwaltete IAM-Richtlinie an die IAM-Rolle an. Ersetzen Sie *111122223333* durch Ihre Konto-ID.

```
aws iam attach-role-policy \
  --policy-arn arn:aws: iam::111122223333:policy/
AWSLoadBalancerControllerIAMPolicy \
  --role-name AmazonEKSLoadBalancerControllerRole
```

f. Kopieren Sie den folgenden Inhalt auf Ihr Gerät. Ersetzen Sie *111122223333* durch Ihre Konto-ID. Nachdem Sie den Text ersetzt haben, führen Sie den geänderten Befehl aus, um die Datei `aws-load-balancer-controller-service-account.yaml` zu erstellen.

```
cat >aws-load-balancer-controller-service-account.yaml <<EOF
apiVersion: v1
kind: ServiceAccount
metadata:
```

```
  labels:
    app.kubernetes.io/component: controller
    app.kubernetes.io/name: aws-load-balancer-controller
  name: aws-load-balancer-controller
  namespace: kube-system
  annotations:
    eks.amazonaws.com/role-arn: arn:aws: iam::111122223333:role/
AmazonEKSLoadBalancerControllerRole
EOF
```

g. Erstellen Sie das Kubernetes-Servicekonto in Ihrem Cluster. Das angegebene Kubernetes-Dienstkonto `aws-load-balancer-controller` ist mit der IAM-Rolle versehen, die Sie mit dem Namen erstellt haben. *AmazonEKSLoadBalancerControllerRole*

```
kubectl apply -f aws-load-balancer-controller-service-account.yaml
```

Schritt 2: **`cert-manager`** installieren

Installieren Sie `cert-manager` mit einer der beiden folgenden Methoden, um die Zertifikatskonfiguration in die Webhooks einzufügen. Weitere Informationen finden Sie unter Erste Schritte in der cert-manager-Dokumentation.

Wir empfehlen die Verwendung der `quay.io`-Container-Registry, um `cert-manager` zu installieren. Sollten Ihre Knoten keinen Zugriff auf die `quay.io`-Container-Registry haben, installieren Sie `cert-manager` mithilfe von Amazon ECR (siehe unten).

Example

Quay.io

a. Wenn die Knoten Zugriff auf die `quay.io`-Container-Registry haben, installieren Sie `cert-manager`, um die Zertifikatskonfiguration in die Webhooks einzufügen.

```
kubectl apply \
    --validate=false \
    -f https://github.com/jetstack/cert-manager/releases/download/v1.13.5/cert-
manager.yaml
```

Amazon ECR

a. Installieren Sie `cert-manager` mit einer der beiden folgenden Methoden, um die Zertifikatskonfiguration in die Webhooks einzufügen. Weitere Informationen finden Sie unter Erste Schritte in der cert-manager-Dokumentation.

b. Laden Sie das Manifest herunter.

```
curl -Lo cert-manager.yaml https://github.com/jetstack/cert-manager/releases/
download/v1.13.5/cert-manager.yaml
```

c. Rufen Sie die folgenden Images ab und verschieben Sie sie in ein Repository, auf das die Knoten Zugriff haben. Weitere Informationen zum Abrufen, Markieren und Verschieben der Images in ein eigenes Repository finden Sie unter the section called “Kopieren eines Images in ein Repository”.

```
quay.io/jetstack/cert-manager-cainjector:v1.13.5
quay.io/jetstack/cert-manager-controller:v1.13.5
quay.io/jetstack/cert-manager-webhook:v1.13.5
```

d. Ersetzen Sie `quay.io` im Manifest für die drei Images durch den Namen Ihrer eigenen Registrierung. Beim folgenden Befehl wird davon ausgegangen, dass der Name Ihres privaten Repositorys mit dem des Quell-Repositorys übereinstimmt. Ersetzen Sie *111122223333.dkr.ecr.region-code.amazonaws.com* durch Ihr privates Register.

```
sed -i.bak -e 's|quay.io|111122223333.dkr.ecr.region-code.amazonaws.com|' ./cert-
manager.yaml
```

e. Das Manifest anwenden.

```
kubectl apply \
    --validate=false \
    -f ./cert-manager.yaml
```

Schritt 3: AWS Load Balancer Controller installieren

1. Laden Sie die Controller-Spezifikation herunter. Weitere Informationen zum Controller finden Sie in der Dokumentation auf GitHub.

```
curl -Lo v2_14_1_full.yaml https://github.com/kubernetes-sigs/aws-load-balancer-controller/releases/download/v2.14.1/v2_14_1_full.yaml
```

2. Nehmen Sie die folgenden Änderungen an der Datei vor.

   a. Wenn Sie die Datei `v2_14_1_full.yaml` heruntergeladen haben, führen Sie den folgenden Befehl aus, um den Abschnitt `ServiceAccount` im Manifest zu entfernen. Wenn Sie diesen Abschnitt nicht entfernen, wird die erforderliche Annotation überschrieben, die Sie in einem vorherigen Schritt für das Servicekonto erstellt haben. Außerdem wird durch das Entfernen dieses Abschnitts das in einem vorherigen Schritt erstellte Servicekonto beibehalten, wenn Sie den Controller löschen.

   ```
   sed -i.bak -e '764,772d' ./v2_14_1_full.yaml
   ```

   Wenn Sie eine andere Dateiversion heruntergeladen haben, öffnen Sie die Datei in einem Editor und entfernen Sie die folgenden Zeilen.

   ```
   apiVersion: v1
   kind: ServiceAccount
   metadata:
     labels:
       app.kubernetes.io/component: controller
       app.kubernetes.io/name: aws-load-balancer-controller
     name: aws-load-balancer-controller
     namespace: kube-system
   ---
   ```

   b. Ersetzen Sie `your-cluster-name` im `Deployment spec`-Abschnitt der Datei durch den Namen Ihres Clusters, indem Sie *`my-cluster`* durch den Namen Ihres Cluster ersetzen.

   ```
   sed -i.bak -e 's|your-cluster-name|my-cluster|' ./v2_14_1_full.yaml
   ```

   c. Wenn Ihre Knoten keinen Zugriff auf die Amazon-ECR-Image-Repositorys von Amazon EKS haben, müssen Sie das folgende Image abrufen und in ein Repository verschieben, auf das die Knoten Zugriff haben. Weitere Informationen zum Abrufen, Markieren und Verschieben von Images in ein eigenes Repository finden Sie unter the section called “Kopieren eines Images in ein Repository”.

   ```
   public.ecr.aws/eks/aws-load-balancer-controller:v2.14.1
   ```

Fügen Sie dem Manifest den Namen Ihrer Registry hinzu. Der folgende Befehl geht davon aus, dass der Name Ihres privaten Repositorys mit dem des Quell-Repositorys übereinstimmt, und fügt der Datei den Namen Ihrer privaten Registry hinzu. Ersetzen Sie *111122223333.dkr.ecr.region-code.amazonaws.com* durch Ihr Register. In dieser Zeile wird davon ausgegangen, dass Sie Ihr privates Repository genauso benannt haben wie das Quell-Repository. Andernfalls ändern Sie den Text `eks/aws-load-balancer-controller` nach den Namen Ihrer privaten Registrierung in den Namen Ihres Repositorys.

```
sed -i.bak -e 's|public.ecr.aws/eks/aws-load-balancer-controller|
111122223333.dkr.ecr.region-code.amazonaws.com/eks/aws-load-balancer-
controller|' ./v2_14_1_full.yaml
```

d. (Nur für Fargate oder eingeschränktes IMDS erforderlich)

Wenn Sie den Controller auf EC2 Amazon-Knoten bereitstellen, die eingeschränkten Zugriff auf den Amazon EC2 Instance Metadata Service (IMDS) haben, oder wenn Sie ihn auf Fargate- oder Amazon EKS-Hybridknoten bereitstellen, fügen Sie das `following parameters` unter hinzu. `- args:`

```
[...]
spec:
      containers:
        - args:
            - --cluster-name=your-cluster-name
            - --ingress-class=alb
            - --aws-vpc-id=vpc-xxxxxxxx
            - --aws-region=region-code


[...]
```

3. Wenden Sie die Datei an.

```
kubectl apply -f v2_14_1_full.yaml
```

4. Laden Sie das `IngressClass`- und `IngressClassParams`-Manifest in Ihren Cluster herunter.

```
curl -Lo v2.14.1_ingclass.yaml https://github.com/kubernetes-sigs/aws-load-balancer-
controller/releases/download/v2.14.1/v2_14_1_ingclass.yaml
```

5. Wenden Sie das Manifest auf Ihren Cluster an.

```
kubectl apply -f v2_14_1_ingclass.yaml
```

Schritt 4: Überprüfen, ob der Controller installiert ist

1. Stellen Sie sicher, dass der Controller installiert ist.

```
kubectl get deployment -n kube-system aws-load-balancer-controller
```

   Eine Beispielausgabe sieht wie folgt aus.

```
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
aws-load-balancer-controller   2/2     2            2           84s
```

   Sie erhalten die vorherige Ausgabe, wenn Sie mit Helm bereitgestellt haben. Wenn Sie mit dem Kubernetes-Manifest bereitgestellt haben, haben Sie nur ein Replikat.

2. Bevor Sie den Controller zur Bereitstellung von AWS Ressourcen verwenden können, muss Ihr Cluster bestimmte Anforderungen erfüllen. Weitere Informationen erhalten Sie unter the section called “Application Load Balancing” und the section called “Netzwerk-Load-Balancer”.

Anwendungen vom veralteten ALB Ingress Controller migrieren

In diesem Thema wird die Migration von veralteten Controller-Versionen beschrieben. Insbesondere wird beschrieben, wie veraltete Versionen des AWS Load Balancer Controllers entfernt werden.

- Veraltete Versionen können nicht aktualisiert werden. Sie müssen diese zuerst entfernen und dann eine aktuelle Version installieren.
- Zu den veralteten Versionen gehören:
  - AWS ALB Ingress Controller für Kubernetes („Ingress Controller“), ein Vorgänger des Load Balancer Controllers. AWS
  - Beliebige `0.1.x` Version des AWS Load Balancer Controllers

Veraltete Controller-Version entfernen

> **Note**
>
> Möglicherweise haben Sie die veraltete Version mit Helm oder manuell mit Kubernetes-Manifesten installiert. Führen Sie den Vorgang mit dem Tool aus, mit dem Sie ihn ursprünglich installiert haben.

1. Wenn Sie das `incubator/aws-alb-ingress-controller`-Helm-Chart installiert haben, deinstallieren Sie es.

   ```
   helm delete aws-alb-ingress-controller -n kube-system
   ```

2. Wenn Sie Version `0.1.x` des `eks-charts/aws-load-balancer-controller`-Charts installiert haben, deinstallieren Sie es. Das Upgrade von `0.1.x` auf Version `1.0.0` funktioniert nicht, da die Webhook-API-Version inkompatibel ist.

   ```
   helm delete aws-load-balancer-controller -n kube-system
   ```

3. Überprüfen Sie, ob der Controller derzeit installiert ist.

   ```
   kubectl get deployment -n kube-system alb-ingress-controller
   ```

   Dies ist die Ausgabe, falls der Controller nicht installiert ist.

   ```
   Error from server (NotFound): deployments.apps "alb-ingress-controller" not found
   ```

   Dies ist die Ausgabe, falls der Controller installiert ist.

   ```
   NAME                     READY UP-TO-DATE AVAILABLE AGE
   alb-ingress-controller 1/1   1          1         122d
   ```

4. Geben Sie die folgenden Befehle ein, um den Controller zu entfernen.

   ```
   kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/aws-alb-ingress-controller/v1.1.8/docs/examples/alb-ingress-controller.yaml
   kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/aws-alb-ingress-controller/v1.1.8/docs/examples/rbac-role.yaml
   ```

Zum AWS Load Balancer Controller migrieren

Um vom ALB Ingress Controller für Kubernetes zum Load AWS Balancer Controller zu migrieren, müssen Sie:

1. Entfernen Sie den ALB Ingress Controller (siehe oben).
2. Installieren Sie den AWS Load Balancer Controller.
3. Fügen Sie der vom Load AWS Balancer Controller verwendeten IAM-Rolle eine zusätzliche Richtlinie hinzu. Diese Richtlinie ermöglicht es dem LBC, die vom ALB Ingress Controller für Kubernetes erstellten Ressourcen zu verwalten.
4. Laden Sie die IAM-Richtlinie herunter. Diese Richtlinie ermöglicht es dem Load AWS Balancer Controller, Ressourcen zu verwalten, die vom ALB Ingress Controller für Kubernetes erstellt wurden. Sie können auch die Richtlinie anzeigen.

   ```
   curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-
   controller/v2.14.1/docs/install/iam_policy_v1_to_v2_additional.json
   ```

5. Wenn sich Ihr Cluster in den Regionen AWS GovCloud (USA-Ost) oder AWS GovCloud (US-West) befindet, ersetzen Sie ihn durch. AWS `arn:aws:` `arn:aws-us-gov:`

   ```
   sed -i.bak -e 's|arn:aws:|arn:aws-us-gov:|' iam_policy_v1_to_v2_additional.json
   ```

6. Erstellen Sie die IAM-Richtlinie und notieren Sie den zurückgegebenen ARN.

   ```
   aws iam create-policy \
     --policy-name AWSLoadBalancerControllerAdditionalIAMPolicy \
     --policy-document file://iam_policy_v1_to_v2_additional.json
   ```

7. Hängen Sie die IAM-Richtlinie an die IAM-Rolle an, die vom AWS Load Balancer Controller verwendet wird. *`your-role-name`*Ersetzen Sie durch den Namen der Rolle, z. B. `AmazonEKSLoadBalancerControllerRole`

   Wenn Sie die Rolle mit`eksctl`, erstellt haben, öffnen Sie die AWS CloudFormation Konsole und wählen Sie den eksctl- ***`my-cluster`*** - addon-iamserviceaccount-kube-system - aws-load-balancer-controller Stack aus, um den Rollennamen zu finden, der erstellt wurde. Wählen Sie die Registerkarte für Resources (Ressourcen). Der Rollenname befindet sich in der Spalte Physische ID.

   ```
   aws iam attach-role-policy \
   ```

```
  --role-name your-role-name \
  --policy-arn arn:aws: iam::111122223333:policy/
AWSLoadBalancerControllerAdditionalIAMPolicy
```

CoreDNS für DNS in Amazon EKS-Clustern verwalten

> **Tip**
>
> Mit Amazon EKS Auto Mode ist es nicht erforderlich, Netzwerk-Add-Ons zu installieren oder zu aktualisieren. Der Automatikmodus umfasst Funktionen für Pod-Netzwerke und Load Balancing.
> Weitere Informationen finden Sie unter EKS Auto Mode.

CoreDNS ist ein flexibler, erweiterbarer DNS-Server, der als Kubernetes-Cluster-DNS dienen kann. Wenn Sie einen Amazon-EKS-Cluster mit mindestens einem Knoten starten, werden standardmäßig zwei Replikate des CoreDNS-Image bereitgestellt, unabhängig von der Anzahl der in Ihrem Cluster bereitgestellten Knoten. Die CoreDNS-Pods bieten eine Namensauflösung für alle Pods im Cluster. Die CoreDNS-Pods können in Fargate-Knoten bereitgestellt werden, wenn Ihr Cluster ein Fargate-Profil mit einem Namespace enthält, der mit dem Namespace für CoreDNS `deployment` übereinstimmt. Weitere Informationen zu CoreDNS finden Sie unter Verwenden von CoreDNS für die Serviceerkennung in der Kubernetes-Dokumentation.

CoreDNS-Versionen

In der folgenden Tabelle ist die neueste Version des Amazon-EKS-Add-On-Typs für jede Kubernetes-Version aufgeführt.

Kubernetes-Version	CoreDNS-Version
1.34	v1.12.4-eksbuild.1
1,33	v1.12.4-eksbuild.1
1.32	v1.11.4-eksbuild.24
1.31	v1.11.4-eksbuild.24

Kubernetes-Version	CoreDNS-Version
1,30	v1.11.4-eksbuild.24
1.29	v1.11.4-eksbuild.24
1,28	v1.10.1-eksbuild.38

> **Important**
>
> Wenn Sie dieses Add-On selbst verwalten, stimmen die Versionen in der Tabelle möglicherweise nicht mit den verfügbaren selbstverwalteten Versionen überein. Weitere Hinweise zur Aktualisierung des selbstverwalteten Typs dieses Add-ons finden Sie unter the section called “Aktualisierung (selbstverwaltet)”.

Wichtige Überlegungen zum CoreDNS-Upgrade

- CoreDNS-Updates verwenden a PodDisruptionBudget , um die Verfügbarkeit des DNS-Dienstes während des Aktualisierungsvorgangs aufrechtzuerhalten.
- Um die Stabilität und Verfügbarkeit der CoreDNS-Bereitstellung zu verbessern, werden Versionen `v1.9.3-eksbuild.6` und höher sowie `v1.10.1-eksbuild.3` mit einem `PodDisruptionBudget` bereitgestellt. Wenn Sie ein vorhandenes `PodDisruptionBudget` bereitgestellt haben, schlägt Ihr Upgrade auf diese Versionen möglicherweise fehl. Schlägt das Upgrade fehl, sollte das Problem durch Ausführen einer der folgenden Aufgaben gelöst werden:
  - Wenn Sie das Upgrade des Amazon-EKS-Add-ons durchführen, entscheiden Sie sich dafür, die vorhandenen Einstellungen als Konfliktlösungsoption zu überschreiben. Wenn Sie weitere benutzerdefinierte Einstellungen für die Bereitstellung vorgenommen haben, stellen Sie sicher, dass Sie Ihre Einstellungen vor dem Upgrade sichern, damit Sie Ihre anderen benutzerdefinierten Einstellungen nach dem Upgrade erneut anwenden können.
  - Entfernen Sie Ihr vorhandenes `PodDisruptionBudget` und versuchen Sie das Upgrade erneut.
- In EKS-Add-On-Versionen `v1.9.3-eksbuild.3` und höher sowie `v1.10.1-eksbuild.6` und höher legt die CoreDNS-Bereitstellung `readinessProbe` für die Verwendung des `/ready`-Endpunkts fest. Dieser Endpunkt ist in der `Corefile`-Konfigurationsdatei für CoreDNS aktiviert.

Wenn Sie eine benutzerdefinierte `Corefile` verwenden, müssen Sie das `ready`-Plugin zur Konfiguration hinzufügen, sodass der `/ready`-Endpunkt in CoreDNS aktiv ist, damit er von der Probe verwendet werden kann.

- In EKS-Add-On-Versionen `v1.9.3-eksbuild.7` und höher sowie `v1.10.1-eksbuild.4` und höher können Sie das `PodDisruptionBudget` ändern. Sie können das Add-on bearbeiten und diese Einstellungen in den optionalen Konfigurationseinstellungen mithilfe der Felder im folgenden Beispiel ändern. Dieses Beispiel zeigt das Standard-`PodDisruptionBudget`.

```
{
    "podDisruptionBudget": {
        "enabled": true,
        "maxUnavailable": 1
        }
}
```

Sie können `maxUnavailable` oder `minAvailable` festlegen, aber Sie können nicht beide gleichzeitig in einem einzelnen `PodDisruptionBudget` festlegen. Weitere Informationen zu finden Sie unter `PodDisruptionBudgets` Specifying a PodDisruptionBudget in der Kubernetes-Dokumentation.

Beachten Sie, dass, wenn Sie `enabled` auf `false` festlegen, das `PodDisruptionBudget` nicht entfernt wird. Nachdem Sie dieses Feld auf `false` gesetzt haben, müssen Sie das `PodDisruptionBudget`-Objekt löschen. Ähnlich verhält es sich, wenn Sie das Add-On bearbeiten, um nach dem Upgrade auf eine Version mit einem `PodDisruptionBudget` eine ältere Version des Add-Ons zu verwenden (das Add-On herunterstufen). Das `PodDisruptionBudget` wird nicht entfernt. Führen Sie den folgenden Befehl aus, um das `PodDisruptionBudget` zu löschen:

```
kubectl delete poddisruptionbudget coredns -n kube-system
```

- Ändern Sie in EKS-Add-On-Versionen `v1.10.1-eksbuild.5` und höher die Standardtoleranz von `node-role.kubernetes.io/master:NoSchedule` auf `node-role.kubernetes.io/control-plane:NoSchedule`, um sie an KEP 2067 anzupassen. Weitere Informationen zu KEP 2067 finden Sie unter KEP-2067: Rename the kubeadm „master“ -Label and taint in den Kubernetes Enhancement Proposals () on. KEPs GitHub

In den EKS-Add-On-Versionen `v1.8.7-eksbuild.8` und höher `v1.9.3-eksbuild.9` und höher sind beide Toleranzen so eingestellt, dass sie mit jeder Kubernetes-Version kompatibel sind.

- In den EKS-Add-On-Versionen `v1.9.3-eksbuild.11` und `v1.10.1-eksbuild.7` und höher legt die CoreDNS-Bereitstellung einen Standardwert für `topologySpreadConstraints` fest. Der Standardwert stellt sicher, dass die CoreDNS-Pods über die Availability Zones verteilt werden, wenn Knoten in mehreren Availability Zones verfügbar sind. Sie können einen benutzerdefinierten Wert festlegen, der anstelle des Standardwerts verwendet wird. Der Standardwert lautet wie folgt:

```
topologySpreadConstraints:
  - maxSkew: 1
    topologyKey: topology.kubernetes.io/zone
    whenUnsatisfiable: ScheduleAnyway
    labelSelector:
      matchLabels:
        k8s-app: kube-dns
```

Überlegungen zum CoreDNS **`v1.11`** Upgrade

- In EKS-Add-On-Versionen `v1.11.1-eksbuild.4` und höher basiert das Container-Image auf einem minimalen Basis-Image, das von Amazon EKS Distro verwaltet wird, minimale Pakete enthält und keine Shells hat. Weitere Informationen finden Sie unter Amazon-EKS-Distro. Die Verwendung und Fehlerbehebung des CoreDNS-Images bleibt unverändert.

Erstellen des CoreDNS-Amazon-EKS-Add-Ons

Erstellen Sie das CoreDNS-Amazon-EKS-Add-On. Sie müssen über einen Cluster verfügen, bevor Sie das Add-On erstellen. Weitere Informationen finden Sie unter the section called “Erstellen eines Clusters”.

1. Sehen Sie, welche Version des Container-Images derzeit auf Ihrem Cluster installiert ist.

```
kubectl describe deployment coredns --namespace kube-system | grep coredns: | cut -
d : -f 3
```

   Eine Beispielausgabe sieht wie folgt aus.

```
v1.10.1-eksbuild.13
```

2. Sehen Sie, welche Version des Container-Images derzeit auf Ihrem Cluster installiert ist. Je nachdem, mit welchem Tool Sie Ihr Cluster erstellt haben, ist der Add-on vom Typ Amazon EKS möglicherweise derzeit nicht auf Ihrem Cluster installiert. Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
    addon.addonVersion --output text
   ```

   Wenn eine Versionsnummer zurückgegeben wird, ist das Add-On vom Typ Amazon EKS auf Ihrem Cluster installiert und Sie müssen die restlichen Schritte in diesem Verfahren nicht ausführen. Wenn ein Fehler zurückgegeben wird, ist der Amazon-EKS-Typ des Add-Ons nicht auf Ihrem Cluster installiert. Führen Sie die verbleibenden Schritte dieses Prozesses aus, um es zu installieren.

3. Speichern Sie die Konfiguration Ihres aktuell installierten Add-ons ab.

   ```
   kubectl get deployment coredns -n kube-system -o yaml > aws-k8s-coredns-old.yaml
   ```

4. Erstellen Sie das Add-On mithilfe der AWS-CLI. Wenn Sie mit AWS-Managementkonsole oder `eksctl` das Add-on erstellen möchten, gehen Sie zu the section called “Add-On erstellen” und geben Sie `coredns` für den Namen des Add-ons an. Kopieren Sie den folgenden Befehl auf Ihr Gerät. Nehmen Sie bei Bedarf die folgenden Änderungen am Befehl vor, und führen Sie dann den geänderten Befehl aus.

   - Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters.
   - Ersetzen Sie *v1.11.3-eksbuild.1* durch die aktuelle Version, die in der aktuellen Versionstabelle für Ihre Cluster-Version aufgeführt ist.

     ```
     aws eks create-addon --cluster-name my-cluster --addon-name coredns --addon-version
      v1.11.3-eksbuild.1
     ```

     Wenn Sie benutzerdefinierte Einstellungen auf Ihr aktuelles Add-on angewendet haben, die mit den Standardeinstellungen des Amazon-EKS-Add-Ons in Konflikt stehen, schlägt die Erstellung möglicherweise fehl. Wenn die Erstellung fehlschlägt, erhalten Sie eine Fehlermeldung, die Sie bei der Problembehebung unterstützt. Alternativ können Sie den vorherigen Befehl mit `--resolve-conflicts OVERWRITE` ergänzen. Dadurch kann das Add-on alle vorhandenen

benutzerdefinierten Einstellungen überschreiben. Sobald Sie das Add-On erstellt haben, können Sie es mit Ihren benutzerdefinierten Einstellungen aktualisieren.

5. Überprüfen Sie, ob die neueste Version des Add-Ons für die Kubernetes-Version Ihres Clusters zu Ihrem Cluster hinzugefügt wurde. Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 addon.addonVersion --output text
```

Es kann einige Sekunden dauern, bis die Erstellung des Add-ons abgeschlossen ist.

Eine Beispielausgabe sieht wie folgt aus.

```
v1.11.3-eksbuild.1
```

6. Wenn Sie benutzerdefinierte Einstellungen für Ihr ursprüngliches Add-on vorgenommen haben, bevor Sie das Add-on vom Typ Amazon EKS erstellt haben, verwenden Sie die Konfiguration, die Sie in einem vorherigen Schritt gespeichert haben, um das Add-on vom Typ Amazon EKS mit Ihren benutzerdefinierten Einstellungen zu aktualisieren. Anweisungen zum Aktualisieren des Add-Ons finden Sie unter the section called “Aktualisierung (EKS-Add-On)”.

Aktualisierung des CoreDNS-Amazon-EKS-Add-Ons

Erstellen Sie das Add-on vom Typ Amazon EKS. Wenn Sie das Amazon-EKS-Add-On nicht zu Ihrem Cluster hinzugefügt haben, fügen Sie es entweder hinzu oder lesen Sie the section called “Aktualisierung (selbstverwaltet)”.

Bevor Sie beginnen, lesen Sie die Hinweise zum Upgrade. Weitere Informationen finden Sie unter the section called “Wichtige Überlegungen zum CoreDNS-Upgrade”.

1. Sehen Sie, welche Version des Container-Images derzeit auf Ihrem Cluster installiert ist. Ersetzen Sie den *cluster-name* durch Ihren Cluster-Namen.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 "addon.addonVersion" --output text
```

Eine Beispielausgabe sieht wie folgt aus.

```
v1.10.1-eksbuild.13
```

Wenn die zurückgegebene Version mit der Version für die Kubernetes-Version Ihres Clusters in der aktuellen Versionstabelle übereinstimmt, haben Sie die neueste Version bereits auf Ihrem Cluster installiert und müssen den Rest dieses Verfahrens nicht abschließen. Wenn Sie eine Fehlermeldung statt einer Versionsnummer in Ihrer Ausgabe erhalten, haben Sie den Add-On vom Typ Amazon EKS nicht auf Ihrem Cluster installiert. Sie müssen das Add-on erstellen, bevor Sie es mit diesem Verfahren aktualisieren können.

2. Speichern Sie die Konfiguration Ihres aktuell installierten Add-ons ab.

```
kubectl get deployment coredns -n kube-system -o yaml > aws-k8s-coredns-old.yaml
```

3. Aktualisieren Sie Ihr Add-On mithilfe der AWS CLI. Informationen dazu, wie Sie mit AWS-Managementkonsole oder `eksctl` das Add-on aktualisieren, finden Sie unter the section called "Add-On aktualisieren". Kopieren Sie den folgenden Befehl auf Ihr Gerät. Nehmen Sie bei Bedarf die folgenden Änderungen am Befehl vor, und führen Sie dann den geänderten Befehl aus.
   - Ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters.
   - Ersetzen Sie *`v1.11.3-eksbuild.1`* durch die aktuelle Version, die in der aktuellen Versionstabelle für Ihre Cluster-Version aufgeführt ist.
   - Die `--resolve-conflicts`*`PRESERVE`* -Option behält die vorhandenen Konfigurationswerte für das Add-On bei. Wenn Sie benutzerdefinierte Werte für Zusatzeinstellungen festgelegt haben und diese Option nicht verwenden, überschreibt Amazon EKS Ihre Werte mit seinen Standardwerten. Wenn Sie diese Option verwenden, empfehlen wir, dass Sie alle Feld- und Wertänderungen auf einem Nicht-Produktionscluster testen, bevor Sie das Add-on auf Ihrem Produktionscluster aktualisieren. Wenn Sie diesen Wert auf `OVERWRITE` ändern, werden alle Einstellungen auf die Amazon-EKS-Standardwerte geändert. Wenn Sie benutzerdefinierte Werte für Einstellungen festgelegt haben, werden diese möglicherweise mit den Amazon-EKS-Standardwerten überschrieben. Wenn Sie diesen Wert auf `none` ändern, ändert Amazon EKS den Wert der Einstellungen nicht, aber die Aktualisierung schlägt möglicherweise fehl. Wenn das Update fehlschlägt, erhalten Sie eine Fehlermeldung, die Sie bei der Behebung des Konflikts unterstützt.
   - Wenn Sie eine Konfigurationseinstellung nicht aktualisieren, entfernen Sie `--configuration-values '{`*`"replicaCount":3`*`}'` aus dem Befehl. Wenn Sie eine Konfigurationseinstellung aktualisieren, ersetzen Sie *`"replicaCount":3`* durch die Einstellung, die Sie festlegen möchten. In diesem Beispiel ist die Anzahl der Replikate von CoreDNS auf `3` festgelegt. Der von Ihnen angegebene Wert muss für das Konfigurationsschema gültig sein. Falls Sie das Konfigurationsschema nicht kennen, führen Sie `aws eks describe-addon-`

`configuration --addon-name coredns --addon-version` *`v1.11.3-eksbuild.1`* aus und ersetzen Sie dabei *`v1.11.3-eksbuild.1`* durch die Versionsnummer des Add-Ons, für das Sie die Konfiguration anzeigen möchten. Das Schema wird in der Ausgabe zurückgegeben. Wenn Sie bereits eine benutzerdefinierte Konfiguration haben, diese komplett entfernen und die Werte für alle Einstellungen auf die Amazon EKS-Standardwerte zurücksetzen möchten, entfernen Sie *`"replicaCount":3`* aus dem Befehl, sodass {} leer ist. Weitere Informationen zu den CoreDNS-Einstellungen finden Sie unter Anpassung des DNS-Services in der Kubernetes-Dokumentation.

```
aws eks update-addon --cluster-name my-cluster --addon-name coredns --addon-version
 v1.11.3-eksbuild.1 \
    --resolve-conflicts PRESERVE --configuration-values '{"replicaCount":3}'
```

Es kann einige Sekunden dauern, bis das Update abgeschlossen ist.

4. Vergewissern Sie sich, dass die Add-on-Version aktualisiert wurde. Ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns
```

Es kann einige Sekunden dauern, bis das Update abgeschlossen ist.

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "addon": {
        "addonName": "coredns",
        "clusterName": "my-cluster",
        "status": "ACTIVE",
        "addonVersion": "v1.11.3-eksbuild.1",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:region:111122223333:addon/my-cluster/coredns/
d2c34f06-1111-2222-1eb0-24f64ce37fa4",
        "createdAt": "2023-03-01T16:41:32.442000+00:00",
        "modifiedAt": "2023-03-01T18:16:54.332000+00:00",
        "tags": {},
        "configurationValues": "{\"replicaCount\":3}"
    }
```

```
}
```

Aktualisierung des selbstverwalteten CoreDNS-Amazon-EKS-Add-Ons

> **Important**
>
> Wir empfehlen, den Amazon-EKS-Typ des Add-Ons zu Ihrem Cluster hinzuzufügen, anstatt den selbstverwalteten Typ des Add-Ons zu verwenden. Wenn Sie noch keine Erfahrung mit den Unterschieden zwischen den Typen haben, finden Sie weitere Informationen unter the section called “Amazon-EKS-Add-ons”. Weitere Informationen zum Hinzufügen eines Amazon-EKS-Add-ons zu Ihrem Cluster finden Sie unter the section called “Add-On erstellen”. Wenn Sie das Amazon-EKS-Add-On nicht verwenden können, empfehlen wir Ihnen, ein Problem zu den Gründen, warum Sie es nicht verwenden können, an das GitHub-Repository Containers Roadmap zu senden.

Bevor Sie beginnen, lesen Sie die Hinweise zum Upgrade. Weitere Informationen finden Sie unter the section called “Wichtige Überlegungen zum CoreDNS-Upgrade”.

1. Vergewissern Sie sich, dass Sie das selbstverwaltete Add-On auf Ihrem Cluster installiert haben. Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
    addon.addonVersion --output text
   ```

   Wenn Sie eine Fehlermeldung erhalten, wird das Add-On als selbstverwaltetes Add-On auf Ihrem Cluster installiert. Führen Sie die verbleibenden Schritte in diesem Verfahren aus. Wenn Sie eine Versionsnummer zurückgeben, wird der Amazon-EKS-Typ des Add-Ons auf Ihrem Cluster installiert. Um den Amazon-EKS-Typ des Add-Ons zu aktualisieren, befolgen Sie die Schritte unter Aktualisierung des CoreDNS-Amazon-EKS-Add-Ons anstelle der hier beschriebenen Vorgehensweise. Wenn Sie mit den Unterschieden zwischen den Add-On-Typen nicht vertraut sind, finden Sie Informationen unter the section called “Amazon-EKS-Add-ons”.
2. Sehen Sie, welche Version des Container-Images derzeit auf Ihrem Cluster installiert ist.

   ```
   kubectl describe deployment coredns -n kube-system | grep Image | cut -d ":" -f 3
   ```

   Eine Beispielausgabe sieht wie folgt aus.

```
v1.8.7-eksbuild.2
```

3. Wenn Ihre aktuelle CoreDNS-Version `v1.5.0` oder höher ist, aber älter als die in der Tabelle der CoreDNS-Versionen aufgeführte Version ist, überspringen Sie diesen Schritt. Wenn Ihre aktuelle Version älter als `1.5.0` ist, müssen Sie die `ConfigMap` für CoreDNS ändern, um das Weiterleitungs-Add-On anstelle des Proxy-Add-Ons zu verwenden.

   a. Öffnen Sie `ConfigMap` mit dem folgenden Befehl.

   ```
   kubectl edit configmap coredns -n kube-system
   ```

   b. Ersetzen Sie `proxy` in der folgenden Zeile durch `forward`. Speichern Sie die Datei und schließen Sie den Editor.

   ```
   proxy . /etc/resolv.conf
   ```

4. Wenn Sie Ihren Cluster ursprünglich auf Kubernetes `1.17` oder früher bereitgestellt haben, müssen Sie möglicherweise eine nicht mehr verwendete Zeile aus Ihrem CoreDNS-Manifest entfernen.

   > **Important**
   >
   > Sie müssen diesen Schritt vor dem Update auf die CoreDNS-Version `1.7.0` abschließen. Es wird jedoch empfohlen, diesen Schritt auch dann durchzuführen, wenn Sie auf eine frühere Version aktualisieren.

   a. Überprüfen Sie, ob Ihr CoreDNS-Manifest die Zeile enthält.

   ```
   kubectl get configmap coredns -n kube-system -o jsonpath='{$.data.Corefile}' |
    grep upstream
   ```

   Wenn keine Ausgabe zurückgegeben wird, enthält Ihr Manifest die Zeile nicht und Sie können mit dem nächsten Schritt fortfahren, um CoreDNS zu aktualisieren. Wenn eine Ausgabe zurückgegeben wird, müssen Sie die Zeile entfernen.

   b. Bearbeiten Sie das `ConfigMap` mit dem folgenden Befehl und entfernen Sie die Zeile in der Datei, die das Wort `upstream` enthält. Ändern Sie sonst nichts in der Datei. Nachdem die Linie entfernt wurde, speichern Sie die Änderungen.

```
kubectl edit configmap coredns -n kube-system -o yaml
```

5. Rufen Sie Ihre aktuelle CoreDNS-Image-Version ab:

```
kubectl describe deployment coredns -n kube-system | grep Image
```

Eine Beispielausgabe sieht wie folgt aus.

```
602401143452.dkr.ecr.region-code.amazonaws.com/eks/coredns:v1.8.7-eksbuild.2
```

6. Wenn Sie auf CoreDNS `1.8.3` aktualisieren, müssen Sie die `system:coredns`-Berechtigung zu `endpointslices` Kubernetes `clusterrole` hinzufügen.

```
kubectl edit clusterrole system:coredns -n kube-system
```

Fügen Sie die folgenden Zeilen unter den vorhandenen Berechtigungszeilen im `rules`-Abschnitt der Datei hinzu.

```
[...]
- apiGroups:
  - discovery.k8s.io
  resources:
  - endpointslices
  verbs:
  - list
  - watch
[...]
```

7. Aktualisieren Sie das CoreDNS-Add-On, indem Sie *602401143452* und den *region-code* durch die Werte aus der im vorherigen Schritt zurückgegebenen Ausgabe ersetzen. Ersetzen Sie *v1.11.3-eksbuild.1* durch die CoreDNS-Version, die in der Tabelle mit den neuesten Versionen für Ihre Kubernetes-Version aufgeführt ist.

```
kubectl set image deployment.apps/coredns -n kube-system
 coredns=602401143452.dkr.ecr.region-code.amazonaws.com/eks/coredns:v1.11.3-
eksbuild.1
```

Eine Beispielausgabe sieht wie folgt aus.

```
deployment.apps/coredns image updated
```

8. Überprüfen Sie die Container-Image-Version erneut, um sicherzustellen, dass sie auf die Version aktualisiert wurde, die Sie im vorherigen Schritt angegeben haben.

```
kubectl describe deployment coredns -n kube-system | grep Image | cut -d ":" -f 3
```

Eine Beispielausgabe sieht wie folgt aus.

```
v1.11.3-eksbuild.1
```

Skalierung von CoreDNS-Pods für hohen DNS-Datenverkehr

Wenn Sie einen Amazon-EKS-Cluster mit mindestens einem Knoten starten, wird standardmäßig eine Bereitstellung von zwei Replikaten des CoreDNS-Images bereitgestellt, unabhängig von der Anzahl der in Ihrem Cluster bereitgestellten Knoten. Die CoreDNS-Pods bieten eine Namensauflösung für alle Pods im Cluster. Anwendungen verwenden die Namensauflösung, um eine Verbindung zu Pods und Services im Cluster sowie zu Services außerhalb des Clusters herzustellen. Wenn die Anzahl der Anfragen zur Namensauflösung (Abfragen) von Pods zunimmt, können die CoreDNS-Pods überlastet werden. Dadurch verlangsamt sich ihre Leistung und sie lehnen Anfragen ab, die sie nicht verarbeiten können.

Um die erhöhte Auslastung der CoreDNS-Pods zu bewältigen, empfehlen wir die Implementierung eines Autoscaling-Systems für CoreDNS. Amazon EKS kann das Autoscaling der CoreDNS-Bereitstellung in der EKS-Add-On-Version von CoreDNS verwalten. Dieser CoreDNS-Autoscaler überwacht kontinuierlich den Status des Clusters, einschließlich der Anzahl der Knoten und CPU-Kerne. Anhand dieser Informationen passt der Controller die Anzahl der Replikate der CoreDNS-Bereitstellung in einem EKS-Cluster dynamisch an. Dieses Feature ist für CoreDNS `v1.9` und höher verfügbar. Weitere Informationen darüber, welche Versionen mit CoreDNS Autoscaling kompatibel sind, finden Sie im folgenden Abschnitt

Das System verwaltet CoreDNS-Replikate automatisch anhand einer dynamischen Formel, die sowohl auf der Anzahl der Knoten als auch auf der Anzahl der CPU-Kerne im Cluster basiert und als Maximum von (Anzahl der Knoten geteilt durch 16) und (Anzahl der CPU-Kerne geteilt durch 256) berechnet wird. Es bewertet die Nachfrage über 10-minütige Spitzenzeiten aus und skaliert bei Bedarf sofort hoch, um eine erhöhte DNS-Abfragelast zu bewältigen. Gleichzeitig

skaliert es schrittweise herunter, indem es die Replikate alle 3 Minuten um 33 % reduziert, um die Systemstabilität aufrechtzuerhalten und Störungen zu vermeiden.

Wir empfehlen, dieses Feature in Verbindung mit anderen bewährten Methoden für die automatische Skalierung von EKS-Clustern zu verwenden, um die allgemeine Anwendungsverfügbarkeit und die Skalierbarkeit des Clusters zu verbessern.

Voraussetzungen

Damit Amazon EKS Ihre CoreDNS-Bereitstellung skalieren kann, müssen drei Voraussetzungen erfüllt sein:

- Sie müssen die EKS-Add-On-Version von CoreDNS verwenden.
- Ihr Cluster muss mindestens die Mindest-Cluster- und Plattformversionen ausführen.
- Ihr Cluster muss mindestens die Mindestversion des EKS-Add-Ons von CoreDNS ausführen.

Minimale Cluster-Version

Die automatische Skalierung von CoreDNS erfolgt durch eine neue Komponente in der Cluster-Steuerebene, die von Amazon EKS verwaltet wird. Aus diesem Grund müssen Sie Ihren Cluster auf eine EKS-Version aktualisieren, welche die Mindestplattformversion mit der neuen Komponente unterstützt.

Ein neuer Amazon-EKS-Cluster. Informationen zum Bereitstellen finden Sie unter Erste Schritte. Der Cluster muss eine der in der folgenden Tabelle aufgeführten Kubernetes-Versionen und Plattformversionen oder eine neuere Version ausführen. Beachten Sie, dass alle Kubernetes- und Plattformversionen, die über die aufgeführten hinausgehen, ebenfalls unterstützt werden. Sie können Ihre aktuelle Kubernetes-Version überprüfen, indem Sie *my-cluster* im folgenden Befehl durch den Namen Ihres Clusters ersetzen und dann den geänderten Befehl ausführen:

```
aws eks describe-cluster --name my-cluster --query cluster.version --output text
```

Kubernetes-Version	Plattformversion
Nicht aufgelistet	Alle Plattformversionen
1.29.3	eks.7

Kubernetes-Version	Plattformversion
1.28.8	eks.13
1.27.12	eks.17
1.26.15	eks.18

> **Note**
>
> Alle Plattformversionen der neueren Kubernetes-Versionen werden ebenfalls unterstützt, beispielsweise Kubernetes-Versionen 1.30 ab eks.1.

Mindestversion des EKS-Add-Ons

Kubernetes-Version	1.29	1.28
	v1.11.1-eksbuild.9	v1.10.1-eksbuild.11

Konfiguration der automatischen Skalierung von CoreDNS in AWS-Managementkonsole

1. Stellen Sie sicher, dass Ihr Cluster die Mindest-Cluster-Version aufweist oder höher ist.

   Amazon EKS aktualisiert Cluster zwischen Plattformversionen derselben Kubernetes-Version automatisch, und Sie können diesen Vorgang nicht selbst starten. Stattdessen können Sie Ihren Cluster auf die nächste Kubernetes-Version aktualisieren, woraufhin der Cluster auf diese K8s-Version und die neueste Plattformversion aktualisiert wird.

   Neue Kubernetes-Versionen führen oft bedeutende Änderungen ein. Wir empfehlen Ihnen daher, das Verhalten Ihrer Anwendungen mit einem separaten Cluster der neuen Kubernetes-Version zu testen, bevor Sie Ihre Produktions-Cluster aktualisieren

   Um einen Cluster auf eine neue Kubernetes-Version zu aktualisieren, befolgen Sie die Schritte unter Vorhandenen Cluster auf neue Kubernetes-Version aktualisieren.
2. Stellen Sie sicher, dass Sie das EKS-Add-On für CoreDNS verwenden und nicht über die selbstverwaltete CoreDNS-Bereitstellung.

Je nachdem, mit welchem Tool Sie Ihr Cluster erstellt haben, ist der Add-on vom Typ Amazon EKS möglicherweise derzeit nicht auf Ihrem Cluster installiert. Um zu überprüfen, welche Art von Add-On auf Ihrem Cluster installiert ist, können Sie den folgenden Befehl ausführen. Ersetzen Sie `my-cluster` mit dem Namen Ihres Clusters.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 addon.addonVersion --output text
```

Wenn eine Versionsnummer zurückgegeben wird, ist das Add-On vom Typ Amazon EKS auf Ihrem Cluster installiert und Sie können mit dem nächsten Schritt fortfahren. Wenn ein Fehler zurückgegeben wird, ist der Amazon-EKS-Typ des Add-Ons nicht auf Ihrem Cluster installiert. Führen Sie die restlichen Schritte des Verfahrens CoreDNS-Amazon-EKS-Add-On erstellen, um die selbstverwaltete Version durch das Amazon-EKS-Add-On zu ersetzen.

3. Stellen Sie sicher, dass Ihr EKS-Add-On für CoreDNS mindestens die Mindestversion des EKS-Add-Ons aufweist.

   Sehen Sie, welche Version des Container-Images derzeit auf Ihrem Cluster installiert ist. Sie können dies in AWS-Managementkonsole überprüfen oder den folgenden Befehl ausführen:

   ```
   kubectl describe deployment coredns --namespace kube-system | grep coredns: | cut -
   d : -f 3
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   v1.10.1-eksbuild.13
   ```

   Vergleichen Sie diese Version mit der Mindestversion des EKS-Add-Ons im vorherigen Abschnitt. Aktualisieren Sie das EKS-Add-On bei Bedarf auf eine höhere Version, indem Sie das Verfahren CoreDNS-Amazon-EKS-Add-On aktualisieren befolgen.

4. Fügen Sie die Autoscaling-Konfiguration zu den Optionalen Konfigurationseinstellungen des EKS-Add-Ons hinzu.

   a. Öffnen Sie die Amazon-EKS-Konsole.

   b. Wählen Sie im linken Navigationsbereich Clusters (Cluster) aus. Wählen Sie anschließend den Namen des Clusters aus, für den Sie das Add-On konfigurieren möchten.

   c. Wählen Sie die Registerkarte Add-ons.

d. Aktivieren Sie das Kontrollkästchen oben rechts im Feld „CoreDNS-Add-On“ und wählen Sie dann Bearbeiten aus.

e. Gehen Sie auf der Seite CoreDNS konfigurieren wie folgt vor:

   i. Wählen Sie die Version aus, die Sie verwenden möchten. Wir empfehlen Ihnen, die gleiche Version wie im vorherigen Schritt beizubehalten und die Version und Konfiguration in separaten Schritten zu aktualisieren.

   ii. Erweitern Sie Optionale Konfigurationseinstellungen.

   iii. Geben Sie den JSON-Schlüssel `"autoscaling":` und den Wert eines verschachtelten JSON-Objekts mit einem Schlüssel `"enabled":` und einem Wert `true` in den Konfigurationswerten ein. Der resultierende Text muss ein gültiges JSON-Objekt sein. Wenn dieser Schlüssel und dieser Wert die einzigen Daten im Textfeld sind, setzen Sie den Schlüssel und den Wert in geschweifte Klammern `{ }`. Das folgende Beispiel zeigt, dass die automatische Skalierung aktiviert ist:

```
{
  "autoScaling": {
    "enabled": true
  }
}
```

   iv. (Optional) Sie können Mindest- und Höchstwerte angeben, auf welche die automatische Skalierung die Anzahl der CoreDNS-Pods skalieren kann.

   Das folgende Beispiel zeigt, dass die automatische Skalierung aktiviert ist und alle optionalen Schlüssel über Werte verfügen. Wir empfehlen, die Mindestanzahl der CoreDNS-Pods immer größer als 2 zu halten, um die Ausfallsicherheit des DNS-Services im Cluster zu gewährleisten.

```
{
  "autoScaling": {
    "enabled": true,
    "minReplicas": 2,
    "maxReplicas": 10
  }
}
```

f. Um die neue Konfiguration durch Ersetzen der CoreDNS-Pods anzuwenden, wählen Sie Änderungen speichern aus.

Amazon EKS wendet Änderungen an den EKS-Add-Ons an, indem es eine Einführung der Kubernetes-Bereitstellung für CoreDNS durchführt. Sie können den Status der Einführung im Aktualisierungsverlauf des Add-Ons in AWS-Managementkonsole und mit `kubectl rollout status deployment/coredns --namespace kube-system` verfolgen.

`kubectl rollout` verfügt über die folgenden Befehle:

```
kubectl rollout

history  -- View rollout history
pause    -- Mark the provided resource as paused
restart  -- Restart a resource
resume   -- Resume a paused resource
status   -- Show the status of the rollout
undo     -- Undo a previous rollout
```

Wenn die Einführung zu lange dauert, macht Amazon EKS die Einführung rückgängig und eine Nachricht mit dem Typ der Add-On-Aktualisierung und dem Status Fehlgeschlagen dem Aktualisierungsverlauf des Add-Ons hinzugefügt. Um etwaige Probleme zu untersuchen, beginnen Sie mit dem Verlauf der Einführung und führen Sie `kubectl logs` in einem CoreDNS-Pod aus, um die Protokolle von CoreDNS anzuzeigen.

5. Wenn der neue Eintrag im Aktualisierungsverlauf den Status Erfolgreich aufweist, ist die Einführung abgeschlossen und das Add-On verwendet die neue Konfiguration in allen CoreDNS-Pods. Wenn Sie die Anzahl der Knoten und CPU-Kerne der Knoten im Cluster ändern, skaliert Amazon EKS die Anzahl der Replikate der CoreDNS-Bereitstellung.

Konfiguration der automatischen Skalierung von CoreDNS in der AWS-Befehlszeilenschnittstelle

1. Stellen Sie sicher, dass Ihr Cluster die Mindest-Cluster-Version aufweist oder höher ist.

   Amazon EKS aktualisiert Cluster zwischen Plattformversionen derselben Kubernetes-Version automatisch, und Sie können diesen Vorgang nicht selbst starten. Stattdessen können Sie Ihren Cluster auf die nächste Kubernetes-Version aktualisieren, woraufhin der Cluster auf diese K8s-Version und die neueste Plattformversion aktualisiert wird.

   Neue Kubernetes-Versionen führen oft bedeutende Änderungen ein. Wir empfehlen Ihnen daher, das Verhalten Ihrer Anwendungen mit einem separaten Cluster der neuen Kubernetes-Version zu testen, bevor Sie Ihre Produktions-Cluster aktualisieren

Um einen Cluster auf eine neue Kubernetes-Version zu aktualisieren, befolgen Sie die Schritte unter Vorhandenen Cluster auf neue Kubernetes-Version aktualisieren.

2. Stellen Sie sicher, dass Sie das EKS-Add-On für CoreDNS verwenden und nicht über die selbstverwaltete CoreDNS-Bereitstellung.

   Je nachdem, mit welchem Tool Sie Ihr Cluster erstellt haben, ist der Add-on vom Typ Amazon EKS möglicherweise derzeit nicht auf Ihrem Cluster installiert. Um zu überprüfen, welche Art von Add-On auf Ihrem Cluster installiert ist, können Sie den folgenden Befehl ausführen. Ersetzen Sie `my-cluster` mit dem Namen Ihres Clusters.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
    addon.addonVersion --output text
   ```

   Wenn Sie eine Versionsnummer zurückgeben, wird der Amazon-EKS-Typ des Add-Ons auf Ihrem Cluster installiert. Wenn ein Fehler zurückgegeben wird, ist der Amazon-EKS-Typ des Add-Ons nicht auf Ihrem Cluster installiert. Führen Sie die restlichen Schritte des Verfahrens CoreDNS-Amazon-EKS-Add-On erstellen, um die selbstverwaltete Version durch das Amazon-EKS-Add-On zu ersetzen.

3. Stellen Sie sicher, dass Ihr EKS-Add-On für CoreDNS mindestens die Mindestversion des EKS-Add-Ons aufweist.

   Sehen Sie, welche Version des Container-Images derzeit auf Ihrem Cluster installiert ist. Sie können dies in AWS-Managementkonsole überprüfen oder den folgenden Befehl ausführen:

   ```
   kubectl describe deployment coredns --namespace kube-system | grep coredns: | cut -
   d : -f 3
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   v1.10.1-eksbuild.13
   ```

   Vergleichen Sie diese Version mit der Mindestversion des EKS-Add-Ons im vorherigen Abschnitt. Aktualisieren Sie das EKS-Add-On bei Bedarf auf eine höhere Version, indem Sie das Verfahren CoreDNS-Amazon-EKS-Add-On aktualisieren befolgen.

4. Fügen Sie die Autoscaling-Konfiguration zu den Optionalen Konfigurationseinstellungen des EKS-Add-Ons hinzu.

Führen Sie den folgenden AWS-CLI-Befehl aus. Ersetzen Sie `my-cluster` durch den Namen Ihres Clusters und den IAM-Rollen-ARN durch die Rolle, die Sie verwenden.

```
aws eks update-addon --cluster-name my-cluster --addon-name coredns \
    --resolve-conflicts PRESERVE --configuration-values '{"autoScaling":
{"enabled":true}}'
```

Amazon EKS wendet Änderungen an den EKS-Add-Ons an, indem es eine Einführung der Kubernetes-Bereitstellung für CoreDNS durchführt. Sie können den Status der Einführung im Aktualisierungsverlauf des Add-Ons in AWS-Managementkonsole und mit `kubectl rollout status deployment/coredns --namespace kube-system` verfolgen.

`kubectl rollout` verfügt über die folgenden Befehle:

```
kubectl rollout

history  -- View rollout history
pause    -- Mark the provided resource as paused
restart  -- Restart a resource
resume   -- Resume a paused resource
status   -- Show the status of the rollout
undo     -- Undo a previous rollout
```

Wenn die Einführung zu lange dauert, macht Amazon EKS die Einführung rückgängig und eine Nachricht mit dem Typ der Add-On-Aktualisierung und dem Status Fehlgeschlagen dem Aktualisierungsverlauf des Add-Ons hinzugefügt. Um etwaige Probleme zu untersuchen, beginnen Sie mit dem Verlauf der Einführung und führen Sie `kubectl logs` in einem CoreDNS-Pod aus, um die Protokolle von CoreDNS anzuzeigen.

5. (Optional) Sie können Mindest- und Höchstwerte angeben, auf welche die automatische Skalierung die Anzahl der CoreDNS-Pods skalieren kann.

   Das folgende Beispiel zeigt, dass die automatische Skalierung aktiviert ist und alle optionalen Schlüssel über Werte verfügen. Wir empfehlen, die Mindestanzahl der CoreDNS-Pods immer größer als 2 zu halten, um die Ausfallsicherheit des DNS-Services im Cluster zu gewährleisten.

   ```
   aws eks update-addon --cluster-name my-cluster --addon-name coredns \
       --resolve-conflicts PRESERVE --configuration-values '{"autoScaling":
   {"enabled":true,"minReplicas":2,"maxReplicas":10}}'
   ```

6. Überprüfen Sie den Status der Aktualisierung des Add-Ons, indem Sie den folgenden Befehl ausführen:

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns
```

   Wenn Sie die folgende Zeile sehen: `"status": "ACTIVE"`, ist die Einführung abgeschlossen und das Add-On verwendet die neue Konfiguration in allen CoreDNS-Pods. Wenn Sie die Anzahl der Knoten und CPU-Kerne der Knoten im Cluster ändern, skaliert Amazon EKS die Anzahl der Replikate der CoreDNS-Bereitstellung.

Überwachung der Kubernetes-DNS-Auflösung mit CoreDNS-Metriken

CoreDNS als EKS-Add-On stellt die Metriken von CoreDNS auf dem Port 9153 im Prometheus-Format im `kube-dns`-Service bereit. Sie können Prometheus, den Amazon-CloudWatch-Agenten oder ein anderes kompatibles System verwenden, um diese Metriken zu erfassen.

Ein Beispiel für eine Scrape-Konfiguration, die sowohl mit Prometheus als auch mit dem CloudWatch-Agenten kompatibel ist, finden Sie unter CloudWatch-Agentenkonfiguration für Prometheus im Amazon-CloudWatch-Benutzerhandbuch.

Verwaltung von **`kube-proxy`** in Amazon-EKS-Clustern

> **Tip**
>
> Mit Amazon EKS Auto Mode ist es nicht erforderlich, Netzwerk-Add-Ons zu installieren oder zu aktualisieren. Der Automatikmodus umfasst Funktionen für Pod-Netzwerke und Load Balancing.
> Weitere Informationen finden Sie unter EKS Auto Mode.

Wir empfehlen, den Amazon-EKS-Typ des Add-Ons zu Ihrem Cluster hinzuzufügen, anstatt den selbstverwalteten Typ des Add-Ons zu verwenden. Wenn Sie noch keine Erfahrung mit den Unterschieden zwischen den Typen haben, finden Sie weitere Informationen unter the section called “Amazon-EKS-Add-ons”. Weitere Informationen zum Hinzufügen eines Amazon-EKS-Add-ons zu Ihrem Cluster finden Sie unter the section called “Add-On erstellen”. Wenn Sie das Amazon EKS-Add-on nicht verwenden können, empfehlen wir Ihnen, ein Problem mit der Begründung, warum Sie das nicht können, an das GitHub Container-Roadmap-Repository zu senden.

Das `kube-proxy` Add-on wird auf jedem EC2 Amazon-Knoten in Ihrem Amazon EKS-Cluster bereitgestellt. Es verwaltet Netzwerkregeln auf Ihren Knoten und ermöglicht die Netzwerkkommunikation mit Ihren Pods. Das Add-On wird nicht auf Fargate-Knoten in Ihrem Cluster bereitgestellt. Weitere Informationen finden Sie unter kube-proxy in der Kubernetes-Dokumentation.

Installation als Amazon-EKS-Add-On

`kube-proxy`-Versionen

In der folgenden Tabelle ist die neueste Version des Amazon-EKS-Add-On-Typs für jede Kubernetes-Version aufgeführt.

Kubernetes-Version	**`kube-proxy`** -Version
1.34	v1.34.1-eksbuild.2
1,33	v1.33.5-eksbuild.2
1.32	v1.32.9-eksbuild.2
1,31	v1.31.13-eksbuild.2
1,30	v1.30.14-eksbuild.8
1,29	v1.29.15-eksbuild.16
1,28	v1.28.15-eksbuild.31

> **Note**
>
> Eine frühere Version der Dokumentation war nicht korrekt. `kube-proxy`-Versionen `v1.28.5`, `v1.27.9` und `v1.26.12` sind nicht verfügbar.
> Wenn Sie dieses Add-On selbst verwalten, stimmen die Versionen in der Tabelle möglicherweise nicht mit den verfügbaren selbstverwalteten Versionen überein.

kube-proxy-Container-Image

Das Container-Image `kube-proxy` basiert auf einem minimalen Basis-Image, das von Amazon EKS Distro verwaltet wird, minimale Pakete enthält und keine Shells hat. Weitere Informationen finden Sie unter Amazon-EKS-Distro.

Die folgende Tabelle listet die neueste verfügbare selbstverwaltete Version des `kube-proxy`-Container-Image auf, die für jede Amazon-EKS-Cluster-Version verfügbar ist.

Version	kube-proxy
1,34	v1.34.1-eksbuild.2
1,33	v1.33.5-minimal-eksbuild.2
1.32	v1.32.9-minimal-eksbuild.2
1,31	v1.31.13-minimal-eksbuild.2
1,30	v1.30.14-minimal-eksbuild.8
1,29	v1.29.15-minimal-eksbuild.16
1,28	v1.28.15-minimal-eksbuild.31

Wenn Sie ein Add-on vom Typ Amazon EKS aktualisieren, geben Sie eine gültige Amazon-EKS-Add-on-Version an, bei der es sich möglicherweise nicht um eine in dieser Tabelle aufgeführte Version handelt. Dies liegt daran, dass die Amazon-EKS-Add-On-Versionen nicht immer mit den Container-Image-Versionen übereinstimmen, die bei der Aktualisierung des selbstverwalteten Typs dieses Add-Ons angegeben wurden. Wenn Sie den selbstverwalteten Typ dieses Add-ons aktualisieren, geben Sie eine gültige Container-Image-Version an, die in dieser Tabelle aufgeführt ist.

Selbstverwaltetes Kubernetes **kube-proxy**-Add-On aktualisieren

> **Important**
>
> Wir empfehlen, den Amazon-EKS-Typ des Add-Ons zu Ihrem Cluster hinzuzufügen, anstatt den selbstverwalteten Typ des Add-Ons zu verwenden. Wenn Sie noch keine Erfahrung mit den Unterschieden zwischen den Typen haben, finden Sie weitere Informationen

unter the section called “Amazon-EKS-Add-ons”. Weitere Informationen zum Hinzufügen eines Amazon-EKS-Add-ons zu Ihrem Cluster finden Sie unter the section called “Add-On erstellen”. Wenn Sie das Amazon-EKS-Add-On nicht verwenden können, empfehlen wir Ihnen, ein Problem zu den Gründen, warum Sie es nicht verwenden können, an das GitHub-Repository Containers Roadmap zu senden.

Voraussetzungen

- Ein vorhandener Amazon-EKS-Cluster. Informationen zum Bereitstellen finden Sie unter Erste Schritte.

Überlegungen

- `Kube-proxy` in einem Amazon-EKS-Cluster verfügt über die gleiche Kompatibilitäts- und Skew-Richtlinie wie Kubernetes. Erfahren Sie, wie Sie die Kompatibilität der Add-On-Version für Amazon EKS mit einem Cluster überprüfen.

  1. Vergewissern Sie sich, dass Sie das selbstverwaltete Add-On auf Ihrem Cluster installiert haben. Ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters.

     ```
     aws eks describe-addon --cluster-name my-cluster --addon-name kube-proxy --query
      addon.addonVersion --output text
     ```

     Wenn Sie eine Fehlermeldung erhalten, wird das Add-On als selbstverwaltetes Add-On auf Ihrem Cluster installiert. Die verbleibenden Schritte in diesem Thema beziehen sich auf die Aktualisierung des selbstverwalteten Typs des Add-Ons. Wenn Sie eine Versionsnummer zurückgeben, wird der Amazon-EKS-Typ des Add-Ons auf Ihrem Cluster installiert. Verwenden Sie zum Aktualisieren das Verfahren unter Aktualisierung eines Amazon-EKS-Add-Ons und nicht das Verfahren in diesem Thema. Wenn Sie mit den Unterschieden zwischen den Add-On-Typen nicht vertraut sind, finden Sie Informationen unter the section called “Amazon-EKS-Add-ons”.
  2. Sehen Sie, welche Version des Container-Images derzeit auf Ihrem Cluster installiert ist.

     ```
     kubectl describe daemonset kube-proxy -n kube-system | grep Image
     ```

     Eine Beispielausgabe sieht wie folgt aus.

```
Image:    602401143452.dkr.ecr.region-code.amazonaws.com/eks/kube-proxy:v1.29.1-
eksbuild.2
```

In der Beispielausgabe ist *v1.29.1-eksbuild.2* die auf dem Cluster installierte Version.

3. Aktualisieren Sie das kube-proxy-Add-On, indem Sie *602401143452* und *region-code* durch die Werte aus Ihrer Ausgabe im vorherigen Schritt ersetzen. Ersetzen Sie *v1.30.6-eksbuild.3* durch die kube-proxy-Version, die in der Tabelle Neueste verfügbare selbstverwaltete Kube-Proxy-Container-Image-Version für jede Amazon-EKS-Cluster-Version aufgeführt ist.

> **Important**
>
> Die Manifeste für jeden Image-Typ sind unterschiedlich und nicht zwischen den Standard- oder minimale Image-Typen kompatibel. Sie müssen denselben Image-Typ wie beim vorherigen Image verwenden, damit der Einstiegspunkt und die Argumente übereinstimmen.

```
kubectl set image daemonset.apps/kube-proxy -n kube-system kube-
proxy=602401143452.dkr.ecr.region-code.amazonaws.com/eks/kube-proxy:v1.30.6-
eksbuild.3
```

Eine Beispielausgabe sieht wie folgt aus.

```
daemonset.apps/kube-proxy image updated
```

4. Vergewissern Sie sich, dass die neue Version jetzt auf Ihrem Cluster installiert ist.

```
kubectl describe daemonset kube-proxy -n kube-system | grep Image | cut -d ":" -f 3
```

Eine Beispielausgabe sieht wie folgt aus.

```
v1.30.0-eksbuild.3
```

5. Wenn Sie x86- und Arm-Knoten im selben Cluster verwenden und Ihr Cluster vor dem 17. August 2020 bereitgestellt wurde. Bearbeiten Sie dann Ihr kube-proxy-Manifest, um einen Knotenselektor für mehrere Hardwarearchitekturen mit dem folgenden Befehl einzuschließen.

Dies ist ein einmaliger Vorgang. Nachdem Sie den Selektor zu Ihrem Manifest hinzugefügt haben, müssen Sie ihn nicht bei jeder Aktualisierung des Add-Ons hinzufügen. Wenn Ihr Cluster am oder nach dem 17. August 2020 bereitgestellt wurde, ist `kube-proxy` bereits Multi-Architektur-fähig.

```
kubectl edit -n kube-system daemonset/kube-proxy
```

Fügen Sie der Datei im Editor den folgenden Knotenselektor hinzu und speichern Sie die Datei. Ein Beispiel dafür, wo dieser Text in den Editor eingefügt werden soll, finden Sie in der CNI-Manifest-Datei auf GitHub. Dadurch kann Kubernetes basierend auf der Hardware-Architektur des Knotens das richtige Hardware-Image abrufen.

```
- key: "kubernetes.io/arch"
  operator: In
  values:
  - amd64
  - arm64
```

6. Wenn Ihr Cluster ursprünglich mit der Kubernetes-Version `1.14` oder höher erstellt wurde, können Sie diesen Schritt überspringen, da `kube-proxy` bereits diese `Affinity Rule` enthält. Wenn Sie ursprünglich einen Amazon-EKS-Cluster mit der Kubernetes-Version `1.13` oder früher erstellt haben und Fargate-Knoten in Ihrem Cluster verwenden älter, bearbeiten Sie Ihr `kube-proxy`-Manifest so, dass es eine `NodeAffinity`-Regel enthält, um zu verhindern, dass `kube-proxy` Pods in Fargate-Knoten geplant werden. Dies ist eine einmalige Bearbeitung. Sobald Sie das `Affinity Rule` zu Ihrem Manifest hinzugefügt haben, müssen Sie es nicht jedes Mal hinzufügen, wenn Sie das Add-On aktualisieren. Bearbeiten Sie Ihr `kube-proxy`-DaemonSet.

   ```
   kubectl edit -n kube-system daemonset/kube-proxy
   ```

   Fügen Sie die folgende `Affinity Rule` zum Abschnitt DaemonSet `spec` der Datei im Editor hinzu und speichern Sie die Datei. Ein Beispiel dafür, wo dieser Text in den Editor eingefügt werden soll, finden Sie in der CNI-Manifest-Datei auf GitHub.

   ```
   - key: eks.amazonaws.com/compute-type
     operator: NotIn
     values:
     - fargate
   ```

Erfahren Sie, wie Sie Workloads und Add-Ons in Amazon EKS bereitstellen.

Ihre Workloads werden in Containern bereitgestellt, die in Pods in Kubernetes bereitgestellt werden. Ein Pod enthält einen oder mehrere Container. In der Regel werden ein oder mehrere Pods, die denselben Service bieten, in einem Kubernetes-Service bereitgestellt. Nachdem Sie mehrere Pods bereitgestellt haben, die denselben Service bieten, können Sie:

- Anzeigen von Informationen über die Workloads, die auf jedem Ihrer Cluster laufen mithilfe von AWS-Managementkonsole ausgeführt wird.
- Die Pods vertikal mit dem Kubernetes Vertical Pod Autoscaler nach oben oder unten skalieren.
- Die Anzahl der Pods horizontal skalieren, die benötigt werden, um die Nachfrage nach oben oder unten mit den Kubernetes Horizontal Pod Autoscaler zu decken.
- Einen externen (für Internet-zugängliche Pods) oder eine interne (für private Pods) Network Load Balancer erstellen, um den Netzwerk-Datenverkehr über Pods auszugleichen. Der Load Balancer leitet den Datenverkehr an Layer 4 des OSI-Modells weiter.
- Einen Application Load Balancer erstellen, um den Anwendungsdatenverkehr über Pods hinweg auszugleichen. Der Application Load Balancer leitet den Datenverkehr auf Layer 7 des OSI-Modells weiter.
- Wenn Sie Kubernetes noch nicht kennen, unterstützt Sie dieses Thema bei der Bereitstellung einer Beispiel-Anwendung.
- Sie können IP-Adressen einschränken, die einem Service zugewiesen werden können mit `externalIPs`.

Eine Beispielanwendung in Linux bereitstellen

In diesem Thema stellen Sie eine Beispielanwendung auf Ihrem Cluster in Linux-Knoten bereit.

Voraussetzungen

- Ein vorhandener Kubernetes-Cluster mit mindestens einem Knoten. Wenn Sie noch keinen vorhandenen Amazon-EKS-Cluster haben, können Sie einen mit einem der Anleitungen in Erste Schritte bereitstellen.

- `Kubectl` auf Ihrem Computer installiert. Weitere Informationen finden Sie unter the section called “kubectl und eksctl einrichten”.
- `Kubectl` für die Kommunikation mit Ihrem Cluster konfiguriert. Weitere Informationen finden Sie unter the section called “Zugriff auf den Cluster mit kubectl”.
- Wenn Sie vorhaben, Ihre Beispiel-Workload in Fargate bereitzustellen, müssen Sie über ein vorhandenes Fargate-Profil verfügen, das denselben in diesem Tutorial erstellten Namespace enthält, nämlich `eks-sample-app`, es sei denn, Sie ändern den Namen. Wenn Sie einen Cluster mit einer der Anleitungen in Erste Schritte erstellt haben, müssen Sie ein neues Profil erstellen oder den Namespace zu Ihrem vorhandenen. Profil hinzufügen. Das in den Anleitungen für den Einstieg erstellte Profil gibt den in diesem Tutorial verwendeten Namespace nicht an. Ihre VPC muss auch über mindestens ein privates Subnetz verfügen.

Obwohl viele Variablen in den folgenden Schritten veränderbar sind, empfehlen wir, Variablenwerte nur zu ändern, wo angegeben. Sobald Sie Kubernetes-Pods, Bereitstellungen und Services besser verstanden haben, können Sie mit dem Ändern anderer Werte experimentieren.

Namespaces erstellen

Ein Namespace ermöglicht es Ihnen, Ressourcen in Kubernetes zu gruppieren. Weitere Informationen finden Sie unter Namespaces in der Kubernetes-Dokumentation. Wenn Sie Ihre Beispielanwendung bereitstellen möchten, um die Rechenverwaltung mit AWS Fargate zu vereinfachen, stellen Sie sicher, dass der Wert für `namespace` in Ihrer Definition, welche Pods beim Start AWS Fargate verwenden, `eks-sample-app` ist.

```
kubectl create namespace eks-sample-app
```

Erstellen Sie eine Kubernetes-Bereitstellung.

Erstellen Sie eine Kubernetes-Bereitstellung. Diese Beispiel-Bereitstellung ruft ein Container-Image aus einem öffentlichen Repository ab und stellt drei Replikate (individuelle Pods) davon in Ihrem Cluster bereit. Weitere Informationen finden Sie unter Bereitstellungen in der Kubernetes-Dokumentation.

1. Speichern Sie den folgenden Inhalt in einer Datei mit dem Namen `eks-sample-deployment.yaml`. Die Container in der Beispielanwendung verwenden keinen Netzwerkspeicher, aber möglicherweise haben Sie Anwendungen, die dies benötigen. Weitere Informationen finden Sie unter Anwendungsdatenspeicher.

- Das `amd64` oder `arm64` `values` unter dem Schlüssel `kubernetes.io/arch` bedeutet, dass die Anwendung in einer der beiden Hardwarearchitekturen bereitgestellt werden kann (wenn Sie beide in Ihrem Cluster haben). Dies ist möglich, weil dieses Image ein Multi-Architektur-Image ist, aber das sind nicht alle. Sie können die Hardware-Architektur bestimmen, auf der das Image unterstützt wird, indem Sie die Image-Details im Repository anzeigen, aus dem Sie es abrufen. Wenn Sie Images bereitstellen, die keinen Hardware-Architektur-Typ unterstützen oder an die das Image nicht bereitgestellt werden soll, entfernen Sie diesen Typ aus dem Manifest. Weitere Informationen finden Sie unter Bekannte Labels, Anmerkungen und Taints in der Kubernetes-Dokumentation.
- Der Eintrag `kubernetes.io/os: linux` `nodeSelector` bedeutet, dass, wenn Sie beispielsweise Windows- und Linux-Knoten in Ihrem Cluster hätten, das Image nur auf Linux-Knoten bereitgestellt würde. Weitere Informationen finden Sie unter Bekannte Labels, Anmerkungen und Taints in der Kubernetes-Dokumentation.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: eks-sample-linux-deployment
  namespace: eks-sample-app
  labels:
    app: eks-sample-linux-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: eks-sample-linux-app
  template:
    metadata:
      labels:
        app: eks-sample-linux-app
    spec:
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
              - key: kubernetes.io/arch
                operator: In
                values:
                - amd64
                - arm64
```

```
      containers:
      - name: nginx
        image: public.ecr.aws/nginx/nginx:1.23
        ports:
        - name: http
          containerPort: 80
        imagePullPolicy: IfNotPresent
      nodeSelector:
        kubernetes.io/os: linux
```

2. Wenden Sie das Bereitstellungs-Manifest auf Ihren Cluster an.

```
kubectl apply -f eks-sample-deployment.yaml
```

Erstellen eines Services

Mit einem Service können Sie über eine einzige IP-Adresse oder einen einzigen Namen auf alle Replikate zugreifen. Weitere Informationen dazu finden Sie unter Service in der Kubernetes-Dokumentation. Obwohl in der Beispielanwendung nicht implementiert, empfehlen wir Ihnen, Kubernetes-Servicekonten für Ihre Pods zu erstellen und sie AWS-IAM-Konten zuzuordnen, wenn Sie über Anwendungen verfügen, die mit anderen AWS-Services interagieren müssen. Durch die Angabe von Dienstkonten verfügen Ihre Pods nur über die Mindestberechtigungen, die Sie für die Interaktion mit anderen Diensten angeben. Weitere Informationen finden Sie unter the section called “Anmeldeinformationen mit IRSA”.

1. Speichern Sie die folgenden Inhalte in einer Datei namens `eks-sample-service.yaml`. Kubernetes weist dem Service eine eigene IP-Adresse zu, auf die nur innerhalb des Clusters zugegriffen werden kann. Um von außerhalb des Clusters auf den Service zuzugreifen, stellen Sie den AWS Load Balancer Controller bereit, um den Anwendungs- oder Netzwerk-Datenverkehr an den Service auszugleichen.

```
apiVersion: v1
kind: Service
metadata:
  name: eks-sample-linux-service
  namespace: eks-sample-app
  labels:
    app: eks-sample-linux-app
spec:
  selector:
```

```
    app: eks-sample-linux-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

2. Wenden Sie das Service-Manifest auf Ihren Cluster an.

```
kubectl apply -f eks-sample-service.yaml
```

Erstellte Ressourcen überprüfen

1. Zeigen Sie alle im `eks-sample-app`-Namespace vorhandenen Ressourcen an.

```
kubectl get all -n eks-sample-app
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME                                                 READY   STATUS    RESTARTS   AGE
pod/eks-sample-linux-deployment-65b7669776-m6qxz   1/1     Running   0          27m
pod/eks-sample-linux-deployment-65b7669776-mmxvd   1/1     Running   0          27m
pod/eks-sample-linux-deployment-65b7669776-qzn22   1/1     Running   0          27m

NAME                                 TYPE         CLUSTER-IP      EXTERNAL-IP   PORT(S)
   AGE
service/eks-sample-linux-service   ClusterIP    10.100.74.8     <none>        80/TCP
   32m

NAME                                          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/eks-sample-linux-deployment 3/3     3            3           27m

NAME                                                      DESIRED   CURRENT   READY
 AGE
replicaset.apps/eks-sample-linux-deployment-776d8f8fd8    3         3         3
 27m
```

In der Ausgabe sehen Sie den Service und die Bereitstellung, die in den Beispielmanifesten, der vorherigen Schritte, angegeben wurden. Sie sehen auch drei Pods. Dies liegt daran, dass `3` `replicas` im Beispielmanifest angegeben wurden. Weitere Informationen zu Pods finden Sie unter Pods in der Kubernetes-Dokumentation. Kubernetes erstellt automatisch die `replicaset-`

Ressource, obwohl sie nicht in den Beispiel-Manifesten angegeben ist. Weitere Informationen zu `ReplicaSets` finden Sie unter ReplicaSet in der Kubernetes-Dokumentation.

> **Note**
>
> Kubernetes behält die Anzahl der Replikate bei, die im Manifest angegeben sind. Wenn es sich um eine Produktionsbereitstellung handelt und Sie möchten, dass Kubernetes die Anzahl der Replikate horizontal skaliert oder die Rechenressourcen für die Pods vertikal skaliert, verwenden Sie dazu die Optionen Pod-Bereitstellungen mit horizontalem Pod-Autoscaler skalieren und Pod-Ressourcen mit vertikalem Pod-Autoscaler anpassen.

2. Zeigen Sie die Details des bereitgestellten Services an.

```
kubectl -n eks-sample-app describe service eks-sample-linux-service
```

Eine Beispielausgabe sieht wie folgt aus.

```
Name:              eks-sample-linux-service
Namespace:         eks-sample-app
Labels:            app=eks-sample-linux-app
Annotations:       <none>
Selector:          app=eks-sample-linux-app
Type:              ClusterIP
IP Families:       <none>
IP:                10.100.74.8
IPs:               10.100.74.8
Port:              <unset>  80/TCP
TargetPort:        80/TCP
Endpoints:         192.168.24.212:80,192.168.50.185:80,192.168.63.93:80
Session Affinity:  None
Events:            <none>
```

In der vorherigen Ausgabe ist der Wert für `IP:` eine eindeutige IP-Adresse, die von jedem Knoten oder Pod innerhalb des Clusters aus erreicht werden kann, jedoch nicht von außerhalb des Clusters. Die Werte für `Endpoints` sind IP-Adressen, die innerhalb Ihrer VPC den Pods zugewiesen werden, die Teil des Services sind.

3. Zeigen Sie die Details eines der in der Ausgabe aufgeführten Pods an, wenn Sie in einem vorherigen Schritt den Namespace angezeigt haben. Ersetzen Sie *776d8f8fd8-78w66* durch den für einen Ihrer Pods zurückgegebenen Wert.

```
kubectl -n eks-sample-app describe pod eks-sample-linux-deployment-65b7669776-m6qxz
```

Gekürzte Beispielausgabe

```
Name:         eks-sample-linux-deployment-65b7669776-m6qxz
Namespace:    eks-sample-app
Priority:     0
Node:         ip-192-168-45-132.us-west-2.compute.internal/192.168.45.132
[...]
IP:           192.168.63.93
IPs:
  IP:           192.168.63.93
Controlled By:  ReplicaSet/eks-sample-linux-deployment-65b7669776
[...]
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
[...]
Events:
  Type    Reason     Age    From
 Message
  ----    ------     ----   ----
 -------
  Normal  Scheduled  3m20s  default-scheduler
 Successfully assigned eks-sample-app/eks-sample-linux-deployment-65b7669776-m6qxz to
 ip-192-168-45-132.us-west-2.compute.internal
[...]
```

In der vorherigen Ausgabe ist der Wert für `IP:` eine eindeutige IP, die dem Pod aus dem CIDR-Block zugewiesen wird, der dem Subnetz zugewiesen ist, in dem sich der Knoten befindet. Wenn Sie es vorziehen, Pods IP-Adressen aus verschiedenen CIDR-Blöcken zuzuweisen, können Sie das Standardverhalten ändern. Weitere Informationen finden Sie unter the section called “Benutzerdefinierte Netzwerke”. Sie können auch sehen, dass der Kubernetes-Scheduler den Pod in `Node` mit der IP-Adresse *`192.168.45.132`* geplant hat.

> **Tip**
>
> Anstatt die Befehlszeile zu verwenden, können Sie viele Details zu Pods, Services, Bereitstellungen und anderen Kubernetes-Ressourcen in der AWS-Managementkonsole anzeigen. Weitere Informationen finden Sie unter the section called “Zugriff auf Cluster-Ressourcen”.

Shell in einem Pod ausführen

1. Führen Sie eine Shell auf dem Pod aus, die Sie im vorherigen Schritt beschrieben haben, und ersetzen Sie *65b7669776-m6qxz* mit der ID eines Ihrer Pods.

```
kubectl exec -it eks-sample-linux-deployment-65b7669776-m6qxz -n eks-sample-app -- /
bin/bash
```

2. Zeigen Sie in der Pod-Shell die Ausgabe des Web-Servers an, der in einem vorherigen Schritt mit Ihrer Bereitstellung installiert wurde. Sie müssen nur den Servicenamen angeben. Er wird von CoreDNS, das mit einem Amazon-EKS-Cluster bereitgestellt wird, standardmäßig in die IP-Adresse des Services aufgelöst.

```
curl eks-sample-linux-service
```

   Eine Beispielausgabe sieht wie folgt aus.

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
[...]
```

3. Zeigen Sie in der Pod-Shell den DNS-Server für den Pod an.

```
cat /etc/resolv.conf
```

   Eine Beispielausgabe sieht wie folgt aus.

```
nameserver 10.100.0.10
```

```
search eks-sample-app.svc.cluster.local svc.cluster.local cluster.local us-
west-2.compute.internal
options ndots:5
```

In der vorherigen Ausgabe wird `10.100.0.10` automatisch als `nameserver` für alle Pods zugewiesen, die im Cluster bereitgestellt werden.

4. Trennen Sie die Verbindung mit dem Pod, indem Sie `exit` eingeben.
5. Wenn Sie die Beispielanwendung nicht mehr benötigen, können Sie den Beispiel-Namespace, den Service und die Bereitstellung mit dem folgenden Befehl entfernen.

```
kubectl delete namespace eks-sample-app
```

Nächste Schritte

Nachdem Sie die Beispielanwendung bereitgestellt haben, können Sie einige der folgenden Übungen ausprobieren:

- Anwendungen und HTTP-Datenverkehr mit Application Load Balancers weiterleiten
- Weiterleitung von TCP- und UDP-Datenverkehr mit Network Load Balancers

Eine Beispielanwendung in Windows bereitstellen

In diesem Thema stellen Sie eine Beispielanwendung auf Ihrem Cluster in Windows-Knoten bereit.

Voraussetzungen

- Ein vorhandener Kubernetes-Cluster mit mindestens einem Knoten. Wenn Sie noch keinen vorhandenen Amazon-EKS-Cluster haben, können Sie einen mit einem der Anleitungen in Erste Schritte bereitstellen. Sie müssen den Windows-Support für Ihren Cluster und mindestens einen Amazon-EC2-Windows-Knoten aktiviert haben.
- `Kubectl` auf Ihrem Computer installiert. Weitere Informationen finden Sie unter the section called “kubectl und eksctl einrichten”.
- `Kubectl` für die Kommunikation mit Ihrem Cluster konfiguriert. Weitere Informationen finden Sie unter the section called “Zugriff auf den Cluster mit kubectl”.
- Wenn Sie vorhaben, Ihre Beispiel-Workload in Fargate bereitzustellen, müssen Sie über ein vorhandenes Fargate-Profil verfügen, das denselben in diesem Tutorial erstellten Namespace

enthält, nämlich `eks-sample-app`, es sei denn, Sie ändern den Namen. Wenn Sie einen Cluster mit einer der Anleitungen in Erste Schritte erstellt haben, müssen Sie ein neues Profil erstellen oder den Namespace zu Ihrem vorhandenen. Profil hinzufügen. Das in den Anleitungen für den Einstieg erstellte Profil gibt den in diesem Tutorial verwendeten Namespace nicht an. Ihre VPC muss auch über mindestens ein privates Subnetz verfügen.

Obwohl viele Variablen in den folgenden Schritten veränderbar sind, empfehlen wir, Variablenwerte nur zu ändern, wo angegeben. Sobald Sie Kubernetes-Pods, Bereitstellungen und Services besser verstanden haben, können Sie mit dem Ändern anderer Werte experimentieren.

Namespaces erstellen

Ein Namespace ermöglicht es Ihnen, Ressourcen in Kubernetes zu gruppieren. Weitere Informationen finden Sie unter Namespaces in der Kubernetes-Dokumentation. Wenn Sie Ihre Beispielanwendung bereitstellen möchten, um die Rechenverwaltung mit AWS Fargate zu vereinfachen, stellen Sie sicher, dass der Wert für `namespace` in Ihrer Definition, welche Pods beim Start AWS Fargate verwenden, `eks-sample-app` ist.

```
kubectl create namespace eks-sample-app
```

Erstellen Sie eine Kubernetes-Bereitstellung.

Diese Beispiel-Bereitstellung ruft ein Container-Image aus einem öffentlichen Repository ab und stellt drei Replikate (individuelle Pods) davon in Ihrem Cluster bereit. Weitere Informationen finden Sie unter Bereitstellungen in der Kubernetes-Dokumentation.

1. Speichern Sie den folgenden Inhalt in einer Datei mit dem Namen `eks-sample-deployment.yaml`. Die Container in der Beispielanwendung verwenden keinen Netzwerkspeicher, aber möglicherweise haben Sie Anwendungen, die dies benötigen. Weitere Informationen finden Sie unter Anwendungsdatenspeicher.
   - Der Eintrag `kubernetes.io/os: windows` `nodeSelector` bedeutet, dass, wenn Sie beispielsweise Windows- und Linux-Knoten in Ihrem Cluster hätten, das Image nur auf Linux-Knoten bereitgestellt würde. Weitere Informationen finden Sie unter Bekannte Labels, Anmerkungen und Taints in der Kubernetes-Dokumentation.

   ```
   apiVersion: apps/v1
   kind: Deployment
   ```

```
metadata:
  name: eks-sample-windows-deployment
  namespace: eks-sample-app
  labels:
    app: eks-sample-windows-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: eks-sample-windows-app
  template:
    metadata:
      labels:
        app: eks-sample-windows-app
    spec:
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
              - key: kubernetes.io/arch
                operator: In
                values:
                - amd64
      containers:
      - name: windows-server-iis
        image: mcr.microsoft.com/windows/servercore:ltsc2019
        ports:
        - name: http
          containerPort: 80
        imagePullPolicy: IfNotPresent
        command:
        - powershell.exe
        - -command
        - "Add-WindowsFeature Web-Server; Invoke-WebRequest -UseBasicParsing
 -Uri 'https://dotnetbinaries.blob.core.windows.net/servicemonitor/2.0.1.6/
ServiceMonitor.exe' -OutFile 'C:\\ServiceMonitor.exe'; echo '<html><body><br/
><br/><marquee><H1>Hello EKS!!!<H1><marquee></body><html>' > C:\\inetpub\\wwwroot\
\default.html; C:\\ServiceMonitor.exe 'w3svc'; "
      nodeSelector:
        kubernetes.io/os: windows
```

2. Wenden Sie das Bereitstellungs-Manifest auf Ihren Cluster an.

```
kubectl apply -f eks-sample-deployment.yaml
```

Erstellen eines Services

Mit einem Service können Sie über eine einzige IP-Adresse oder einen einzigen Namen auf alle Replikate zugreifen. Weitere Informationen dazu finden Sie unter Service in der Kubernetes-Dokumentation. Obwohl in der Beispielanwendung nicht implementiert, empfehlen wir Ihnen, Kubernetes-Servicekonten für Ihre Pods zu erstellen und sie AWS-IAM-Konten zuzuordnen, wenn Sie über Anwendungen verfügen, die mit anderen AWS-Services interagieren müssen. Durch die Angabe von Dienstkonten verfügen Ihre Pods nur über die Mindestberechtigungen, die Sie für die Interaktion mit anderen Diensten angeben. Weitere Informationen finden Sie unter the section called “Anmeldeinformationen mit IRSA”.

1. Speichern Sie die folgenden Inhalte in einer Datei namens `eks-sample-service.yaml`. Kubernetes weist dem Service eine eigene IP-Adresse zu, auf die nur innerhalb des Clusters zugegriffen werden kann. Um von außerhalb des Clusters auf den Service zuzugreifen, stellen Sie den AWS Load Balancer Controller bereit, um den Anwendungs- oder Netzwerk-Datenverkehr an den Service auszugleichen.

```
apiVersion: v1
kind: Service
metadata:
  name: eks-sample-windows-service
  namespace: eks-sample-app
  labels:
    app: eks-sample-windows-app
spec:
  selector:
    app: eks-sample-windows-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

2. Wenden Sie das Service-Manifest auf Ihren Cluster an.

```
kubectl apply -f eks-sample-service.yaml
```

Erstellte Ressourcen überprüfen

1. Zeigen Sie alle im `eks-sample-app`-Namespace vorhandenen Ressourcen an.

```
kubectl get all -n eks-sample-app
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME                                                 READY   STATUS    RESTARTS   AGE
pod/eks-sample-windows-deployment-65b7669776-m6qxz   1/1     Running   0          27m
pod/eks-sample-windows-deployment-65b7669776-mmxvd   1/1     Running   0          27m
pod/eks-sample-windows-deployment-65b7669776-qzn22   1/1     Running   0          27m

NAME                                 TYPE         CLUSTER-IP      EXTERNAL-IP   PORT(S)
   AGE
service/eks-sample-windows-service   ClusterIP    10.100.74.8     <none>        80/
TCP    32m

NAME                                            READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/eks-sample-windows-deployment 3/3     3            3           27m

NAME                                                       DESIRED   CURRENT   READY
 AGE
replicaset.apps/eks-sample-windows-deployment-776d8f8fd8   3         3         3
   27m
```

In der Ausgabe sehen Sie den Service und die Bereitstellung, die in den Beispielmanifesten, der vorherigen Schritte, angegeben wurden. Sie sehen auch drei Pods. Dies liegt daran, dass 3 `replicas` im Beispielmanifest angegeben wurden. Weitere Informationen zu Pods finden Sie unter Pods in der Kubernetes-Dokumentation. Kubernetes erstellt automatisch die `replicaset`-Ressource, obwohl sie nicht in den Beispiel-Manifesten angegeben ist. Weitere Informationen zu `ReplicaSets` finden Sie unter ReplicaSet in der Kubernetes-Dokumentation.

> **Note**
>
> Kubernetes behält die Anzahl der Replikate bei, die im Manifest angegeben sind. Wenn es sich um eine Produktionsbereitstellung handelt und Sie möchten, dass Kubernetes die Anzahl der Replikate horizontal skaliert oder die Rechenressourcen für die Pods vertikal

skaliert, verwenden Sie dazu die Optionen Pod-Bereitstellungen mit horizontalem Pod-Autoscaler skalieren und Pod-Ressourcen mit vertikalem Pod-Autoscaler anpassen.

2. Zeigen Sie die Details des bereitgestellten Services an.

```
kubectl -n eks-sample-app describe service eks-sample-windows-service
```

Eine Beispielausgabe sieht wie folgt aus.

```
Name:              eks-sample-windows-service
Namespace:         eks-sample-app
Labels:            app=eks-sample-windows-app
Annotations:       <none>
Selector:          app=eks-sample-windows-app
Type:              ClusterIP
IP Families:       <none>
IP:                10.100.74.8
IPs:               10.100.74.8
Port:              <unset>  80/TCP
TargetPort:        80/TCP
Endpoints:         192.168.24.212:80,192.168.50.185:80,192.168.63.93:80
Session Affinity:  None
Events:            <none>
```

In der vorherigen Ausgabe ist der Wert für `IP:` eine eindeutige IP-Adresse, die von jedem Knoten oder Pod innerhalb des Clusters aus erreicht werden kann, jedoch nicht von außerhalb des Clusters. Die Werte für `Endpoints` sind IP-Adressen, die innerhalb Ihrer VPC den Pods zugewiesen werden, die Teil des Services sind.

3. Zeigen Sie die Details eines der in der Ausgabe aufgeführten Pods an, wenn Sie in einem vorherigen Schritt den Namespace angezeigt haben. Ersetzen Sie *776d8f8fd8-78w66* durch den für einen Ihrer Pods zurückgegebenen Wert.

```
kubectl -n eks-sample-app describe pod eks-sample-windows-deployment-65b7669776-m6qxz
```

Gekürzte Beispielausgabe

```
Name:         eks-sample-windows-deployment-65b7669776-m6qxz
Namespace:    eks-sample-app
Priority:     0
```

```
Node:         ip-192-168-45-132.us-west-2.compute.internal/192.168.45.132
[...]
IP:           192.168.63.93
IPs:
  IP:           192.168.63.93
Controlled By:  ReplicaSet/eks-sample-windows-deployment-65b7669776
[...]
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
[...]
Events:
  Type    Reason     Age    From
 Message
  ----    ------     ----   ----
 -------
  Normal  Scheduled  3m20s  default-scheduler
 Successfully assigned eks-sample-app/eks-sample-windows-deployment-65b7669776-m6qxz
 to ip-192-168-45-132.us-west-2.compute.internal
[...]
```

In der vorherigen Ausgabe ist der Wert für `IP:` eine eindeutige IP, die dem Pod aus dem CIDR-Block zugewiesen wird, der dem Subnetz zugewiesen ist, in dem sich der Knoten befindet. Wenn Sie es vorziehen, Pods IP-Adressen aus verschiedenen CIDR-Blöcken zuzuweisen, können Sie das Standardverhalten ändern. Weitere Informationen finden Sie unter the section called “Benutzerdefinierte Netzwerke”. Sie können auch sehen, dass der Kubernetes-Scheduler den Pod in Node mit der IP-Adresse *192.168.45.132* geplant hat.

> **Tip**
>
> Anstatt die Befehlszeile zu verwenden, können Sie viele Details zu Pods, Services, Bereitstellungen und anderen Kubernetes-Ressourcen in der AWS-Managementkonsole anzeigen. Weitere Informationen finden Sie unter the section called “Zugriff auf Cluster-Ressourcen”.

Shell in einem Pod ausführen

1. Führen Sie eine Shell auf dem Pod aus, die Sie im vorherigen Schritt beschrieben haben, und ersetzen Sie *65b7669776-m6qxz* mit der ID eines Ihrer Pods.

```
kubectl exec -it eks-sample-windows-deployment-65b7669776-m6qxz -n eks-sample-app --
 powershell.exe
```

2. Zeigen Sie in der Pod-Shell die Ausgabe des Web-Servers an, der in einem vorherigen Schritt mit Ihrer Bereitstellung installiert wurde. Sie müssen nur den Servicenamen angeben. Er wird von CoreDNS, das mit einem Amazon-EKS-Cluster bereitgestellt wird, standardmäßig in die IP-Adresse des Services aufgelöst.

```
Invoke-WebRequest -uri eks-sample-windows-service/default.html -UseBasicParsing
```

   Eine Beispielausgabe sieht wie folgt aus.

```
StatusCode        : 200
StatusDescription : OK
Content           : < h t m l > < b o d y > < b r / > < b r / > < m a r q u e e > < H
 1 > H e l l o
                      E K S ! ! ! < H 1 > < m a r q u e e > < / b o d y > < h t m l >
```

3. Zeigen Sie in der Pod-Shell den DNS-Server für den Pod an.

```
Get-NetIPConfiguration
```

   Gekürzte Ausgabe

```
InterfaceAlias       : vEthernet
[...]
IPv4Address          : 192.168.63.14
[...]
DNSServer            : 10.100.0.10
```

   In der vorherigen Ausgabe wird `10.100.0.10` automatisch als DNS-Server für alle Pods zugewiesen, die im Cluster bereitgestellt werden.

4. Trennen Sie die Verbindung mit dem Pod, indem Sie `exit` eingeben.

5. Wenn Sie die Beispielanwendung nicht mehr benötigen, können Sie den Beispiel-Namespace, den Service und die Bereitstellung mit dem folgenden Befehl entfernen.

```
kubectl delete namespace eks-sample-app
```

Nächste Schritte

Nachdem Sie die Beispielanwendung bereitgestellt haben, können Sie einige der folgenden Übungen ausprobieren:

- Anwendungen und HTTP-Datenverkehr mit Application Load Balancers weiterleiten
- Weiterleitung von TCP- und UDP-Datenverkehr mit Network Load Balancers

Pod-Ressourcen mit Vertical Pod Autoscaler anpassen

Der Kubernetes Vertical Pod Autoscaler passt die CPU- und Speicherreservierungen für Ihre Pods automatisch an, um Ihre Anwendungen auf die „richtige Größe“ zu skalieren. Diese Anpassung kann die Cluster-Ressourcennutzung verbessern und CPU und Arbeitsspeicher für andere Pods freigeben. In diesem Thema können Sie den Vertical Pod Autoscaler für Ihren Cluster bereitstellen und überprüfen, ob er funktioniert.

- Sie haben einen Amazon EKS-Cluster Falls nicht, finden Sie weitere Informationen unter Erste Schritte.
- Sie haben den Kubernetes-Metrik-Server installiert. Weitere Informationen finden Sie unter the section called “Kennzahlen-Server”.
- Sie verwenden einen `kubectl`-Client, der für die Kommunikation mit Ihrem Amazon EKS-Cluster konfiguriert ist.
- OpenSSL `1.1.1` oder höher ist auf Ihrem Gerät installiert.

Bereitstellen des Vertical Pod Autoscalers

In diesem Abschnitt stellen Sie den Vertical Pod Autoscaler für Ihren Cluster bereit.

1. Öffnen Sie ein Terminalfenster und navigieren Sie zu einem Verzeichnis, in das Sie den Vertical Pod Autoscaler-Quellcode herunterladen möchten.

2. Klonen Sie das GitHub-Repository kubernetes/autoscaler.

```
git clone https://github.com/kubernetes/autoscaler.git
```

3. Wechseln Sie in das `vertical-pod-autoscaler`-Verzeichnis.

```
cd autoscaler/vertical-pod-autoscaler/
```

4. (Optional) Wenn Sie bereits eine andere Version des Vertical Pod Autoscalers bereitgestellt haben, entfernen Sie ihn mit dem folgenden Befehl.

```
./hack/vpa-down.sh
```

5. Wenn Ihre Knoten keinen Internetzugriff auf die Container-Registry `registry.k8s.io` haben, müssen Sie die folgenden Images abrufen und in ein eigenes privates Repository verschieben. Weitere Informationen zum Abrufen und Verschieben der Images in ein eigenes privates Repository finden Sie unter the section called “Kopieren eines Images in ein Repository”.

```
registry.k8s.io/autoscaling/vpa-admission-controller:0.10.0
registry.k8s.io/autoscaling/vpa-recommender:0.10.0
registry.k8s.io/autoscaling/vpa-updater:0.10.0
```

   Wenn Sie die Images in ein privates Amazon-ECR-Repository verschieben, ersetzen Sie `registry.k8s.io` in den Manifesten durch Ihr Registry. Ersetzen Sie *`111122223333`* durch Ihre Konto-ID. Ersetzen Sie den *`region-code`* durch die AWS-Region, in der sich Ihr Cluster befindet. Beim folgenden Befehl wird davon ausgegangen, dass Sie Ihr Repository genauso benannt haben wie das Repository im Manifest. Wenn Sie das Repository anders benannt haben, müssen Sie es ebenfalls ändern.

```
sed -i.bak -e 's/registry.k8s.io/111122223333.dkr.ecr.region-code.amazonaws.com/' ./
deploy/admission-controller-deployment.yaml
sed -i.bak -e 's/registry.k8s.io/111122223333.dkr.ecr.region-code.amazonaws.com/' ./
deploy/recommender-deployment.yaml
sed -i.bak -e 's/registry.k8s.io/111122223333.dkr.ecr.region-code.amazonaws.com/' ./
deploy/updater-deployment.yaml
```

6. Stellen Sie den Vertical Pod Autoscaler mit dem folgenden Befehl für Ihren Cluster bereit.

```
./hack/vpa-up.sh
```

7. Überprüfen Sie, ob die Vertical Pod Autoscaler-Pods erfolgreich erstellt wurden.

```
kubectl get pods -n kube-system
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME                                        READY   STATUS    RESTARTS   AGE
[...]
metrics-server-8459fc497-kfj8w              1/1     Running   0          83m
vpa-admission-controller-68c748777d-ppspd   1/1     Running   0          7s
vpa-recommender-6fc8c67d85-gljpl            1/1     Running   0          8s
vpa-updater-786b96955c-bgp9d                1/1     Running   0          8s
```

Testen der Installation von Vertical Pod Autoscaler

In diesem Abschnitt stellen Sie eine Beispielanwendung bereit, um zu überprüfen, ob der Vertical Pod Autoscaler funktioniert.

1. Stellen Sie das Vertical Pod Autoscaler-Beispiel `hamster.yaml` mit dem folgenden Befehl bereit.

```
kubectl apply -f examples/hamster.yaml
```

2. Rufen Sie die Pods aus der `hamster`-Beispielanwendung ab.

```
kubectl get pods -l app=hamster
```

Eine Beispielausgabe sieht wie folgt aus.

```
hamster-c7d89d6db-rglf5   1/1     Running   0          48s
hamster-c7d89d6db-znvz5   1/1     Running   0          48s
```

3. Beschreiben Sie einen der Pods, um die zugehörige `cpu`- und `memory`-Reservierung anzuzeigen. Ersetzen Sie *`c7d89d6db-rglf5`* durch eine der IDs, die in der Ausgabe im vorherigen Schritt zurückgegeben wurden.

```
kubectl describe pod hamster-c7d89d6db-rglf5
```

Eine Beispielausgabe sieht wie folgt aus.

```
[...]
Containers:
  hamster:
    Container ID:  docker://
e76c2413fc720ac395c33b64588c82094fc8e5d590e373d5f818f3978f577e24
    Image:         registry.k8s.io/ubuntu-slim:0.1
    Image ID:      docker-pullable://registry.k8s.io/ubuntu-
slim@sha256:b6f8c3885f5880a4f1a7cf717c07242eb4858fdd5a84b5ffe35b1cf680ea17b1
    Port:          <none>
    Host Port:     <none>
    Command:
      /bin/sh
    Args:
      -c
      while true; do timeout 0.5s yes >/dev/null; sleep 0.5s; done
    State:          Running
      Started:      Fri, 27 Sep 2019 10:35:16 -0700
    Ready:          True
    Restart Count:  0
    Requests:
      cpu:        100m
      memory:     50Mi
[...]
```

Sie können sehen, dass der ursprüngliche Pod 100 Milli-CPU CPU und 50 Mebibyte Speicher reserviert. Für diese Beispielanwendung ist 100 Milli-CPU weniger als der Pod zum Ausführen benötigt, sodass er CPU-eingeschränkt ist. Außerdem reserviert er viel weniger Arbeitsspeicher, als er benötigt. Die Bereitstellung des Vertical Pod Autoscaler `vpa-recommender` analysiert die Hamster-Pods, um festzustellen, ob die CPU- und Speicheranforderungen angemessen sind. Wenn Anpassungen erforderlich sind, werden die Pods mit aktualisierten Werten aus `vpa-updater` neu geladen.

4. Warten Sie, bis `vpa-updater` neue Hamster-Pods startet. Dies sollte ein oder zwei Minuten dauern. Sie können die Pods mit dem folgenden Befehl überwachen.

> **Note**
>
> Wenn Sie nicht sicher sind, ob ein neuer Pod gestartet wurde, vergleichen Sie die Pod-Namen mit Ihrer vorherigen Liste. Wenn der neue Pod gestartet wird, wird ein neuer Pod-Name angezeigt.

```
kubectl get --watch Pods -l app=hamster
```

5. Wenn ein neuer Hamster-Pod gestartet wird, beschreiben Sie ihn und zeigen Sie die aktualisierten CPU- und Speicherreservierungen an.

```
kubectl describe pod hamster-c7d89d6db-jxgfv
```

Eine Beispielausgabe sieht wie folgt aus.

```
[...]
Containers:
  hamster:
    Container ID:
 docker://2c3e7b6fb7ce0d8c86444334df654af6fb3fc88aad4c5d710eac3b1e7c58f7db
    Image:         registry.k8s.io/ubuntu-slim:0.1
    Image ID:      docker-pullable://registry.k8s.io/ubuntu-
slim@sha256:b6f8c3885f5880a4f1a7cf717c07242eb4858fdd5a84b5ffe35b1cf680ea17b1
    Port:          <none>
    Host Port:     <none>
    Command:
      /bin/sh
    Args:
      -c
      while true; do timeout 0.5s yes >/dev/null; sleep 0.5s; done
    State:          Running
      Started:      Fri, 27 Sep 2019 10:37:08 -0700
    Ready:          True
    Restart Count:  0
    Requests:
      cpu:        587m
      memory:     262144k
[...]
```

In der vorherigen Ausgabe können Sie feststellen, dass sich die `cpu`-Reservierung auf 587 Millicpu erhöht hat, was mehr als dem Fünffachen des ursprünglichen Werts entspricht. Der `memory` wurde auf 262.144 Kilobyte erhöht, was etwa 250 Mebibyte bzw. dem Fünffachen des ursprünglichen Werts entspricht. Diesem Pod standen zu wenige Ressourcen zur Verfügung. Daher hat der Vertical Pod Autoscaler unsere Schätzung mit einem wesentlich geeigneteren Wert korrigiert.

6. Beschreiben Sie die `hamster-vpa`-Ressource, um die neue Empfehlung anzuzeigen.

```
kubectl describe vpa/hamster-vpa
```

Eine Beispielausgabe sieht wie folgt aus.

```
Name:         hamster-vpa
Namespace:    default
Labels:       <none>
Annotations:  kubectl.kubernetes.io/last-applied-configuration:
                {"apiVersion":"autoscaling.k8s.io/
v1beta2","kind":"VerticalPodAutoscaler","metadata":{"annotations":{},"name":"hamster-
vpa","namespace":"d...
API Version:  autoscaling.k8s.io/v1beta2
Kind:         VerticalPodAutoscaler
Metadata:
  Creation Timestamp:  2019-09-27T18:22:51Z
  Generation:          23
  Resource Version:    14411
  Self Link:           /apis/autoscaling.k8s.io/v1beta2/namespaces/default/
verticalpodautoscalers/hamster-vpa
  UID:                 d0d85fb9-e153-11e9-ae53-0205785d75b0
Spec:
  Target Ref:
    API Version:  apps/v1
    Kind:         Deployment
    Name:         hamster
Status:
  Conditions:
    Last Transition Time:  2019-09-27T18:23:28Z
    Status:                True
    Type:                  RecommendationProvided
  Recommendation:
    Container Recommendations:
      Container Name:  hamster
      Lower Bound:
        Cpu:     550m
        Memory:  262144k
      Target:
        Cpu:     587m
        Memory:  262144k
      Uncapped Target:
        Cpu:     587m
        Memory:  262144k
```

```
      Upper Bound:
        Cpu:     21147m
        Memory:  387863636
Events:          <none>
```

7. Wenn Sie mit dem Experimentieren mit der Beispielanwendung fertig sind, können Sie sie mit dem folgenden Befehl löschen.

```
kubectl delete -f examples/hamster.yaml
```

Skalierung von Pod-Bereitstellungen mit Horizontal Pod Autoscaler

Der Horizontal Pod Autoscaler von Kubernetes skaliert automatisch die Anzahl der Pods in einer Bereitstellung, einem Replikations-Controller oder einem Replikatsatz basierend auf der CPU-Auslastung dieser Ressource. Auf diese Weise können Ihre Anwendungen aufskalieren, um den erhöhten Bedarf zu erfüllen, oder abskalieren, wenn Ressourcen nicht benötigt werden, sodass Ihre Knoten für andere Anwendungen freigegeben werden. Wenn Sie einen CPU-Auslastungsprozentsatz festlegen, skaliert der Horizontal Pod Autoscaler Ihre Anwendung in beide Richtungen, um zu versuchen, dieses Ziel zu erreichen.

Der Horizontal Pod Autoscaler ist eine Standard-API-Ressource in Kubernetes, die einfach erfordert, dass eine Metrikquelle (z. B. der Kubernetes-Metrikserver) auf Ihrem Amazon-EKS-Cluster installiert ist. Sie müssen den Horizontal Pod Autoscaler nicht auf Ihrem Cluster bereitstellen oder installieren, um mit der Skalierung Ihrer Anwendungen zu beginnen. Weitere Informationen finden Sie unter Horizontal Pod Autoscaler in der Kubernetes-Dokumentation.

Verwenden Sie dieses Thema, um den Horizontal Pod Autoscaler für Ihren Amazon-EKS-Cluster vorzubereiten und zu überprüfen, ob er mit einer Beispielanwendung funktioniert.

> **Note**
>
> Dieses Thema basiert auf der Anleitung zum Horizontal Pod Autoscaler in der Kubernetes-Dokumentation.

- Sie haben einen vorhandenen Amazon-EKS-Cluster. Falls nicht, finden Sie weitere Informationen unter Erste Schritte.

- Sie haben den Kubernetes-Metrik-Server installiert. Weitere Informationen finden Sie unter the section called “Kennzahlen-Server”.
- Sie verwenden einen `kubectl`-Client, der für die Kommunikation mit Ihrem Amazon EKS-Cluster konfiguriert ist.

Ausführen einer Horizontal Pod Autoscaler-Testanwendung

In diesem Abschnitt stellen Sie eine Beispielanwendung bereit, um zu überprüfen, ob der Horizontal Pod Autoscaler funktioniert.

> **Note**
>
> Dieses Beispiel basiert auf der Anleitung zum Horizontal Pod Autoscaler in der Kubernetes-Dokumentation.

1. Stellen Sie mit dem folgenden Befehl eine einfache Apache-Webserveranwendung bereit.

   ```
   kubectl apply -f https://k8s.io/examples/application/php-apache.yaml
   ```

   Dieser Apache-Webserver-Pod verfügt über eine CPU-Begrenzung von 500 Millicpu und ist auf Port 80 erreichbar.
2. Erstellen Sie eine Horizontal Pod Autoscaler-Ressource für die `php-apache`-Bereitstellung.

   ```
   kubectl autoscale deployment php-apache --cpu-percent=50 --min=1 --max=10
   ```

   Dieser Befehl erstellt einen Autoscaler, der 50 Prozent CPU-Auslastung für die Bereitstellung anvisiert, mit mindestens einem Pod und maximal zehn Pods. Wenn die durchschnittliche CPU-Auslastung unter 50 Prozent liegt, versucht der Autoscaler, die Anzahl der Pods in der Bereitstellung auf minimal eins reduzieren. Wenn die Last größer als 50 Prozent ist, versucht der Autoscaler, die Anzahl der Pods in der Bereitstellung auf maximal zehn zu erhöhen. Weitere Informationen finden Sie unter Wie funktioniert ein Horizontal Pod Autoscaler? in der Kubernetes-Dokumentation.
3. Beschreiben Sie den Autoscaler mit dem folgenden Befehl, um seine Details anzuzeigen.

   ```
   kubectl get hpa
   ```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME         REFERENCE                TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
php-apache   Deployment/php-apache    0%/50%    1         10        1          51s
```

Wie Sie sehen, ist die aktuelle CPU-Last `0%`, weil es noch keine Last auf dem Server gibt. Die Pod-Anzahl befindet sich bereits an der untersten Grenze (eins), daher kann sie nicht weiter nach unten skaliert werden.

4. Erstellen Sie eine Last für den Webserver, indem Sie einen Container ausführen.

```
kubectl run -i \
    --tty load-generator \
    --rm --image=busybox \
    --restart=Never \
    -- /bin/sh -c "while sleep 0.01; do wget -q -O- http://php-apache; done"
```

5. Um die Skalierung der Bereitstellung zu beobachten, führen Sie in regelmäßigen Abständen den folgenden Befehl in einem anderen Terminal als demjenigen aus, in dem Sie den vorherigen Schritt ausgeführt haben.

```
kubectl get hpa php-apache
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME         REFERENCE                TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
php-apache   Deployment/php-apache    250%/50%   1         10        5          4m44s
```

Es kann mehr als eine Minute dauern, bis die Anzahl der Replikate zunimmt. Solange der tatsächliche CPU-Prozentsatz höher als der Zielprozentsatz ist, erhöht sich die Replikatanzahl auf bis zu 10. In diesem Fall ist dies `250%`, so dass die Anzahl der `REPLICAS` weiter steigt.

> **Note**
>
> Es kann einige Minuten dauern, bis die Replikatanzahl ihr Maximum erreicht. Wenn beispielsweise nur 6 Replikate erforderlich sind, damit die CPU-Last bei oder unter 50 % bleibt, wird die Last nicht über 6 Replikate hinaus skaliert.

6. Stoppen Sie die Last. Halten Sie im Terminal-Fenster, in dem Sie die Last erzeugen, die Last an, indem Sie die `Ctrl+C`-Tasten gedrückt halten. Sie können beobachten, wie die Replikate auf 1 zurückskalieren, indem Sie den folgenden Befehl erneut in dem Terminal ausführen, in dem Sie die Skalierung beobachten.

```
kubectl get hpa
```

   Eine Beispielausgabe sieht wie folgt aus.

```
NAME         REFERENCE                  TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
php-apache   Deployment/php-apache      0%/50%    1         10        1          25m
```

   > **Note**
   >
   > Der Standardzeitrahmen für das Zurückskalieren beträgt fünf Minuten. Es wird daher einige Zeit dauern, bis die Replikatanzahl wieder 1 erreicht, selbst wenn der aktuelle CPU-Prozentsatz 0 Prozent beträgt. Der Zeitrahmen ist veränderbar. Weitere Informationen finden Sie unter Horizontal Pod Autoscaler in der Kubernetes-Dokumentation.

7. Wenn Sie mit dem Experimentieren mit Ihrer Beispielanwendung fertig sind, löschen Sie die `php-apache`-Ressourcen.

```
kubectl delete deployment.apps/php-apache service/php-apache
 horizontalpodautoscaler.autoscaling/php-apache
```

Weiterleitung von TCP- und UDP-Datenverkehr mit Network Load Balancers

> **Note**
>
> Neu: Amazon EKS Auto Mode automatisiert Routineaufgaben für den Lastausgleich. Weitere Informationen finden Sie unter:
>
> - the section called “Load Balancer bereitstellen”
> - the section called “Service erstellen”

Das Load Balancing des Netzwerkverkehrs erfolgt auf L4 des OSI-Modells. Um den Anwendungsdatenverkehr unter zu verteilenL7, stellen Sie ein Kubernetes bereitingress, das einen AWS Application Load Balancer bereitstellt. Weitere Informationen finden Sie unter the section called “Application Load Balancing”. Weitere Informationen zu den Unterschieden zwischen den beiden Arten von Load Balancing finden Sie auf der AWS Website unter Elastic Load Balancing Balancing-Funktionen.

Wenn Sie einen Kubernetes-Typ erstellenLoadBalancer, erstellt Service der Load Balancer-Controller des AWS Cloud-Anbieters standardmäßig AWSClassic Load Balancer, kann aber auch Network Load Balancer erstellen AWS . Dieser Controller erhält in Zukunft nur kritische Fehlerbehebungen. Weitere Informationen zur Verwendung des AWS Cloud-Provider-Loadbalancers finden Sie unter Cloud-Provider-Load Balancer-Controller in der AWS Kubernetes-Dokumentation. Seine Verwendung wird in diesem Thema nicht behandelt.

Wir empfehlen, dass Sie Version 2.7.2 oder höher des Load AWS Balancer Controllers anstelle des AWS Cloud-Provider-Load Balancer-Controllers verwenden. Der Load AWS Balancer Controller erstellt AWS Network Load Balancer, aber keine AWS Classic Load Balancer. Der Rest dieses Themas befasst sich mit der Verwendung des Load AWS Balancer Controllers.

Ein AWS Network Load Balancer kann den Netzwerkverkehr auf Pods verteilen, die auf Amazon EC2 IP- und Instance-Zielen bereitgestellt werden, auf AWS Fargate-IP-Ziele oder auf Amazon EKS-Hybridknoten als IP-Ziele. Weitere Informationen finden Sie unter AWS Load Balancer Controller on GitHub.

Voraussetzungen

Bevor Sie mit dem Load Balancer Controller einen AWS Lastenausgleich für den Netzwerkverkehr durchführen können, müssen Sie die folgenden Anforderungen erfüllen.

- Ein Cluster ist vorhanden. Falls Sie keinen vorhandenen Cluster haben, siehe Erste Schritte. Wenn Sie die Version eines vorhandenen Clusters aktualisieren müssen, finden Sie Informationen dazu unter the section called “Kubernetes-Version aktualisieren”.
- Lassen Sie den AWS Load Balancer Controller auf Ihrem Cluster bereitstellen. Weitere Informationen finden Sie unter the section called “ AWS Load Balancer Balancer-Controller”. Wir empfehlen Version 2.7.2 oder höher.
- Mindestens ein Subnetz. Werden in einer Availability Zone mehrere markierte Subnetze gefunden, wählt der Controller das erste Subnetz, dessen Subnetz-ID lexikografisch an erster Stelle steht. Das Subnetz muss mindestens acht verfügbare IP-Adressen aufweisen.

- Wenn Sie die Load AWS Balancer Controller-Version `2.1.1` oder eine frühere Version verwenden, müssen Subnetze wie folgt gekennzeichnet werden. Bei Verwendung der Version `2.1.2` oder höher ist dieses Tag optional. Möglicherweise möchten Sie ein Subnetz taggen, wenn mehrere Cluster in derselben VPC ausgeführt werden, oder wenn mehrere AWS Dienste Subnetze in einer VPC gemeinsam nutzen, und Sie mehr Kontrolle darüber haben möchten, wo Load Balancer für jeden Cluster bereitgestellt werden. Wenn Sie das Subnetz explizit IDs als Anmerkung zu einem Serviceobjekt angeben, verwenden Kubernetes und der Load AWS Balancer Controller diese Subnetze direkt, um den Load Balancer zu erstellen. Subnetz-Tagging ist nicht erforderlich, wenn Sie diese Methode für die Bereitstellung von Load Balancern verwenden. In diesem Fall können Sie die folgenden Anforderungen für das Tagging privater und öffentlicher Subnetze überspringen Ersetzen Sie *`my-cluster`* mit Ihrem Clusternamen.
  - Schlüssel – `kubernetes.io/cluster/<my-cluster>`
  - Wert – `shared` oder `owned`
- Ihre öffentlichen und privaten Subnetze müssen die folgenden Anforderungen erfüllen, es sei denn, Sie geben das Subnetz explizit IDs als Anmerkung zu einem Service- oder Eingangsobjekt an. Wenn Sie Load Balancer bereitstellen, indem Sie das Subnetz explizit IDs als Anmerkung zu einem Service- oder Ingress-Objekt angeben, verwenden Kubernetes und der Load AWS Balancer Controller diese Subnetze direkt, um den Load Balancer zu erstellen, und die folgenden Tags sind nicht erforderlich.
  - Private Subnetze – Müssen im folgenden Format markiert sein. Auf diese Weise wissen Kubernetes und der Load AWS Balancer Controller, dass die Subnetze für interne Load Balancer verwendet werden können. Wenn Sie eine Amazon AWS AWS CloudFormation EKS-Vorlage verwenden`eksctl`, um Ihre VPC nach dem 26. März 2020 zu erstellen, werden die Subnetze bei der Erstellung entsprechend gekennzeichnet. Weitere Informationen über die Amazon-EKS-AWS - AWS CloudFormation -VPC-Vorlagen finden Sie unter the section called “Erstellen einer VPC”.
    - Schlüssel – `kubernetes.io/role/internal-elb`
    - Wert – 1
  - Öffentliche Subnetze – Müssen im folgenden Format markiert sein. Auf diese Weise weiß Kubernetes, dass es nur diese Subnetze für externe Load Balancer verwenden muss, anstatt in jeder Availability Zone ein öffentliches Subnetz auszuwählen (basierend auf der lexikografischen Reihenfolge des Subnetzes). IDs Wenn Sie eine Amazon AWS CloudFormation EKS-Vorlage verwenden`eksctl`, um Ihre VPC nach dem 26. März 2020 zu erstellen, werden die Subnetze bei der Erstellung entsprechend gekennzeichnet. Weitere Informationen zu den Amazon AWS CloudFormation EKS-VPC-Vorlagen finden Sie unterthe section called “Erstellen einer VPC”.

- Schlüssel – `kubernetes.io/role/elb`
- Wert – 1

Wenn die Tags für Subnetz-Rollen nicht explizit hinzugefügt werden, prüft der Kubernetes-Service-Controller die Routing-Tabelle der Cluster-VPC-Subnetze, um festzustellen, ob das Subnetz privat oder öffentlich ist. Es wird empfohlen, dass Sie sich nicht auf dieses Verhalten verlassen und stattdessen explizit die privaten oder öffentlichen Rollen-Tags hinzufügen. Der Load AWS Balancer Controller untersucht keine Routing-Tabellen und setzt voraus, dass die privaten und öffentlichen Tags für eine erfolgreiche auto Erkennung vorhanden sind.

Überlegungen

- Die Konfiguration Ihres Load Balancers wird durch Anmerkungen gesteuert, die zu dem Manifest für Ihren Service hinzugefügt werden. Dienstanmerkungen unterscheiden sich bei Verwendung des Load AWS Balancer Controllers von denen bei Verwendung des AWS Cloud-Provider-Load Balancer-Controllers. Stellen Sie sicher, dass Sie die Anmerkungen für den Load AWS Balancer Controller überprüfen, bevor Sie Dienste bereitstellen.
- Bei Verwendung des Amazon VPC CNI-Plug-ins für Kubernetes kann der Load Balancer Controller einen AWS Lastausgleich auf Amazon EC2 IP- oder Instance-Ziele und Fargate-IP-Ziele durchführen. Bei Verwendung von Alternativen kompatiblen CNI-Plugins kann der Controller die Last nur auf Instance-Ziele ausgleichen, es sei denn, Sie führen das Load Balancing in Amazon EKS Hybrid Nodes durch. Bei Hybridknoten kann der Controller die IP-Ziele ausgleichen. Weitere Informationen zu Network-Load-Balancer-Zieltypen finden Sie unter Zieltyp im Benutzerhandbuch für Network Load Balancers.
- Wenn Sie dem Load Balancer Tags hinzufügen möchten, wenn oder nachdem er erstellt wurde, fügen Sie die folgende Annotation in Ihre Service-Spezifikation ein. Weitere Informationen finden Sie unter AWS Resource Tags in der Load AWS Balancer Controller-Dokumentation.

```
service.beta.kubernetes.io/aws-load-balancer-additional-resource-tags
```

- Sie können Elastic IP-Adressen zum Network Load Balancer hinzufügen, indem Sie die folgende Anmerkung hinzufügen. Ersetzen Sie die Beispielwerte durch die `Allocation IDs` Ihrer Elastic IP-Adressen. Die Anzahl der `Allocation IDs` muss mit der Anzahl der Subnetze übereinstimmen, die für den Load Balancer verwendet werden. Weitere Informationen finden Sie unter Ressourcen-Tags in der AWS -Load-Balancer-Controller-Dokumentation.

```
service.beta.kubernetes.io/aws-load-balancer-eip-allocations: eipalloc-
xxxxxxxxxxxxxxxx,eipalloc-yyyyyyyyyyyyyyyyy
```

- Für jeden Network Load Balancer, den Sie erstellen, fügt Amazon EKS der Sicherheitsgruppe des Knotens für Client-Datenverkehr eine eingehende Regel und für jedes Load-Balancer-Subnetz in der VPC für Integritätsprüfungen eine Regel hinzu. Bereitstellung eines Dienstes vom Typ`LoadBalancer`kann fehlschlagen, wenn Amazon EKS versucht, Regeln zu erstellen, die das Kontingent für die maximal zulässige Anzahl von Regeln für eine Sicherheitsgruppe überschreiten. Weitere Informationen dazu finden Sie im Abschnitt Sicherheitsgruppe in Amazon VPC-Kontingenten im Amazon VPC-Benutzerhandbuch. Berücksichtigen Sie die folgenden Optionen, um die Wahrscheinlichkeit zu minimieren, dass die maximale Anzahl von Regeln für eine Sicherheitsgruppe überschritten wird:
  - Fordern Sie eine Erhöhung Ihrer Regeln pro Sicherheitsgruppenkontingent an. Weitere Informationen dazu finden Sie unter Beantragen einer Kontingenterhöhung im Service-Quotas-Benutzerhandbuch.
  - Verwenden Sie IP-Ziele anstelle von Instance-Zielen. Mit IP-Zielen können Sie Regeln für dieselben Zielports freigeben. Sie können Load-Balancer-Subnetze manuell mit einer Anmerkung angeben. Weitere Informationen finden Sie unter Anmerkungen zu GitHub.
  - Verwenden Sie einen Eingang anstelle eines Services vom Typ `LoadBalancer`, um Datenverkehr an Ihren Service zu senden. Der AWS Application Load Balancer benötigt weniger Regeln als Network Load Balancer. Sie können einen ALB für mehrere Eingänge freigeben. Weitere Informationen finden Sie unter the section called “Application Load Balancing”. Sie können einen Network Load Balancer nicht über mehrere Services hinweg freigeben.
  - Stellen Sie Ihre Cluster für mehrere Konten bereit.
- Wenn Ihre Pods unter Windows in einem Amazon-EKS-Cluster ausgeführt werden, kann ein einzelner Service mit einem Load Balancer bis zu 1024 Backend-Pods unterstützen. Jeder Pod hat seine eigene eindeutige IP-Adresse.
- Wir empfehlen, nur neue Network Load Balancer mit dem Load AWS Balancer Controller zu erstellen. Der Versuch, vorhandene Network Load Balancer zu ersetzen, die mit dem Load Balancer-Controller des AWS Cloud-Anbieters erstellt wurden, kann zu mehreren Network Load Balancers führen, was zu Anwendungsausfällen führen kann.

Erstellen eines Network Load Balancers

Sie können einen Netzwerk-Load-Balancer mit IP- oder Instance-Zielen erstellen.

Network Load Balancer erstellen – IP-Ziele

- Sie können IP-Ziele mit Pods verwenden, die auf EC2 Amazon-Knoten, Fargate oder Amazon EKS-Hybridknoten bereitgestellt werden. Ihr Kubernetes-Service muss als Typ `LoadBalancer` erstellt werden. Weitere Informationen finden Sie LoadBalancer in der Kubernetes-Dokumentation unter Type.

  Um einen Load Balancer zu erstellen, der IP-Ziele verwendet, fügen Sie die folgende Anmerkung zu einem Service-Manifest hinzu und stellen Sie den Dienst bereit. Der `external` `aws-load-balancer-type` Wert für veranlasst den Load AWS Balancer Controller und nicht den Load Balancer Controller des AWS Cloud-Anbieters, den Network Load Balancer zu erstellen. Sie können ein Beispiel-Service-Manifest mit den Anmerkungen anzeigen.

  ```
  service.beta.kubernetes.io/aws-load-balancer-type: "external"
  service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: "ip"
  ```

  > **Note**
  >
  > Wenn Sie Load Balancing auf `IPv6` Pods durchführen, fügen Sie die folgende Annotation hinzu. Load Balancing über `IPv6` funktioniert nur auf IP-Ziele, nicht auf Instance-Ziele. Ohne diese Anmerkung verläuft das Load Balancing über `IPv4`.

  ```
  service.beta.kubernetes.io/aws-load-balancer-ip-address-type: dualstack
  ```

  Network Load Balancer werden standardmäßig mit dem `internal` `aws-load-balancer-scheme` erstellt. Sie können Network Load Balancer in jedem Subnetz in der VPC Ihres Clusters starten, einschließlich Subnetze, die bei der Erstellung Ihres Clusters nicht angegeben wurden.

  Kubernetes untersucht die Routing-Tabelle auf Ihre Subnetze, um zu identifizieren, ob sie öffentlich oder privat sind. Öffentliche Subnetze verfügen über einen direkten Zugang zum Internet über ein Internet-Gateway, nicht aber private Subnetze.

  Wenn Sie einen Network Load Balancer in einem öffentlichen Subnetz für den Lastenausgleich auf EC2 Amazon-Knoten erstellen möchten (Fargate kann nur privat sein), geben Sie dies `internet-facing` mit der folgenden Anmerkung an:

```
service.beta.kubernetes.io/aws-load-balancer-scheme: "internet-facing"
```

> **Note**
>
> Aus Gründen der Abwärtskompatibilität wird `service.beta.kubernetes.io/aws-load-balancer-type: "nlb-ip"` weiterhin unterstützt. Wir empfehlen jedoch die Verwendung der vorherigen Anmerkungen für neue Load Balancer anstelle von `service.beta.kubernetes.io/aws-load-balancer-type: "nlb-ip"`.

> **Important**
>
> Bearbeiten Sie die Anmerkungen nicht, nachdem Sie Ihren Service erstellt haben. Wenn Sie es ändern müssen, löschen Sie das Service-Objekt und erstellen es erneut mit dem gewünschten Wert für diese Anmerkung.

Network Load Balancer erstellen – Instance-Ziele

- Der Load Balancer-Controller des AWS Cloud-Anbieters erstellt Network Load Balancer nur mit Instanzzielen. Version `2.2.0` und höher des Load AWS Balancer Controllers erstellen auch Network Load Balancer mit Instanzzielen. Wir empfehlen, ihn anstelle des Load Balancer-Controllers des AWS Cloud-Anbieters zu verwenden, um neue Network Load Balancer zu erstellen. Sie können Network Load Balancer Balancer-Instance-Ziele mit Pods verwenden, die auf EC2 Amazon-Knoten, aber nicht auf Fargate bereitgestellt werden. Um den Netzwerkverkehr über Pods auszugleichen, die in Fargate bereitgestellt werden, müssen Sie IP-Ziele verwenden.

  Um einen Network Load Balancer in einem privaten Subnetz bereitzustellen, muss Ihre Service-Spezifikation über die folgenden Anmerkungen verfügen. Sie können ein Beispiel-Service-Manifest mit den Anmerkungen anzeigen. Der `external` `aws-load-balancer-type` Wert für veranlasst den Load AWS Balancer Controller und nicht den Load Balancer Controller des AWS Cloud-Anbieters, den Network Load Balancer zu erstellen.

  ```
  service.beta.kubernetes.io/aws-load-balancer-type: "external"
  service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: "instance"
  ```

Network Load Balancer werden standardmäßig mit dem `internal aws-load-balancer-scheme` erstellt. Für interne Network Load Balancer muss Ihr Amazon-EKS-Cluster so konfiguriert werden, dass er mindestens ein privates Subnetz in Ihrer VPC verwendet. Kubernetes untersucht die Routing-Tabelle auf Ihre Subnetze, um zu identifizieren, ob sie öffentlich oder privat sind. Öffentliche Subnetze verfügen über einen direkten Zugang zum Internet über ein Internet-Gateway, nicht aber private Subnetze.

Wenn Sie einen Network Load Balancer in einem öffentlichen Subnetz für den Lastenausgleich auf EC2 Amazon-Knoten erstellen möchten, geben Sie dies `internet-facing` mit der folgenden Anmerkung an:

```
service.beta.kubernetes.io/aws-load-balancer-scheme: "internet-facing"
```

> **Important**
>
> Bearbeiten Sie die Anmerkungen nicht, nachdem Sie Ihren Service erstellt haben. Wenn Sie es ändern müssen, löschen Sie das Service-Objekt und erstellen es erneut mit dem gewünschten Wert für diese Anmerkung.

(Optional) Bereitstellen einer Beispielanwendung

- Mindestens ein öffentliches oder privates Subnetz in Ihrer Cluster-VPC.
- Lassen Sie den AWS Load Balancer Controller auf Ihrem Cluster bereitstellen. Weitere Informationen finden Sie unter the section called “ AWS Load Balancer Balancer-Controller”. Wir empfehlen Version `2.7.2` oder höher.

  1. Wenn Sie für Fargate bereitstellen, stellen Sie sicher, dass Sie über ein verfügbares privates Subnetz in Ihrer VPC verfügen, und erstellen Sie ein Fargate-Profil. Wenn Sie keine Bereitstellung für Fargate durchführen, überspringen Sie diesen Schritt. Sie können das Profil erstellen, indem Sie den folgenden Befehl ausführen oder in AWS-Managementkonsole die gleichen Werte für `name` und `namespace` verwenden, die im Befehl enthalten sind. Ersetzen Sie die Beispielwerte durch Ihre eigenen Werte.

     ```
     eksctl create fargateprofile \
         --cluster my-cluster \
         --region region-code \
     ```

```
  --name nlb-sample-app \
  --namespace nlb-sample-app
```

2. Bereitstellen einer Beispielanwendung

   a. Erstellen Sie einen Namespace für die Anwendung.

   ```
   kubectl create namespace nlb-sample-app
   ```

   b. Speichern Sie den folgenden Inhalt in einer Datei mit dem Namen `sample-deployment.yaml` auf Ihrem Computer.

   ```
   apiVersion: apps/v1
   kind: Deployment
   metadata:
     name: nlb-sample-app
     namespace: nlb-sample-app
   spec:
     replicas: 3
     selector:
       matchLabels:
         app: nginx
     template:
       metadata:
         labels:
           app: nginx
       spec:
         containers:
           - name: nginx
             image: public.ecr.aws/nginx/nginx:1.23
             ports:
               - name: tcp
                 containerPort: 80
   ```

   c. Wenden Sie die Manifestdatei auf Ihren Cluster an.

   ```
   kubectl apply -f sample-deployment.yaml
   ```

3. Erstellen Sie einen Service mit einem zum Internet offenen Network Load Balancer, der Load Balancing auf IP-Ziele vornimmt.

   a. Speichern Sie den folgenden Inhalt in einer Datei mit dem Namen `sample-service.yaml` file auf Ihrem Computer. Wenn Sie in Fargate-Knoten bereitstellen, entfernen Sie die

`service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing`-Zeile.

```
apiVersion: v1
kind: Service
metadata:
  name: nlb-sample-service
  namespace: nlb-sample-app
  annotations:
    service.beta.kubernetes.io/aws-load-balancer-type: external
    service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: ip
    service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing
spec:
  ports:
    - port: 80
      targetPort: 80
      protocol: TCP
  type: LoadBalancer
  selector:
    app: nginx
```

b. Wenden Sie die Manifestdatei auf Ihren Cluster an.

```
kubectl apply -f sample-service.yaml
```

4. Stellen Sie sicher, dass der Dienst bereitgestellt wurde.

```
kubectl get svc nlb-sample-service -n nlb-sample-app
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME             TYPE           CLUSTER-IP         EXTERNAL-IP
                                                 PORT(S)        AGE
sample-service   LoadBalancer   10.100.240.137   k8s-nlbsampl-nlbsampl-xxxxxxxxxx-
xxxxxxxxxxxxxxxx.elb.region-code.amazonaws.com  80:32400/TCP   16h
```

> **Note**
>
> Die Werte für *10.100.240.137* und *xxxxxxxxxx* - unterscheiden *xxxxxxxxxxxxxxxx* sich von der Beispielausgabe (sie gelten nur für Ihren Load

Balancer) und *us-west-2* können für Sie unterschiedlich sein, je nachdem, in welcher AWS Region sich Ihr Cluster befindet.

5. Öffne den Amazon EC2 AWS-Managementkonsole. Wählen Sie Target Groups (Zielgruppen) unter Load Balancing (Lastausgleich) im linken Navigationsbereich aus. Wählen Sie in der Spalte Name den Namen der Zielgruppe aus, wobei der Wert in der Spalte Load Balancer einem Teil des Namens in der `EXTERNAL-IP`-Spalte der Ausgabe im vorherigen Schritt entspricht. Sie wählen beispielsweise die Zielgruppe mit dem Namen `k8s-default-samplese-`*xxxxxxxxxx*, wenn Ihre Ausgabe mit der vorigen Ausgabe übereinstimmte. Der Zieltyp ist `IP`, weil dies im Beispiel-Manifest des Services angegeben wurde.
6. Wählen Sie Ihre Zielgruppe und danach die Registerkarte Ziele aus. Unter Registrierte Ziele sollten Sie drei IP-Adressen der drei Replikate sehen, die in einem vorherigen Schritt bereitgestellt wurden. Warten Sie, bis der Status aller Ziele fehlerfrei ist, bevor Sie fortfahren. Möglicherweise dauert es ein paar Minuten, bis alle Ziele `healthy` sind. Die Ziele können sich in einem `unhealthy`-Zustand befinden, bevor sie in einen `healthy`-Zustand wechseln.
7. Senden Sie Datenverkehr an den Dienst, indem Sie *xxxxxxxxxx-xxxxxxxxxxxxxxxx* und *us-west-2* durch die Werte ersetzen, die in der Ausgabe für einen vorherigen Schritt für `EXTERNAL-IP` zurückgegeben wurden. Wenn Sie in einem privaten Subnetz bereitgestellt haben, müssen Sie die Seite von einem Gerät in Ihrer VPC aus anzeigen, z. B. von einem Bastion-Host. Weitere Informationen finden Sie unter Linux-Bastion-Hosts in AWS.

   ```
   curl k8s-default-samplese-xxxxxxxxxx-xxxxxxxxxxxxxxxx.elb.region-code.amazonaws.com
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   <!DOCTYPE html>
   <html>
   <head>
   <title>Welcome to nginx!</title>
   [...]
   ```
8. Wenn Sie mit der Beispiel-Bereitstellung, dem Service und Namespace fertig sind, entfernen Sie sie.

   ```
   kubectl delete namespace nlb-sample-app
   ```

Anwendungen und HTTP-Datenverkehr mit Application Load Balancers weiterleiten

> **Note**
>
> Neu: Amazon EKS Auto Mode automatisiert Routineaufgaben für den Lastausgleich. Weitere Informationen finden Sie unter:
>
> - the section called “Load Balancer bereitstellen”
> - the section called “Ingress-Klasse erstellen”

Wenn Sie ein Kubernetes erstellen`ingress`, wird ein AWS Application Load Balancer (ALB) bereitgestellt, der den Anwendungsdatenverkehr ausgleicht. Weitere Informationen finden Sie unter Was ist ein Application Load Balancer? im Application Load Balancers-Benutzerhandbuch und Ingress in der Kubernetes-Dokumentation. ALBs kann mit Pods verwendet werden, die auf Knoten oder in AWS Fargate bereitgestellt werden. Sie können einen ALB für öffentliche oder private Subnetze bereitstellen.

Der Anwendungsverkehr wird bei `L7` des OSI-Modells ausgeglichen. Um den Netzwerkverkehr auf `L4` auszugleichen, stellen Sie einen Kubernetes-`service` vom Typ `LoadBalancer` bereit. Dieser Typ stellt einen AWS Network Load Balancer bereit. Weitere Informationen finden Sie unter the section called “Netzwerk-Load-Balancer”. Weitere Informationen zu den Unterschieden zwischen den beiden Arten von Load Balancing finden Sie auf der AWS Website unter Elastic Load Balancing Balancing-Funktionen.

Voraussetzungen

Bevor Sie den Anwendungsdatenverkehr auf eine Anwendung verteilen können, müssen Sie die folgenden Anforderungen erfüllen.

- Ein Cluster ist vorhanden. Falls Sie keinen vorhandenen Cluster haben, siehe Erste Schritte. Wenn Sie die Version eines vorhandenen Clusters aktualisieren müssen, finden Sie Informationen dazu unter the section called “Kubernetes-Version aktualisieren”.
- Lassen Sie den AWS Load Balancer Controller auf Ihrem Cluster bereitstellen. Weitere Informationen finden Sie unter the section called “ AWS Load Balancer Balancer-Controller”. Wir empfehlen Version `2.7.2` oder höher.

- Mindestens zwei Subnetze in verschiedenen Availability Zones. Der AWS Load Balancer Controller wählt aus jeder Availability Zone ein Subnetz aus. Wenn mehrere markierte Subnetze in einer Availability Zone gefunden werden, wählt der Controller das Subnetz aus, dessen Subnetz-ID lexikografisch an erster Stelle steht. Ein Subnetz muss jeweils mindestens acht verfügbare IP-Adressen aufweisen.

  Wenn Sie mehrere an den Worker-Knoten angefügte Sicherheitsgruppen verwenden, muss genau eine Sicherheitsgruppe wie folgt gekennzeichnet werden. Ersetzen Sie *my-cluster* mit Ihrem Clusternamen.

  - Schlüssel – `kubernetes.io/cluster/<my-cluster>`
  - Wert – `shared` oder `owned`
- Wenn Sie die Load AWS Balancer Controller-Version `2.1.1` oder eine frühere Version verwenden, müssen Subnetze im folgenden Format gekennzeichnet werden. Wenn Sie Version `2.1.2` oder höher verwenden, ist die Kennzeichnung optional. Wir empfehlen jedoch, ein Subnetz zu markieren, falls einer der folgenden Fälle zutrifft. Sie haben mehrere Cluster, die in derselben VPC ausgeführt werden, oder Sie haben mehrere AWS Dienste, die sich Subnetze in einer VPC teilen. Oder Sie möchten mehr Kontrolle darüber haben, wo Load Balancer für jeden Cluster bereitgestellt werden. Ersetzen Sie *my-cluster* durch Ihren Clusternamen.
  - Schlüssel – `kubernetes.io/cluster/<my-cluster>`
  - Wert – `shared` oder `owned`
- Ihre öffentlichen und privaten Subnetze müssen die folgenden Anforderungen erfüllen. Dies ist der Fall, es sei denn, Sie geben das Subnetz explizit IDs als Anmerkung zu einem Service- oder Eingangsobjekt an. Gehen Sie davon aus, dass Sie Load Balancer bereitstellen, indem Sie das Subnetz explizit IDs als Anmerkung zu einem Dienst- oder Eingangsobjekt angeben. In dieser Situation verwenden Kubernetes und der AWS Load Balancer-Controller diese Subnetze direkt, um den Load Balancer zu erstellen, und die folgenden Tags sind nicht erforderlich.
  - Private Subnetze – Müssen im folgenden Format markiert sein. Auf diese Weise wissen Kubernetes und der Load Balancer-Controller, dass die Subnetze für interne Load AWS Balancer verwendet werden können. Wenn Sie nach dem 26. März 2020 eine Amazon AWS CloudFormation EKS-Vorlage verwenden`eksctl`, um Ihre VPC zu erstellen, werden die Subnetze bei der Erstellung entsprechend gekennzeichnet. Weitere Informationen zu den Amazon AWS CloudFormation EKS-VPC-Vorlagen finden Sie unterthe section called “Erstellen einer VPC”.
    - Schlüssel – `kubernetes.io/role/internal-elb`
    - Wert – 1

- Öffentliche Subnetze – Müssen im folgenden Format markiert sein. So erkennt Kubernetes, dass nur die Subnetze verwendbar sind, die für externe Load Balancer angegeben wurden. Auf diese Weise wählt Kubernetes nicht in jeder Availability Zone ein öffentliches Subnetz (lexikografisch basierend auf ihrer Subnetz-ID). Wenn Sie nach dem 26. März 2020 eine Amazon AWS CloudFormation EKS-Vorlage verwendeneksctl, um Ihre VPC zu erstellen, werden die Subnetze bei der Erstellung entsprechend gekennzeichnet. Weitere Informationen zu den Amazon AWS CloudFormation EKS-VPC-Vorlagen finden Sie unterthe section called “Erstellen einer VPC”.
  - Schlüssel – kubernetes.io/role/elb
  - Wert – 1

  Wenn die Subnetz-Rollen-Tags nicht explizit hinzugefügt werden, untersucht der Kubernetes-Service-Controller die Routing-Tabelle Ihrer Cluster-VPC-Subnetze. Dadurch wird festgestellt, ob das Subnetz privat oder öffentlich ist. Wir empfehlen, sich nicht auf dieses Verhalten zu verlassen. Fügen Sie stattdessen explizit die Rollen-Tags privat oder öffentlich hinzu. Der AWS Load Balancer Controller untersucht keine Routentabellen. Außerdem müssen die Tags privat und öffentlich für eine erfolgreiche automatische Erkennung vorhanden sein.
- Der Load AWS Balancer Controller erstellt ALBs und die erforderlichen unterstützenden AWS Ressourcen, wenn eine Kubernetes-Eingangsressource auf dem Cluster mit der Anmerkung erstellt wird. kubernetes.io/ingress.class: alb Die Ressource für eingehenden Datenverkehr konfiguriert den ALB so, dass HTTP- oder HTTPS-Datenverkehr an verschiedene Pods im Cluster weitergeleitet wird. Um sicherzustellen, dass Ihre Ingress-Objekte den Load AWS Balancer Controller verwenden, fügen Sie Ihrer Kubernetes-Ingress-Spezifikation die folgende Anmerkung hinzu. Weitere Informationen finden Sie unter Ingress-Spezifikation auf GitHub.

```
annotations:
    kubernetes.io/ingress.class: alb
```

> **Note**
>
> Wenn Sie Load Balancing für IPv6-Pods verwenden, fügen Sie die folgende Annotation zu Ihrer Ingress-Spezifikation hinzu. Load Balancing über IPv6 funktioniert nur auf IP-Ziele, nicht auf Instance-Ziele. Ohne diese Anmerkung verläuft das Load Balancing über IPv4.

```
alb.ingress.kubernetes.io/ip-address-type: dualstack
```

- Der AWS Load Balancer Controller unterstützt die folgenden Verkehrsmodi:
  - Instance – Registriert Knoten in Ihrem Cluster als Ziele für den ALB. Der Datenverkehr, der den ALB erreicht, wird für Ihren Service an `NodePort` und dann an Ihre Pods weitergeleitet. Dies ist der standardmäßige Datenverkehrsmodus. Sie können ihn auch explizit mit der `alb.ingress.kubernetes.io/target-type: instance`-Anmerkung angeben.

    > **Note**
    >
    > Ihr Kubernetes-Service muss den Typ `NodePort` oder `LoadBalancer` angeben, um diesen Datenverkehrsmodus verwenden zu können.

  - IP – Registriert Pods als Ziele für den ALB. Der Datenverkehr, der den ALB erreicht, wird für Ihren Service direkt an Pods weitergeleitet. Sie müssen die `alb.ingress.kubernetes.io/target-type: ip`-Anmerkung angeben, um diesen Datenverkehrsmodus verwenden zu können. Der IP-Zieltyp ist erforderlich, wenn Ziel-Pods in Fargate oder Amazon EKS Hybrid Nodes ausgeführt werden.
- Um ein vom Controller ALBs erstelltes Tag zu erstellen, fügen Sie dem Controller die folgende Anmerkung hinzu:`alb.ingress.kubernetes.io/tags`. Eine Liste aller verfügbaren Anmerkungen, die vom Load AWS Balancer Controller unterstützt werden, finden Sie unter Ingress-Anmerkungen auf. GitHub
- Ein Upgrade oder Downgrade der ALB-Controller-Version kann zu einer Unterbrechung der Abwärtskompatibilität für Features führen, die darauf angewiesen sind. Weitere Informationen zu den Breaking Changes, die in jeder Version eingeführt werden, finden Sie in den Versionshinweisen zum ALB-Controller auf GitHub.

Wiederverwendung mit Ingress-Gruppen ALBs

Sie können einen Application Load Balancer über mehrere Service-Ressourcen hinweg mithilfe von `IngressGroups` gemeinsam nutzen.

Um einen Ingress einer Gruppe hinzuzufügen, fügen Sie die folgende Anmerkung zu einer Kubernetes-Ingress-Ressourcenspezifikation hinzu.

```
alb.ingress.kubernetes.io/group.name: my-group
```

Der Gruppenname muss:

- Eine Länge von 63 oder weniger Zeichen haben.
- Aus Kleinbuchstaben, Zahlen, `-` und `.` bestehen
- Muss mit einer Zahl oder einem Buchstaben beginnen und enden.

Der Controller führt automatisch Ingress-Regeln für alle Ingresses in derselben Ingress-Gruppe zusammen. Es unterstützt sie mit einem einzigen ALB. Die meisten Anmerkungen, die auf einem Ingress definiert sind, gelten nur für die von diesem Ingress definierten Pfade. Standardmäßig gehören Ingress-Ressourcen keiner Ingress-Gruppe an.

> **Warning**
>
> Potentielles Sicherheitsrisiko
> Geben Sie nur dann eine Ingress-Gruppe für einen Ingress an, wenn alle Kubernetes-Benutzer mit RBAC-Berechtigung zum Erstellen oder Ändern von Ingress-Ressourcen innerhalb derselben Vertrauensgrenze liegen. Wenn Sie die Anmerkung mit einem Gruppennamen hinzufügen, können andere Kubernetes-Benutzer ihre Ingresses erstellen oder ändern, sodass sie derselben Ingress-Gruppe angehören. Dies kann zu unerwünschtem Verhalten führen, z. B. zum Überschreiben vorhandener Regeln durch Regeln mit höherer Priorität.

Sie können eine Reihenfolgennummer zu Ihrer Ingress-Ressource hinzufügen.

```
alb.ingress.kubernetes.io/group.order: '10'
```

Die Zahl kann 1-1000 sein. Die niedrigste Zahl für alle Ingresse in derselben Ingress-Gruppe wird zuerst ausgewertet. Alle Ingresses ohne diese Anmerkung werden mit dem Wert null bewertet. Doppelte Regeln mit einer höheren Nummer können Regeln mit einer niedrigeren Nummer überschreiben. Standardmäßig wird die Regelreihenfolge zwischen Ingressen innerhalb derselben Ingress-Gruppe lexikografisch basierend auf Namespace und Name bestimmt.

> **Important**
>
> Stellen Sie sicher, dass jeder Ingress in derselben Ingress-Gruppe eine eindeutige Prioritätsnummer hat. In Ingresses dürfen keine doppelten Bestellnummern vorhanden sein.

(Optional) Bereitstellen einer Beispielanwendung

- Mindestens ein öffentliches oder privates Subnetz in Ihrer Cluster-VPC.
- Lassen Sie den AWS Load Balancer Controller auf Ihrem Cluster bereitstellen. Weitere Informationen finden Sie unter the section called “ AWS Load Balancer Balancer-Controller”. Wir empfehlen Version `2.7.2` oder höher.

Sie können die Beispielanwendung auf einem Cluster ausführen, der EC2 Amazon-Knoten, Fargate Pods oder beides hat.

1. Wenn Sie keine Bereitstellung für Fargate durchführen, überspringen Sie diesen Schritt. Wenn Sie Fargate bereitstellen, erstellen Sie ein Fargate-Profil. Sie können das Profil erstellen, indem Sie den folgenden Befehl ausführen oder in AWS-Managementkonsole die gleichen Werte für `name` und `namespace` verwenden, die im Befehl enthalten sind. Ersetzen Sie die Beispielwerte durch Ihre eigenen Werte.

   ```
   eksctl create fargateprofile \
       --cluster my-cluster \
       --region region-code \
       --name alb-sample-app \
       --namespace game-2048
   ```

2. Stellen Sie das Spiel 2048 als Beispielanwendung bereit, um zu überprüfen, ob der Load AWS Balancer Controller als Ergebnis des AWS Eingangsobjekts eine ALB erstellt. Führen Sie die Schritte für den Typ des Subnetzes aus, in dem Sie bereitstellen.

   a. Wenn Sie in Pods in einem Cluster bereitstellen, den Sie mit der `IPv6`-Familie erstellt haben, fahren Sie mit dem nächsten Schritt fort.

      - Öffentlich::

      ```
      kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/aws-load-
      balancer-controller/v2.14.1/docs/examples/2048/2048_full.yaml
      ```

- Privat::

  A. Laden Sie das Manifest herunter.

  ```
  curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.14.1/docs/examples/2048/2048_full.yaml
  ```

  B. Bearbeiten Sie die Datei und suchen Sie die Zeile mit der Aufschrift `alb.ingress.kubernetes.io/scheme: internet-facing`.

  C. Ändern Sie *`internet-facing`* in `internal` und speichern Sie die Datei.

  D. Wenden Sie das Manifest auf Ihren Cluster an.

  ```
  kubectl apply -f 2048_full.yaml
  ```

b. Wenn Sie die Bereitstellung auf Pods in einem Cluster durchführen, den Sie mit der IPv6 Familie erstellt haben, gehen Sie wie folgt vor.

   i. Laden Sie das Manifest herunter.

   ```
   curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.14.1/docs/examples/2048/2048_full.yaml
   ```

   ii. Öffnen Sie die Datei in einem Editor und fügen Sie die folgende Zeile zu den Anmerkungen in der Ingress-Spezifikation hinzu.

   ```
   alb.ingress.kubernetes.io/ip-address-type: dualstack
   ```

   iii. Wenn Sie den Lastenausgleich in interne Pods anstelle von internetfähige-Pods anwenden, ändern Sie die Zeile, die `alb.ingress.kubernetes.io/scheme: internet-facing` zu `alb.ingress.kubernetes.io/scheme: internal` angibt

   iv. Speichern Sie die Datei.

   v. Wenden Sie das Manifest auf Ihren Cluster an.

   ```
   kubectl apply -f 2048_full.yaml
   ```

3. Überprüfen Sie nach ein paar Minuten mit dem folgenden Befehl, ob die Ressource für eingehenden Datenverkehr erstellt wurde.

```
kubectl get ingress/ingress-2048 -n game-2048
```

Eine Beispielausgabe sieht wie folgt aus.

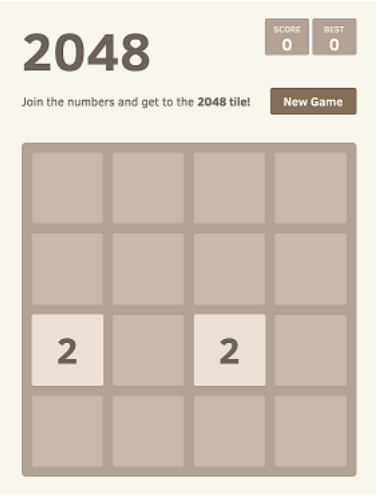
```
NAME           CLASS    HOSTS   ADDRESS
                     PORTS   AGE
ingress-2048   <none>   *       k8s-game2048-ingress2-xxxxxxxxxx-yyyyyyyyyy.region-
code.elb.amazonaws.com   80      2m32s
```

> **Note**
>
> Wenn Sie den Load Balancer in einem privaten Subnetz erstellt haben, wird dem Wert unter ADDRESS in der vorherigen Ausgabe `internal-` vorangestellt.

Wenn Ihr Ingress nach einigen Minuten nicht erfolgreich erstellt wurde, führen Sie den folgenden Befehl aus, um die Load AWS Balancer Controller-Protokolle anzuzeigen. Diese Protokolle können Fehlermeldungen enthalten, mit denen Sie Probleme mit Ihrer Bereitstellung diagnostizieren können.

```
kubectl logs -f -n kube-system -l app.kubernetes.io/instance=aws-load-balancer-
controller
```

1. Wenn Sie in einem öffentlichen Subnetz bereitgestellt haben, öffnen Sie einen Browser, und navigieren Sie zur ADDRESS-URL aus der vorherigen Befehlausgabe, um die Beispielanwendung anzuzeigen. Wenn nichts angezeigt wird, aktualisieren Sie Ihren Browser und versuchen Sie es erneut. Wenn Sie in einem privaten Subnetz bereitgestellt haben, müssen Sie die Seite von einem Gerät in Ihrer VPC aus anzeigen, z. B. von einem Bastion-Host. Weitere Informationen finden Sie unter Linux-Bastion-Hosts in AWS.



2. Wenn Sie mit dem Experimentieren mit Ihrer Beispielanwendung fertig sind, löschen Sie sie, indem Sie einen der folgenden Befehle ausführen.
   - Wenn Sie das Manifest angewendet haben, anstatt eine heruntergeladene Kopie anzuwenden, verwenden Sie den folgenden Befehl.

     ```
     kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.14.1/docs/examples/2048/2048_full.yaml
     ```

   - Wenn Sie das Manifest heruntergeladen und bearbeitet haben, verwenden Sie den folgenden Befehl.

     ```
     kubectl delete -f 2048_full.yaml
     ```

Einschränkung externer IP-Adressen, die Services zugewiesen werden können

Kubernetes-Services können von innerhalb eines Clusters aus erreicht werden durch:

- Eine Cluster-IP-Adresse, die automatisch von Kubernetes zugewiesen wird
- Jede IP-Adresse, die Sie für die `externalIPs`-Eigenschaft in einer Service-Spezifikation festlegen. Externe IP-Adressen werden nicht von Kubernetes verwaltet und liegen in der Verantwortung des Cluster-Administrators. Externe IP-Adressen, die mit `externalIPs` festgelegt sind, unterscheiden sich von der externen IP-Adresse, die einem Service vom Typ `LoadBalancer` von einem Cloud-Anbieter zugewiesen ist.

Für weitere Informationen über Kubernetes Service, siehe Service in der Kubernetes-Dokumentation. Sie können die IP-Adressen einschränken, die für `externalIPs` in einer Service-Spezifikation festgelegt werden können.

1. Stellen Sie `cert-manager` bereit, um Webhook-Zertifikate zu verwalten. Weitere Informationen dazu finden Sie in der cert-manager-Dokumentation.

   ```
   kubectl apply -f https://github.com/jetstack/cert-manager/releases/download/v1.5.4/cert-manager.yaml
   ```

2. Überprüfen Sie, ob die `cert-manager`-Pods ausgeführt werden.

```
kubectl get pods -n cert-manager
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME                                       READY   STATUS    RESTARTS   AGE
cert-manager-58c8844bb8-nlx7q              1/1     Running   0          15s
cert-manager-cainjector-745768f6ff-696h5   1/1     Running   0          15s
cert-manager-webhook-67cc76975b-4v4nk      1/1     Running   0          14s
```

3. Überprüfen Sie Ihre vorhandenen Services, um sicherzustellen, dass keiner von ihnen externe IP-Adressen zugewiesen sind, die nicht in dem CIDR-Block enthalten sind, auf den Sie Adressen beschränken möchten.

```
kubectl get services -A
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAMESPACE                   NAME                                      TYPE
 CLUSTER-IP       EXTERNAL-IP     PORT(S)         AGE
cert-manager                cert-manager                              ClusterIP
 10.100.102.137   <none>          9402/TCP        20m
cert-manager                cert-manager-webhook                      ClusterIP
 10.100.6.136     <none>          443/TCP         20m
default                     kubernetes                                ClusterIP
 10.100.0.1       <none>          443/TCP         2d1h
externalip-validation-system   externalip-validation-webhook-service   ClusterIP
 10.100.234.179   <none>          443/TCP         16s
kube-system                 kube-dns                                  ClusterIP
 10.100.0.10      <none>          53/UDP,53/TCP   2d1h
my-namespace                my-service                                ClusterIP
 10.100.128.10    192.168.1.1     80/TCP          149m
```

Wenn es sich bei einem der Werte um IP-Adressen handelt, die sich nicht innerhalb des Blocks befinden, auf den Sie den Zugriff einschränken möchten, müssen Sie die Adressen so ändern, dass sie sich innerhalb des Blocks befinden, und die Services erneut bereitstellen. Zum Beispiel, der `my-service`-Service in der vorherigen Ausgabe hat eine externe IP-Adresse zugewiesen, die sich nicht innerhalb des CIDR-Blockbeispiels in Schritt 5 befindet.

4. Laden Sie das externe IP-Webhook-Verzeichnis herunter. Sie können auch den Quellcode für Webhook auf GitHub ansehen.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/docs/externalip-webhook.yaml
```

5. CIDR-Blöcke angeben. Öffnen Sie die heruntergeladene Datei in Ihrem Editor und entfernen Sie `\#` beim Start der folgenden Zeilen.

   ```
   #args:
   #- --allowed-external-ip-cidrs=10.0.0.0/8
   ```

   Ersetzen Sie `10.0.0.0/8` mit Ihrem eigenen CIDR-Block. Sie können so viele Blöcke wie gewünscht festlegen. Wenn Sie mehrere Blöcke angeben, fügen Sie ein Komma zwischen Blöcken hinzu.

6. Wenn sich Ihr Cluster nicht in der Region `us-west-2` AWS befindet, ersetzen Sie `us-west-2`, `602401143452` und `amazonaws.com` in der Datei durch die folgenden Befehle. Ersetzen Sie vor dem Ausführen der Befehle *`region-code`* und *`111122223333`* durch den Wert für Ihre AWS-Region aus der Liste in Amazon-Container-Image-Registrierungen für Amazon-EKS-Add-Ons anzeigen.

   ```
   sed -i.bak -e 's|602401143452|111122223333|' externalip-webhook.yaml
   sed -i.bak -e 's|us-west-2|region-code|' externalip-webhook.yaml
   sed -i.bak -e 's|amazonaws.com||' externalip-webhook.yaml
   ```

7. Wenden Sie das Manifest auf Ihren Cluster an.

   ```
   kubectl apply -f externalip-webhook.yaml
   ```

   Ein Versuch, einen Service auf Ihrem Cluster mit einer IP-Adresse bereitzustellen, die für `externalIPs` festgelegt ist, die nicht in den Blöcken enthalten ist, die Sie im Schritt CIDR-Blöcke angeben angegeben haben, schlägt fehl.

Kopieren eines Container-Images von einem Repository in ein anderes

In diesem Thema wird beschrieben, wie Sie ein Container-Image aus einem Repository abrufen, auf das Ihre Knoten keinen Zugriff haben, und in ein Repository verschieben, auf das sie Zugriff haben. Das Image können Sie in Amazon ECR oder ein alternatives Repository verschieben, auf das die Knoten Zugriff haben.

- Die Docker-Engine ist auf Ihrem Computer installiert und konfiguriert. Weitere Informationen finden Sie unter Installieren der Docker-Engine in der Docker-Dokumentation.
- Version `2.12.3` oder höher oder Version `1.27.160` oder höher der auf Ihrem Gerät installierten und konfigurierten AWS Befehlszeilenschnittstelle (AWS CLI) oder AWS CloudShell. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Paketmanager wie `yum`apt-get, oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der AWS CLI. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Die AWS CLI-Version, in der installiert ist, AWS CloudShell kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zur Aktualisierung finden Sie im AWS CloudShell Benutzerhandbuch unter AWS CLI in Ihrem Home-Verzeichnis installieren.
- Ein Schnittstellen-VPC-Endpunkt für Amazon ECR, wenn die Nodes Container-Images aus einem privaten Amazon-ECR-Repository über das Amazon-Netzwerk abrufen bzw. in ein solches verschieben sollen. Weitere Informationen finden Sie unter Erstellen der VPC Endpunkte für Amazon ECR im Benutzerhandbuch von Amazon Elastic Container Registry.

Gehen Sie wie folgt vor, um ein Container-Image aus einem Repository abzurufen und es in ein eigenes Repository zu verschieben. In den folgenden Beispielen, die in diesem Thema bereitgestellt werden, wird das Image für das Amazon-VPC-CNI-Plugin für den Kubernetes-Metrik-Helper abgerufen. Ersetzen Sie beim Ausführen dieser Schritte die Beispielwerte unbedingt durch eigene Werte.

1. Wenn Sie noch nicht über ein Amazon-ECR-Repository oder ein anderes Repository verfügen, erstellen Sie eines, auf das die Knoten zugreifen können. Mit dem folgenden Befehl wird ein privates Amazon-ECR-Repository erstellt. Der Name eines privaten Amazon-ECR-Repositorys muss mit einem Buchstaben beginnen. Er darf nur Kleinbuchstaben, Zahlen, Bindestriche (-), Unterstriche (_) und Schrägstriche (/) enthalten. Weitere Informationen finden Sie unter Erstellen eines privaten Repositorys im Benutzerhandbuch von Amazon Elastic Container Registry.

   Sie können *`cni-metrics-helper`* mit einem beliebigen Namen ersetzen, den Sie wählen. Erstellen Sie – als bewährte Methode – ein separates Repository für jedes Image. Das wird empfohlen, weil Image-Tags in einem Repository eindeutig sein müssen. *`region-code`*Ersetzen Sie durch eine AWS Region, die von Amazon ECR unterstützt wird.

   ```
   aws ecr create-repository --region region-code --repository-name cni-metrics-helper
   ```

2. Bestimmen Sie die Registrierung, das Repository und das Tag (optional) des Images, das die Knoten abrufen müssen. Diese Informationen liegen im Format `registry/repository[:tag]` vor.

   Bei vielen der Amazon-EKS-Themen zur Installation von Images müssen Sie eine Manifestdatei anwenden oder das Image mithilfe eines Helm-Charts installieren. Bevor Sie eine Manifest-Datei anwenden oder ein Helm-Chart installieren, sollten Sie sich jedoch zunächst den Inhalt des Manifests oder der Datei `values.yaml` des Charts ansehen. So können Sie die Registrierung, das Repository und das Tag für den Abrufvorgang bestimmen.

   Die folgende Zeile beispielsweise finden Sie in der Manifest-Datei für das Amazon-VPC-CNI-Plugin für den Kubernetes-Metrik-Helper. Die Registrierung ist `602401143452.dkr.ecr.us-west-2.amazonaws.com`, eine private Amazon-ECR-Registrierung. Das Repository ist `cni-metrics-helper`.

   ```
   image: "602401143452.dkr.ecr.us-west-2.amazonaws.com/cni-metrics-helper:v1.12.6"
   ```

   Die folgenden Variationen sind bei einem Image-Speicherort möglich:

   - Nur `repository-name:tag`. In diesem Fall ist in der Regel `docker.io` die Registrierung, aber nicht spezifiziert, da Kubernetes sie standardmäßig einem Repository-Namen voranstellt, wenn keine Registrierung angegeben ist.
   - `repository-name/repository-namespace/repository:tag`. Ein Repository-Namespace ist optional, wird jedoch manchmal vom Repository-Besitzer zum Kategorisieren von Images angegeben. Beispielsweise verwenden alle EC2 Amazon-Bilder in der Amazon ECR Public Gallery den `aws-ec2` Namespace.

     Zeigen Sie vor dem Installieren eines Images mit Helm die Helm-Datei `values.yaml` an, um den Image-Speicherort zu bestimmen. Beispielsweise enthält die Datei values.yaml für das Amazon-VPC-CNI-Plugin für den Kubernetes-Metrik-Helper die folgenden Zeilen.

     ```
     image:
       region: us-west-2
       tag: v1.12.6
       account: "602401143452"
       domain: "amazonaws.com"
     ```

3. Rufen Sie das in der Manifestdatei angegebene Container-Image ab.

a. Beim Abruf aus einer öffentlichen Registry wie zum Beispiel Amazon ECR Public Gallery können Sie zum nächsten Unterschritt springen, da keine Authentifizierung erforderlich ist. In diesem Beispiel authentifizieren Sie sich bei einer privaten Amazon-ECR-Registrierung, die das Repository für das Helper-Image für CNI-Kennzahlen enthält. Amazon EKS verwaltet das Image in jeder Registry, die unter Amazon-Container-Image-Registries für Amazon-EKS-Add-Ons anzeigen aufgeführt ist. Die Authentifizierung ist bei jeder der Registrierungen möglich. Dazu ersetzen Sie *602401143452* und *region-code* durch die Informationen für eine andere Registrierung. Für jede AWS -Region, in der Amazon EKS unterstützt wird, existiert eine separate Registry.

```
aws ecr get-login-password --region region-code | docker login --username AWS --
password-stdin 602401143452.dkr.ecr.region-code.amazonaws.com
```

b. Rufen Sie das Image ab. In diesem Beispiel erfolgt der Abruf aus der Registrierung, bei der Sie sich im letzten Unterschritt authentifiziert haben. Ersetzen Sie *602401143452* und *region-code* durch die Informationen, die Sie im vorherigen Unterschritt angegeben haben.

```
docker pull 602401143452.dkr.ecr.region-code.amazonaws.com/cni-metrics-
helper:v1.12.6
```

4. Markieren Sie das abgerufene Image mit Ihrer Registrierung, Ihrem Repository und Ihrem Tag. Im folgenden Beispiel wird davon ausgegangen, dass Sie das Image über die Manifestdatei abgerufen haben und es in das private Amazon-ECR-Repository verschieben, das Sie im ersten Schritt erstellt haben. Ersetzen Sie *111122223333* durch Ihre Konto-ID. *region-code*Ersetzen Sie es durch die AWS Region, in der Sie Ihr privates Amazon ECR-Repository erstellt haben.

```
docker tag cni-metrics-helper:v1.12.6 111122223333.dkr.ecr.region-code.amazonaws.com/
cni-metrics-helper:v1.12.6
```

5. Authentifizieren Sie sich bei Ihrer Registrierung. In diesem Beispiel authentifizieren Sie sich bei der privaten Amazon-ECR-Registrierung, die Sie im ersten Schritt erstellt haben. Weitere Informationen finden Sie unter Registrierungsauthentifizierung im Benutzerhandbuch von Amazon Elastic Container Registry.

```
aws ecr get-login-password --region region-code | docker login --username AWS --
password-stdin 111122223333.dkr.ecr.region-code.amazonaws.com
```

6. Verschieben Sie das Image in Ihr Repository. In diesem Beispiel verschieben Sie das Image in das private Amazon-ECR-Repository, das Sie im ersten Schritt erstellt haben. Weitere Informationen

finden Sie unter Verschieben eines Docker-Images im Benutzerhandbuch von Amazon Elastic Container Registry.

```
docker push 111122223333.dkr.ecr.region-code.amazonaws.com/cni-metrics-helper:v1.12.6
```

7. Aktualisieren Sie die Manifestdatei, mit deren Hilfe Sie das Image in einem vorherigen Schritt bestimmt haben, mit den Werten `registry/repository:tag` für das verschobene Image. Bei der Installation mit einem Helm-Chart gibt es oft eine Option zur Angabe von `registry/repository:tag`. Geben Sie bei der Installation des Charts die Werte `registry/repository:tag` für das Image an, das Sie in das Repository verschoben haben.

Amazon-Container-Image-Registries für Amazon-EKS-Add-Ons anzeigen

Wenn Sie Add-Ons von AWS Amazon-EKS in Ihrem Cluster bereitstellen, rufen Ihre Knoten die erforderlichen Container-Images aus der Registrierung ab, die im Installationsmechanismus für das Add-On angegeben ist, z. B. ein Installationsmanifest oder eine `values.yaml`-Helm-Datei. Die Images werden aus einem privaten Amazon-ECR-Repository abgerufen. Amazon EKS repliziert die Bilder in ein Repository in jeder von Amazon EKS unterstützten AWS Region. Die Knoten können das Container-Image über das Internet aus einer der folgenden Registrierungen abrufen. Alternativ können Ihre Knoten das Image über das Netzwerk von Amazon abrufen, wenn Sie in Ihrer VPC einen VPC-Schnittstellen-Endpunkt für Amazon ECR (AWS PrivateLink) erstellt haben. Die Registries erfordern eine Authentifizierung mit einem IAM-Konto. AWS Ihre Knoten authentifizieren sich mithilfe der Amazon EKS-Knoten-IAM-Rolle, der die Berechtigungen in der Amazon EC2 ContainerRegistryReadOnly Managed IAM-Richtlinie zugeordnet sind.

AWS Region	Registrierung
af-south-1	877085696533.dkr.ecr.af-south-1.amazonaws.com
ap-east-1	800184023465.dkr.ecr.ap-east-1.amazonaws.com
ap-east-2	533267051163.dkr. ecr.ap-east-1.amazonaws.com

AWS Region	Registrierung
ap-southeast-3	296578399912.dkr.ecr.ap-southeast-3.amazonaws.com
ap-south-2	900889452093.dkr.ecr.ap-south-2.amazonaws.com
ap-southeast-4	491585149902.dkr.ecr.ap-southeast-4.amazonaws.com
ap-southeast-6	333609536671.dkr. ecr.ap-southeast-6.amazonaws.com
ap-south-1	602401143452.dkr.ecr.ap-south-1.amazonaws.com
ap-northeast-3	602401143452.dkr.ecr.ap-northeast-3.amazonaws.com
ap-northeast-2	602401143452.dkr.ecr.ap-northeast-2.amazonaws.com
ap-southeast-1	602401143452.dkr.ecr.ap-southeast-1.amazonaws.com
ap-southeast-2	602401143452.dkr.ecr.ap-southeast-2.amazonaws.com
ap-southeast-7	121268973566.dkr. ecr.ap-southeast-7.amazonaws.com
ap-northeast-1	602401143452.dkr.ecr.ap-northeast-1.amazonaws.com
cn-north-1	918309763551.dkr.ecr.cn-north-1.amazonaws.com.cn
cn-northwest-1	961992271922.dkr.ecr.cn-northwest-1.amazonaws.com.cn

AWS Region	Registrierung
eu-central-1	602401143452.dkr.ecr.eu-central-1.amazonaws.com
eu-west-1	602401143452.dkr.ecr.eu-west-1.amazonaws.com
eu-west-2	602401143452.dkr.ecr.eu-west-2.amazonaws.com
eu-south-1	590381155156.dkr.ecr.eu-south-1.amazonaws.com
eu-west-3	602401143452.dkr.ecr.eu-west-3.amazonaws.com
eu-south-2	455263428931.dkr.ecr.eu-south-2.amazonaws.com
eu-north-1	602401143452.dkr.ecr.eu-north-1.amazonaws.com
eu-central-2	900612956339.dkr.ecr.eu-central-2.amazonaws.com
il-central-1	066635153087.dkr.ecr.il-central-1.amazonaws.com
mx-central-1	730335286997.dkr.ecr.mx-central-1.amazonaws.com
me-south-1	558608220178.dkr.ecr.me-south-1.amazonaws.com
me-central-1	759879836304.dkr.ecr.me-central-1.amazonaws.com
us-east-1	602401143452.dkr.ecr.us-east-1.amazonaws.com

AWS Region	Registrierung
us-east-2	602401143452.dkr.ecr.us-east-2.amazonaws.com
us-west-1	602401143452.dkr.ecr.us-west-1.amazonaws.com
us-west-2	602401143452.dkr.ecr.us-west-2.amazonaws.com
ca-central-1	602401143452.dkr.ecr.ca-central-1.amazonaws.com
ca-west-1	761377655185.dkr.ecr.ca-west-1.amazonaws.com
sa-east-1	602401143452.dkr.ecr.sa-east-1.amazonaws.com
us-gov-east-1	151742754352.dkr.ecr. us-gov-east-1.amazonaws.com
us-gov-west-1	013241004608.dkr.ecr. us-gov-west-1.amazonaws.com

Amazon-EKS-Add-ons

Ein Add-on ist eine Software, die unterstützende Betriebsfunktionen für Kubernetes-Anwendungen bereitstellt, aber nicht anwendungsspezifisch ist. Dazu gehören Software wie Observability-Agenten oder Kubernetes-Treiber, die es dem Cluster ermöglichen, mit den zugrunde liegenden AWS Ressourcen für Netzwerke, Berechnungen und Speicher zu interagieren. Zusatzsoftware wird in der Regel von der Kubernetes-Community, Cloud-Anbietern oder Drittanbietern entwickelt und verwaltet. AWS Amazon EKS installiert automatisch selbstverwaltete Add-Ons wie das Amazon-VPC-CNI-Plugin für Kubernetes, `kube-proxy` und CoreDNS für jeden Cluster. Beachten Sie, dass das VPC-CNI-Add-On nicht mit Amazon EKS Hybrid Nodes kompatibel ist und nicht in Hybridknoten bereitgestellt wird. Sie können die Standardkonfiguration der Add-ons ändern und sie bei Bedarf aktualisieren.

Amazon-EKS-Add-ons bieten die Installation und Verwaltung eines ausgewählten Satzes von Add-ons für Amazon-EKS-Cluster. Alle Amazon EKS-Add-Ons enthalten die neuesten Sicherheitspatches und Bugfixes und sind für die Verwendung AWS mit Amazon EKS validiert. Mit Amazon-EKS-Add-ons können Sie konsequent sicherstellen, dass Ihre Amazon-EKS-Cluster sicher und stabil sind und den Arbeitsaufwand reduzieren, den Sie für die Installation, Konfiguration und Aktualisierung von Add-ons benötigen. Wenn ein selbstverwaltetes Add-on wie `kube-proxy` bereits auf Ihrem Cluster ausgeführt wird und als Amazon-EKS-Add-on verfügbar ist, können Sie das Amazon-EKS-Add-on `kube-proxy` installieren, um die Vorteile der Funktionen von Amazon-EKS-Add-ons zu nutzen.

Sie können bestimmte von Amazon EKS verwaltete Konfigurationsfelder für Amazon-EKS-Add-ons über die Amazon EKS API aktualisieren. Sie können auch Konfigurationsfelder, die nicht von Amazon EKS verwaltet werden, direkt im Kubernetes-Cluster ändern, sobald das Add-on gestartet wird. Dazu gehört das Definieren spezifischer Konfigurationsfelder für ein Add-on, falls zutreffend. Diese Änderungen werden von Amazon EKS nicht überschrieben, sobald sie vorgenommen wurden. Dies ist möglich mit der serverseitigen Anwendungsfunktion von Kubernetes. Weitere Informationen finden Sie unter the section called “Felder, die angepasst werden können”.

Sie können Amazon-EKS-Add-Ons mit allen Amazon-EKS-Knotentypen verwenden. Weitere Informationen finden Sie unter Verwaltung von Rechenleistung.

Sie können Amazon EKS-Add-Ons mithilfe der Amazon EKS-API, AWS CLI und hinzufügen AWS-Managementkonsole, aktualisieren oder löschen`eksctl`. Sie können Amazon-EKS-Add-ons auch mit AWS CloudFormation erstellen.

Überlegungen

Beachten Sie Folgendes, wenn Sie Amazon-EKS-Add-Ons verwenden:

- Um Add-ons für den Cluster zu konfigurieren, muss der IAM–Prinzipal über IAM-Berechtigungen verfügen, um mit Add-ons zu arbeiten. Weitere Informationen finden Sie in den Aktionen mit `Addon` in ihrem Namen in Von Amazon Elastic Kubernetes Service definierte Aktionen.
- Amazon-EKS-Add-ons werden auf den Knoten ausgeführt, die Sie für Ihren Cluster bereitstellen oder konfigurieren. Zu den Knotentypen gehören EC2 Amazon-Instances, Fargate und Hybridknoten.
- Sie können Felder ändern, die nicht von Amazon EKS verwaltet werden, um die Installation eines Amazon-EKS-Add-Ons anzupassen. Weitere Informationen finden Sie unter the section called “Felder, die angepasst werden können”.

- Wenn Sie einen Cluster mit den Add-Ons Amazon EKS AWS-Managementkonsole`kube-proxy`, Amazon VPC CNI für Kubernetes und CoreDNS erstellen, werden Amazon EKS Add-Ons automatisch zu Ihrem Cluster hinzugefügt. Wenn Sie `config` verwenden, um Ihren Cluster mit einer `eksctl`-Datei zu erstellen, kann `eksctl`den Cluster auch mit Amazon-EKS-Add-ons erstellen. Wenn Sie einen Cluster mit `eksctl` ohne eine `config`-Datei oder mit einem anderen Tool erstellen, werden die selbstverwalteten `kube-proxy`, Amazon-VPC-CNI-Plugin für Kubernetes und CoreDNS-Add-Ons anstelle der Amazon-EKS-Add-Ons installiert. Sie können sie entweder selbst verwalten oder die Amazon-EKS-Add-ons nach der Clustererstellung manuell hinzufügen. Unabhängig von der Methode, die Sie zum Erstellen Ihres Clusters verwenden, wird das VPC-CNI-Add-On nicht auf Hybridknoten installiert.
- Die `eks:addon-cluster-admin` `ClusterRoleBinding` verknüpft das `cluster-admin` `ClusterRole` mit der `eks:addon-manager`-Kubernetes-Identität. Die Rolle verfügt über die erforderlichen Berechtigungen für die `eks:addon-manager`-Identität, um Kubernetes-Namespaces zu erstellen und Add-Ons in Namespaces zu installieren. Wenn `eks:addon-cluster-admin` `ClusterRoleBinding` entfernt wird, funktioniert der Amazon-EKS-Cluster weiterhin, Amazon EKS kann jedoch keine Add-Ons mehr verwalten. Alle Cluster ab den folgenden Plattformversionen verwenden das neue `ClusterRoleBinding`.
- Eine Teilmenge der EKS-Add-Ons von AWS wurde auf ihre Kompatibilität mit Amazon EKS Hybrid Nodes überprüft. Weitere Informationen finden Sie in der Kompatibilitätstabelle unter the section called “ AWS Add-Ons”.

Benutzerdefinierter Namespace für Add-Ons

Für Community- und AWS Add-Ons können Sie bei der Erstellung des Add-ons optional einen benutzerdefinierten Namespace angeben. Wenn Sie ein Add-On in einem bestimmten Namespace installieren, müssen Sie das Add-On entfernen und neu erstellen, um seinen Namespace zu ändern.

Wenn Sie keinen Namespace angeben, wird der vordefinierte Namespace für das Add-on verwendet.

Verwenden Sie benutzerdefinierte Namespaces für eine bessere Organisation und Isolierung von Add-on-Objekten innerhalb Ihres EKS-Clusters. Diese Flexibilität ermöglicht es Ihnen, Add-Ons an Ihre betrieblichen Anforderungen und Ihre vorhandene Namespace-Strategie anzupassen.

Sie können beim Erstellen eines Add-Ons einen benutzerdefinierten Namespace festlegen. Weitere Informationen finden Sie unter the section called “Add-On erstellen”.

Vordefinierten Namespace für Add-Ons abrufen

Der vordefinierte Namespace für ein Add-On ist der Namespace, in dem es installiert wird, wenn Sie keinen angeben.

Um den vordefinierten Namespace für ein Add-On abzurufen, verwenden Sie den folgenden Befehl:

```
aws eks describe-addon-versions --addon-name <addon-name> --query
 "addons[].defaultNamespace"
```

Beispielausgabe:

```
[
    "kube-system"
]
```

Überlegungen zu Amazon EKS Auto Mode

Amazon EKS Auto Mode umfasst Funktionen, die wesentliche Cluster-Funktionalitäten bereitstellen, darunter:

- Pod-Netzwerke
- Service-Netzwerke
- Cluster DNS
- Auto Scaling
- Blockspeicher
- Load Balancer Controller
- Pod Identity Agent
- Knotenüberwachungsagent

Mit der automatischen Berechnungsfunktion werden viele häufig verwendete EKS-Add-Ons überflüssig, darunter:

- Amazon VPC CNI
- kube-proxy

- CoreDNS
- Amazon-EBS-CSI-Treiber
- EKS Pod Identity Agent

Wenn Ihr Cluster den automatischen Modus jedoch mit anderen Rechenoptionen wie selbstverwalteten EC2 Instances, Managed Node Groups oder AWS Fargate kombiniert, bleiben diese Add-Ons weiterhin erforderlich. AWS hat EKS-Add-Ons um Anti-Affinitätsregeln erweitert, die automatisch sicherstellen, dass Add-On-Pods nur für unterstützte Compute-Typen geplant werden. Darüber hinaus können Benutzer jetzt die `DescribeAddonVersions`-API der EKS-Add-Ons nutzen, um die unterstützten ComputeTypes für jedes Add-On und seine spezifischen Versionen zu überprüfen. Darüber hinaus laufen die oben aufgeführten Controller im EKS-Automatikmodus auf der AWS eigenen Infrastruktur. Daher werden sie möglicherweise nicht in Ihren Konten angezeigt, es sei denn, Sie verwenden EKS Auto Mode mit anderen Rechnertypen. In diesem Fall werden die Controller angezeigt, die Sie in Ihrem Cluster installiert haben.

Wenn Sie EKS Auto Mode in einem vorhandenen Cluster aktivieren möchten, müssen Sie möglicherweise die Version bestimmter Add-Ons aktualisieren. Weitere Informationen finden Sie unter the section called “Erforderliche Add-On-Versionen” EKS Auto Mode.

Support

AWS veröffentlicht mehrere Arten von Add-Ons mit unterschiedlichen Unterstützungsstufen.

- AWS -Add-Ons: Diese Add-Ons werden von AWS entwickelt und vollständig unterstützt.
  - Verwenden Sie ein AWS Add-on, um mit anderen AWS Diensten wie Amazon EFS zu arbeiten.
  - Weitere Informationen finden Sie unter the section called “ AWS Add-Ons”.
- AWS Marketplace-Add-ons: Diese Add-Ons werden von einem unabhängigen AWS Partner gescannt AWS und von diesem unterstützt.
  - Nutzen Sie ein Marktplatz-Add-On, um Ihrem Cluster wertvolle und anspruchsvolle Features hinzuzufügen, wie beispielsweise die Überwachung mit Splunk.
  - Weitere Informationen finden Sie unter the section called “Marketplace-Add-Ons”.
- Community-Add-Ons: Diese Add-Ons werden von der Open-Source-Community gescannt, AWS aber von ihr unterstützt.
  - Verwenden Sie ein Community-Add-On, um die Installation gängiger Open-Source-Software wie Kubernetes Metrics Server zu vereinfachen.

- Community-Add-Ons werden aus dem Quellcode von gepackt AWS. AWS validiert nur Community-Add-Ons auf Versionskompatibilität.
- Weitere Informationen finden Sie unter the section called “Community-Erweiterungen”.

Die folgende Tabelle enthält Einzelheiten zum Support-Umfang für jeden Add-On-Typ:

Kategorie	Feature	AWS Add-Ons	AWS Marketplace-Add-ons	Community-Erweiterungen
Entwicklung	Gebaut von AWS	Ja	Nein	Ja
Entwicklung	Validiert von AWS	Ja	Nein	Ja*
Entwicklung	Vom AWS Partner validiert	Nein	Ja	Nein
Wartung	Gescannt von AWS	Ja	Ja	Ja
Wartung	Gepatcht von AWS	Ja	Nein	Ja
Wartung	Vom Partner gepatcht AWS	Nein	Ja	Nein
Distribution	Veröffentlicht von AWS	Ja	Nein	Ja
Distribution	Herausgegeben von AWS Partner	Nein	Ja	Nein
Support	Grundlegende Installationsunterstützung von AWS	Ja	Ja	Ja

Kategorie	Feature	AWS Add-Ons	AWS Marketplace-Add-ons	Community-Erweiterungen
Support	Umfassender AWS Support	Ja	Nein	Nein
Support	AWS Umfassender Partner-Support	Nein	Ja	Nein

*: Die Validierung für Community-Add-Ons umfasst nur die Kubernetes-Versionskompatibilität. Wenn Sie beispielsweise ein Community-Add-on auf einem Cluster installieren, prüfen Sie, AWS ob es mit der Kubernetes-Version Ihres Clusters kompatibel ist.

AWS Marketplace-Add-Ons können zusätzliche Softwareabhängigkeiten von externen Quellen außerhalb von herunterladen AWS. Diese externen Abhängigkeiten werden von nicht gescannt oder validiert AWS. Berücksichtigen Sie Ihre Sicherheitsanforderungen bei der Bereitstellung von AWS Marketplace-Add-Ons, die externe Abhängigkeiten abrufen.

AWS Add-Ons

Die folgenden Amazon-EKS-Add-Ons können in Ihrem Cluster erstellt werden. Sie können die aktuelle Liste der verfügbaren Add-Ons mithilfe `eksctl` der AWS-Managementkonsole oder der AWS CLI anzeigen. Informationen zu allen verfügbaren Add-Ons oder zur Installation eines Add-Ons finden Sie unter the section called “Add-On erstellen”. Wenn ein Add-On IAM-Berechtigungen erfordert, müssen Sie über einen IAM OpenID Connect (OIDC)-Anbieter für Ihren Cluster verfügen. Um festzustellen, ob Sie über einen solchen verfügen oder um einen zu erstellen, lesen Sie the section called “IAM-OIDC-Anbieter”. Sie können ein Add-On erstellen oder löschen, nachdem Sie es installiert haben. Für weitere Informationen siehe the section called “Add-On aktualisieren” oder the section called “Add-On entfernen”. Weitere Informationen zu den spezifischen Überlegungen zur Ausführung von EKS-Add-Ons mit Amazon EKS Hybrid Nodes finden Sie unter the section called “Add-Ons konfigurieren”.

Sie können jedes der folgenden Amazon-EKS-Add-Ons verwenden.

Description	Weitere Informationen	Kompatible Rechentypen
Bereitstellung eines nativen VPC-Netzwerks für Ihren Cluster	the section called “Amazon-VPC-CNI-Plugin für Kubernetes”	EC2
Ein flexibler, erweiterbarer DNS-Server, der als DNS für den Kubernetes-Cluster dienen kann	the section called “CoreDNS”	EC2, Fargate, EKS-Automatikmodus, EKS-Hybridknoten
Pflegen Sie die Netzwerkregeln auf jedem EC2 Amazon-Knoten	the section called “`kube-proxy`”	EC2, EKS-Hybridknoten
Bereitstellung von Amazon-EBS-Speicher für Ihren Cluster	the section called “Amazon-EBS-CSI-Treiber”	EC2
Bereitstellung von Amazon-EFS-Speicher für Ihren Cluster	the section called “Amazon EFS-CSI-Treiber”	EC2, EKS-Automatikmodus
Stellen Sie Amazon FSx for Lustre-Speicher für Ihren Cluster bereit	the section called “Amazon FSx CSI-Treiber”	EC2, EKS-Automatikmodus
Bereitstellung von Amazon-S3-Speicher für Ihren Cluster	the section called “CSI-Treiber für Mountpoint für Amazon S3”	EC2, EKS-Automatikmodus
Erkennen zusätzlicher Knotenintegritätsprobleme	the section called “Knotenüberwachungsagent”	EC2, EKS-Hybridknoten
Aktivierung der Verwendung der Snapshot-Funktionalität in kompatiblen CSI-Treibern, wie beispielsweise dem Amazon-EBS-CSI-Treiber	the section called “CSI-Snapshot-Controller”	EC2, Fargate, EKS-Automatikmodus, EKS-Hybridknoten

Description	Weitere Informationen	Kompatible Rechentypen
SageMaker HyperPod Task Governance optimiert die Zuweisung und Nutzung von Rechenressourcen durch Teams in Amazon EKS-Clust ern und behebt damit Ineffizie nzen bei der Priorisierung von Aufgaben und der gemeinsam en Nutzung von Ressourcen.	the section called “ SageMaker HyperPod Amazon-Aufgabenverwaltung”	EC2, Automatischer EKS-Modus,
Amazon SageMaker HyperPod Observability AddOn bietet umfassend e Überwachungs- und Beobachtbarkeitsfunktionen für HyperPod Cluster.	the section called “Amazon SageMaker HyperPod Observability-Zusatzmodul”	EC2, Automatischer EKS-Modus,
Amazon SageMaker HyperPod Training Operator ermöglicht effiziente verteilte Schulungen auf Amazon EKS-Clustern mit erweitert en Planungs- und Ressource nverwaltungsfunktionen.	the section called “ SageMaker HyperPod Amazon-Schulungsleiter”	EC2, EKS-Automatikmodus
Ein Kubernetes-Agent, der Netzwerkflussdaten sammelt und an Amazon CloudWatch meldet und so eine umfassend e Überwachung von TCP-Verbindungen zwischen Clusterknoten ermöglicht.	the section called “AWS Network Flow Monitor-Agent”	EC2, EKS-Automatikmodus

Description	Weitere Informationen	Kompatible Rechentypen
Sicherer, produktionsreifer, AWS unterstützter Vertrieb des Projekts OpenTelemetry	the section called “AWS Distribution für OpenTelemetry”	EC2, Fargate, EKS-Automatikmodus, EKS-Hybridknoten
Sicherheitsüberwachungsservice, der grundlegende Datenquellen analysiert und verarbeitet, darunter AWS CloudTrail Verwaltungsereignisse und Amazon VPC-Flow-Logs. Amazon verarbeitet GuardDuty auch Funktionen wie Kubernetes-Audit-Logs und Laufzeitüberwachung.	the section called “ GuardDuty Amazon-Vertreter”	EC2, EKS-Automatikmodus
Überwachungs- und Beobachtbarkeitsservice bereitgestellt von AWS. Dieses Add-on installiert den CloudWatch Agenten und aktiviert sowohl CloudWatch Application Signals als auch CloudWatch Container Insights mit verbesserter Observability für Amazon EKS	the section called “Amazon CloudWatch Observability-Agent”	EC2, EKS-Automatikmodus, EKS-Hybridknoten
Möglichkeit, Anmeldeinformationen für Ihre Anwendungen zu verwalten, ähnlich wie EC2 Instanzprofile Anmeldeinformationen für EC2 Instanzen bereitstellen	the section called “EKS Pod Identity Agent”	EC2, EKS-Hybridknoten

Description	Weitere Informationen	Kompatible Rechentypen
Aktivieren Sie den Cert-Manager, um X.509-Zertifikate von AWS einer privaten CA auszustellen. Erfordert cert-manager.	the section called “AWS Privater CA-Connector für Kubernetes”	EC2, Fargate, EKS-Automatikmodus, EKS-Hybridknoten
Generierung von Prometheus-Metriken zur Leistung von SR-IOV-Netzwerkgeräten	the section called “SR-IOV-Netzwerkmetrik-Exportprogramm”	EC2
Rufen Sie Geheimnisse aus AWS Secrets Manager und Parameter aus dem AWS Systems Manager Parameter Store ab und mounten Sie sie als Dateien in Kubernetes-Pods.	the section called “AWS Secrets Store CSI-Treiberanbieter”	EC2, EKS-Automatikmodus, EKS-Hybridknoten
Mit Spaces können Sie Anwendungen im Code-Editor erstellen JupyterLab und verwalten, um interaktive ML-Workloads auszuführen.	the section called “Amazon SageMaker Spaces”	Hyperpod

Amazon-VPC-CNI-Plugin für Kubernetes

Das Amazon-VPC-CNI-Plugin für das Kubernetes Amazon-EKS-Add-On ist ein Kubernetes Container Network Interface (CNI)-Plugin, das native VPC-Netzwerke für Ihren Cluster bereitstellt. Der selbstverwaltete oder verwaltete Typ dieses Add-ons ist standardmäßig auf jedem EC2 Amazon-Knoten installiert. Weitere Informationen finden Sie unter Kubernetes Container Network Interface (CNI)-Plugin.

> **Note**
>
> Sie müssen dieses Add-On nicht in Clustern von Amazon EKS Auto Mode installieren. Weitere Informationen finden Sie unter the section called “Überlegungen zu Amazon EKS Auto Mode”.

Der Name des Amazon-EKS-Add-Ons lautet `vpc-cni`.

Erforderliche IAM-Berechtigungen

Dieses Add-on verwendet die IAM-Rollen für die Servicekontofunktion von Amazon EKS. Weitere Informationen finden Sie unter the section called “Anmeldeinformationen mit IRSA”.

Wenn Ihr Cluster die `IPv4`-Familie verwendet, sind die Berechtigungen in der AmazonEKS_CNI_Policy erforderlich. Wenn Ihr Cluster die `IPv6` Familie verwendet, müssen Sie eine IAM-Richtlinie mit den Berechtigungen im IPv6 Modus erstellen. Sie können eine IAM-Rolle erstellen, ihr eine der Richtlinien anfügen und das vom Add-On verwendete Kubernetes-Servicekonto mit dem folgenden Befehl kommentieren.

Ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters und *`AmazonEKSVPCCNIRole`* durch den Namen Ihrer Rolle. Wenn der Cluster die `IPv6`-Familie verwendet, ersetzen Sie *`AmazonEKS_CNI_Policy`* durch den Namen der Richtlinie, die Sie erstellt haben. Dieser Befehl erfordert, dass Sie eksctl auf Ihrem Gerät installiert haben. Wenn Sie ein anderes Tool verwenden müssen, um die Rolle zu erstellen, ihr die Richtlinie zuzuordnen und das Kubernetes-Servicekonto mit Annotationen zu versehen, siehe the section called “IAM-Rolle zuweisen”.

```
eksctl create iamserviceaccount --name aws-node --namespace kube-system --cluster my-
cluster --role-name AmazonEKSVPCCNIRole \
    --role-only --attach-policy-arn arn:aws: iam::aws:policy/AmazonEKS_CNI_Policy --
approve
```

Aktualisieren der Informationen

Sie können jeweils nur eine Nebenversion aktualisieren. Wenn Ihre aktuelle Version beispielsweise `1.28.`*`x`*`-eksbuild.`*`y`* ist und Sie auf `1.30.`*`x`*`-eksbuild.`*`y`* aktualisieren möchten, müssen Sie zuerst ihre aktuelle Version auf `1.29.`*`x`*`-eksbuild.`*`y`* und dann auf `1.30.`*`x`*`-eksbuild.`*`y`* aktualisieren. Weitere Informationen zum Aktualisieren des Add-ons finden Sie unter the section called “Aktualisierung (EKS-Add-On)”.

CoreDNS

CoreDNS-Amazon-EKS-Add-On ist ein flexibler, erweiterbarer DNS-Server, der als Kubernetes-Cluster-DNS dienen kann. Der selbstverwaltete oder verwaltete Typ dieses Add-Ons wurde standardmäßig installiert, als Sie Ihren Cluster erstellt haben. Wenn Sie einen Amazon-EKS-Cluster mit mindestens einem Knoten starten, werden standardmäßig zwei Replikate des CoreDNS-Image bereitgestellt, unabhängig von der Anzahl der in Ihrem Cluster bereitgestellten Knoten. Die CoreDNS-Pods bieten eine Namensauflösung für alle Pods im Cluster. Sie können CoreDNS-Pods in Fargate-Knoten bereitstellen, wenn Ihr Cluster ein Fargate-Profil mit einem Namespace enthält, der mit dem Namespace für die CoreDNS-Bereitstellung übereinstimmt. Weitere Informationen finden Sie unter the section called “Profile definieren”.

> **Note**
>
> Sie müssen dieses Add-On nicht in Clustern von Amazon EKS Auto Mode installieren. Weitere Informationen finden Sie unter the section called “Überlegungen zu Amazon EKS Auto Mode”.

Der Name des Amazon-EKS-Add-Ons lautet `coredns`.

Erforderliche IAM-Berechtigungen

Dieses Add-On erfordert keine Berechtigungen.

Zusätzliche Informationen

Weitere Informationen zu CoreDNS finden Sie unter Verwenden von CoreDNS für die Serviceerkennung und Anpassen des DNS-Services in der Kubernetes-Dokumentation.

`Kube-proxy`

Das `Kube-proxy` Amazon EKS-Add-on verwaltet die Netzwerkregeln auf jedem EC2 Amazon-Knoten. Es ermöglicht die Netzwerkkommunikation zu Ihren Pods. Der selbstverwaltete oder verwaltete Typ dieses Add-ons ist standardmäßig auf jedem EC2 Amazon-Knoten in Ihrem Cluster installiert.

> **Note**
>
> Sie müssen dieses Add-On nicht in Clustern von Amazon EKS Auto Mode installieren. Weitere Informationen finden Sie unter the section called “Überlegungen zu Amazon EKS Auto Mode”.

Der Name des Amazon-EKS-Add-Ons lautet `kube-proxy`.

Erforderliche IAM-Berechtigungen

Dieses Add-On erfordert keine Berechtigungen.

Aktualisieren der Informationen

Bevor Sie Ihre aktuelle Version aktualisieren, sollten Sie die folgenden Anforderungen berücksichtigen:

- `Kube-proxy` in einem Amazon-EKS-Cluster verfügt über die gleiche Kompatibilitäts- und Skew-Richtlinie wie Kubernetes.

Zusätzliche Informationen

Weitere Informationen zu `kube-proxy` finden Sie unter kube-proxy in der Kubernetes-Dokumentation.

Amazon-EBS-CSI-Treiber

Das Amazon-EKS-Add-On für Amazon-EBS-CSI-Treiber ist ein Kubernetes Container Storage Interface (CSI)-Plugin, das Amazon-EBS-Speicher für Ihren Cluster bereitstellt.

> **Note**
>
> Sie müssen dieses Add-On nicht in Clustern von Amazon EKS Auto Mode installieren. Der Automatikmodus umfasst eine Block-Speicherfunktion. Weitere Informationen finden Sie unter the section called “Workload mit Statusverwaltung bereitstellen”.

Der Name des Amazon-EKS-Add-Ons lautet `aws-ebs-csi-driver`.

Erforderliche IAM-Berechtigungen

Dieses Add-On nutzt die IAM-Rollen für Servicekonten-Funktionalität von Amazon EKS. Weitere Informationen finden Sie unter the section called “Anmeldeinformationen mit IRSA”. Die Berechtigungen in der von Amazon EBSCSIDriver Policy AWS verwalteten Richtlinie sind erforderlich. Sie können eine IAM-Rolle erstellen und ihr die verwaltete Richtlinie mit dem folgenden Befehl anfügen. Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters und *AmazonEKS_EBS_CSI_DriverRole* durch den Namen Ihrer Rolle. Dieser Befehl erfordert, dass Sie eksctl auf Ihrem Gerät installiert haben. Wenn Sie ein anderes Tool oder einen benutzerdefinierten KMS-Schlüssel für die Verschlüsselung verwenden müssen, finden Sie weitere Informationen unter the section called “Schritt 1: Erstellen einer IAM-Rolle”.

```
eksctl create iamserviceaccount \
    --name ebs-csi-controller-sa \
    --namespace kube-system \
    --cluster my-cluster \
    --role-name AmazonEKS_EBS_CSI_DriverRole \
    --role-only \
    --attach-policy-arn arn:aws: iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy
 \
    --approve
```

Zusätzliche Informationen

Weitere Informationen zum Add-On finden Sie unter the section called “Amazon EBS”.

Amazon EFS-CSI-Treiber

Das Amazon-EKS-Add-On für Amazon-EFS-CSI-Treiber ist ein Kubernetes Container Storage Interface (CSI)-Plugin, das Amazon-EFS-Speicher für Ihren Cluster bereitstellt.

Der Name des Amazon-EKS-Add-Ons lautet `aws-efs-csi-driver`.

Erforderliche IAM-Berechtigungen

Erforderliche IAM-Berechtigungen – Dieses Add-On nutzt die Funktion IAM-Rollen für Servicekonten von Amazon EKS. Weitere Informationen finden Sie unter the section called “Anmeldeinformationen mit IRSA”. Die Berechtigungen in der von Amazon EFSCSIDriver Policy AWS verwalteten Richtlinie sind erforderlich. Sie können eine IAM-Rolle erstellen und ihr die verwaltete Richtlinie mit dem folgenden Befehl anfügen. Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters und *AmazonEKS_EFS_CSI_DriverRole* durch den Namen Ihrer Rolle. Diese Befehle erfordern, dass

Sie eksctl auf Ihrem Gerät installiert haben. Wenn Sie ein anderes Tool verwenden müssen, finden Sie Informationen unter the section called “Schritt 1: Erstellen einer IAM-Rolle”.

```
export cluster_name=my-cluster
export role_name=AmazonEKS_EFS_CSI_DriverRole
eksctl create iamserviceaccount \
    --name efs-csi-controller-sa \
    --namespace kube-system \
    --cluster $cluster_name \
    --role-name $role_name \
    --role-only \
    --attach-policy-arn arn:aws: iam::aws:policy/service-role/AmazonEFSCSIDriverPolicy
 \
    --approve
TRUST_POLICY=$(aws iam get-role --output json --role-name $role_name --query
 'Role.AssumeRolePolicyDocument' | \
    sed -e 's/efs-csi-controller-sa/efs-csi-*/' -e 's/StringEquals/StringLike/')
aws iam update-assume-role-policy --role-name $role_name --policy-document
 "$TRUST_POLICY"
```

Zusätzliche Informationen

Weitere Informationen zum Add-On finden Sie unter the section called “Amazon EFS”.

Amazon FSx CSI-Treiber

Das Amazon EKS-Add-on für Amazon FSx CSI-Treiber ist ein Kubernetes Container Storage Interface (CSI) -Plugin, das Amazon FSx for Lustre-Speicher für Ihren Cluster bereitstellt.

Der Name des Amazon-EKS-Add-Ons lautet `aws-fsx-csi-driver`.

> **Note**
>
> - Bereits vorhandene Amazon FSx CSI-Treiberinstallationen im Cluster können zu Fehlern bei der Installation von Add-Ons führen. Wenn Sie versuchen, die Amazon EKS-Add-On-Version zu installieren, während ein FSx Nicht-EKS-CSI-Treiber vorhanden ist, schlägt die Installation aufgrund von Ressourcenkonflikten fehl. Verwenden Sie während der Installation das Flag `OVERWRITE`, um dieses Problem zu beheben:
>
>   ```
>   aws eks create-addon --addon-name aws-fsx-csi-driver --cluster-name my-cluster
>    --resolve-conflicts OVERWRITE
>   ```

- Das Amazon FSx CSI Driver EKS-Add-On benötigt den EKS Pod Identity-Agenten für die Authentifizierung. Ohne diese Komponente schlägt das Add-On mit dem Fehler `Amazon EKS Pod Identity agent is not installed in the cluster` fehl, wodurch Volume-Operationen verhindert werden. Installieren Sie den Pod Identity-Agenten vor oder nach der Bereitstellung des FSx CSI-Treiber-Add-ons. Weitere Informationen finden Sie unter the section called “Agent einrichten”.

Erforderliche IAM-Berechtigungen

Dieses Add-On nutzt die IAM-Rollen für Servicekonten-Funktionalität von Amazon EKS. Weitere Informationen finden Sie unter the section called “Anmeldeinformationen mit IRSA”. Die Berechtigungen in der von Amazon FSx FullAccess AWS verwalteten Richtlinie sind erforderlich. Sie können eine IAM-Rolle erstellen und ihr die verwaltete Richtlinie mit dem folgenden Befehl anfügen. Ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters und *`AmazonEKS_FSx_CSI_DriverRole`* durch den Namen Ihrer Rolle. Dieser Befehl erfordert, dass Sie eksctl auf Ihrem Gerät installiert haben.

```
eksctl create iamserviceaccount \
    --name fsx-csi-controller-sa \
    --namespace kube-system \
    --cluster my-cluster \
    --role-name AmazonEKS_FSx_CSI_DriverRole \
    --role-only \
    --attach-policy-arn arn:aws: iam::aws:policy/AmazonFSxFullAccess \
    --approve
```

Zusätzliche Informationen

Weitere Informationen zum Add-On finden Sie unter the section called “Amazon FSx für Lustre”.

CSI-Treiber für Mountpoint für Amazon S3

Das Amazon-EKS-Add-On für CSI-Treiber für Mountpoint für Amazon S3 ist ein Kubernetes Container Storage Interface (CSI)-Plugin, das Amazon-S3-Speicher für Ihren Cluster bereitstellt.

Der Name des Amazon-EKS-Add-Ons lautet `aws-mountpoint-s3-csi-driver`.

Erforderliche IAM-Berechtigungen

Dieses Add-on verwendet die IAM-Rollen für die Servicekontofunktion von Amazon EKS. Weitere Informationen finden Sie unter the section called “Anmeldeinformationen mit IRSA”.

Für die erstellte IAM-Rolle ist eine Richtlinie erforderlich, die Zugriff auf S3 gewährt. Folgen Sie bei der Erstellung der Richtlinie den Empfehlungen für Mountpoint-IAM-Berechtigungen. Alternativ können Sie die AWS verwaltete Richtlinie AmazonS3 verwendenFullAccess, aber diese verwaltete Richtlinie gewährt mehr Berechtigungen, als für Mountpoint erforderlich sind.

Sie können eine IAM-Rolle erstellen und ihr Ihre Richtlinie mit dem folgenden Befehl anfügen. *my-cluster*Ersetzen Sie es durch den Namen Ihres Clusters, *region-code* durch den richtigen AWS Regionalcode, *AmazonEKS_S3_CSI_DriverRole* durch den Namen für Ihre Rolle und *AmazonEKS_S3_CSI_DriverRole_ARN* durch den Rollen-ARN. Diese Befehle erfordern, dass Sie eksctl auf Ihrem Gerät installiert haben. Anweisungen zur Verwendung der IAM-Konsole oder AWS CLI finden Sie unterthe section called “Schritt 2: Erstellen einer IAM-Rolle”.

```
CLUSTER_NAME=my-cluster
REGION=region-code
ROLE_NAME=AmazonEKS_S3_CSI_DriverRole
POLICY_ARN=AmazonEKS_S3_CSI_DriverRole_ARN
eksctl create iamserviceaccount \
    --name s3-csi-driver-sa \
    --namespace kube-system \
    --cluster $CLUSTER_NAME \
    --attach-policy-arn $POLICY_ARN \
    --approve \
    --role-name $ROLE_NAME \
    --region $REGION \
    --role-only
```

Zusätzliche Informationen

Weitere Informationen zum Add-On finden Sie unter the section called “Mountpoint für Amazon S3”.

CSI-Snapshot-Controller

Der Container Storage Interface (CSI)-Snapshot-Controller ermöglicht die Verwendung der Snapshot-Funktion in kompatiblen CSI-Treibern, wie dem Amazon-EBS-CSI-Treiber.

Der Name des Amazon-EKS-Add-Ons lautet `snapshot-controller`.

Erforderliche IAM-Berechtigungen

Dieses Add-On erfordert keine Berechtigungen.

Zusätzliche Informationen

Weitere Informationen zum Add-On finden Sie unter the section called “CSI-Snapshot-Controller”.

SageMaker HyperPod Amazon-Aufgabenverwaltung

SageMaker HyperPod Task Governance ist ein robustes Managementsystem, das entwickelt wurde, um die Ressourcenzuweisung zu optimieren und eine effiziente Nutzung der Rechenressourcen durch Teams und Projekte für Ihre Amazon EKS-Cluster sicherzustellen. Dadurch haben Administratoren die Möglichkeit, Folgendes festzulegen:

- Prioritätsstufen für verschiedene Aufgaben
- Rechenkapazitätszuweisung für jedes Team
- Wie jedes Team ungenutzte Rechenleistung verleiht und ausleiht
- Wenn ein Team seine eigenen Aufgaben vorwegnimmt

HyperPod Task Governance bietet auch Amazon EKS-Cluster-Observability und bietet so Echtzeiteinblicke in die Clusterkapazität. Dazu gehört die Verfügbarkeit und Nutzung von Rechenleistung, die Zuweisung und Auslastung von Teams sowie Informationen zu Aufgabenausführung und Wartezeiten, sodass Sie fundierte Entscheidungen treffen und Ressourcen proaktiv verwalten können.

Der Name des Amazon-EKS-Add-Ons lautet `amazon-sagemaker-hyperpod-taskgovernance`.

Erforderliche IAM-Berechtigungen

Dieses Add-On erfordert keine Berechtigungen.

Zusätzliche Informationen

Weitere Informationen über das Add-on finden Sie unter SageMaker HyperPod Task Governance

Amazon SageMaker HyperPod Observability-Zusatzmodul

Das Amazon SageMaker HyperPod Observability Add-on bietet umfassende Überwachungs- und Beobachtbarkeitsfunktionen für HyperPod Cluster. Dieses Add-on implementiert und verwaltet automatisch wichtige Überwachungskomponenten wie den Node-Exporter, den DCGM-Exporter

und den EFA-Exporter. kube-state-metrics Es erfasst Metriken und leitet diese an eine vom Kunden festgelegte Amazon Managed Prometheus (AMP)-Instance weiter. Darüber hinaus stellt es einen OTLP-Endpunkt für benutzerdefinierte Metriken und die Erfassung von Ereignissen aus Kunden-Trainingsaufträgen bereit.

Das Add-on lässt sich in das breitere HyperPod Ökosystem integrieren, indem es Metriken aus verschiedenen Komponenten wie dem HyperPod Task Governance-Add-on, Training Operator, HyperPod Kubeflow und KEDA abruft. Alle erfassten Metriken werden in Amazon Managed Prometheus zentralisiert, sodass Kunden über Dashboards zu Amazon Managed Grafana eine einheitliche Übersicht erhalten. Dies bietet end-to-end Einblick in den Zustand des Clusters, die Ressourcennutzung und die Leistung der Schulungsaufgaben in der gesamten Umgebung. HyperPod

Der Name des Amazon-EKS-Add-Ons lautet `amazon-sagemaker-hyperpod-observability`.

Erforderliche IAM-Berechtigungen

Dieses Add-on verwendet die IAM-Rollen für die Servicekontofunktion von Amazon EKS. Weitere Informationen finden Sie unter the section called “Anmeldeinformationen mit IRSA”. Die folgenden verwalteten Richtlinien sind erforderlich.

- `AmazonPrometheusRemoteWriteAccess` – für Schreibmetriken aus der Ferne vom Cluster zu AMP
- `CloudWatchAgentServerPolicy`- für das Remote-Schreiben der Protokolle vom Cluster in CloudWatch

Zusätzliche Informationen

Weitere Informationen über das Add-on und seine Funktionen finden Sie unter SageMaker HyperPod Observability.

SageMaker HyperPod Amazon-Schulungsleiter

Der Amazon SageMaker HyperPod Training Operator hilft Ihnen dabei, die Entwicklung generativer KI-Modelle zu beschleunigen, indem er verteilte Schulungen über große GPU-Cluster effizient verwaltet. Er bietet intelligente Funktionen zur Fehlerbehebung, Erkennung von Blockierungen und Verwaltungsfunktionen auf Prozessebene, die Trainingsunterbrechungen minimieren und Kosten senken. Im Gegensatz zur herkömmlichen Trainingsinfrastruktur, bei der der Job bei Ausfällen komplett neu gestartet werden muss, implementiert dieser Operator die Wiederherstellung chirurgischer Prozesse, um einen reibungslosen Ablauf Ihrer Trainingsaufgaben zu gewährleisten.

Der Operator arbeitet auch mit HyperPod den Funktionen zur Zustandsüberwachung und Beobachtbarkeit und bietet so Echtzeiteinblicke in die Trainingsausführung sowie die automatische Überwachung kritischer Kennzahlen wie Verlustspitzen und Durchsatzeinbußen. Sie können Wiederherstellungsrichtlinien durch einfache YAML-Konfigurationen ohne Codeänderungen definieren, sodass Sie schnell auf nicht wiederherstellbare Trainingszustände reagieren und diese beheben können. Diese Überwachungs- und Wiederherstellungsfunktionen sorgen gemeinsam für eine optimale Trainingsleistung bei minimalem Betriebsaufwand.

Der Name des Amazon-EKS-Add-Ons lautet `amazon-sagemaker-hyperpod-training-operator`.

Weitere Informationen finden Sie unter Using the HyperPod Training Operator im Amazon SageMaker Developer Guide.

Erforderliche IAM-Berechtigungen

Dieses Add-On erfordert IAM-Berechtigungen und verwendet EKS Pod Identity.

AWS schlägt die `AmazonSageMakerHyperPodTrainingOperatorAccess` verwaltete Richtlinie vor.

Weitere Informationen finden Sie unter Installation des Training Operators im Amazon SageMaker Developer Guide.

Zusätzliche Informationen

Weitere Informationen über das Add-on finden Sie unter SageMaker HyperPod Training Operator.

AWS Network Flow Monitor-Agent

Der Amazon CloudWatch Network Flow Monitor Agent ist eine Kubernetes-Anwendung, die TCP-Verbindungsstatistiken von allen Knoten in einem Cluster sammelt und Netzwerkflussberichte für Amazon CloudWatch Network Flow Monitor Ingestion veröffentlicht. APIs

Der Name des Amazon-EKS-Add-Ons lautet `aws-network-flow-monitoring-agent`.

Erforderliche IAM-Berechtigungen

Für dieses Add-On sind IAM -Berechtigungen erforderlich.

Sie müssen die von `CloudWatchNetworkFlowMonitorAgentPublishPolicy` verwaltete Richtlinie an das AddO-n anfügen.

Weitere Informationen zum erforderlichen IAM-Setup finden Sie unter IAM-Richtlinie im Amazon CloudWatch Network Flow Monitor GitHub Agent-Repository.

Weitere Informationen zur verwalteten Richtlinie finden Sie CloudWatchNetworkFlowMonitorAgentPublishPolicyim CloudWatch Amazon-Benutzerhandbuch.

Zusätzliche Informationen

Weitere Informationen über das Add-on finden Sie im Amazon CloudWatch Network Flow Monitor Agent GitHub Repo.

Knotenüberwachungsagent

Das Amazon-EKS-Add-On für den Knoten-Überwachungsagent kann zusätzliche Probleme mit der Knotenintegrität erkennen. Diese zusätzlichen Zustandssignale können auch vom optionalen automatischen Knotenreparaturfeature genutzt werden, um Knoten bei Bedarf automatisch zu ersetzen.

> **Note**
>
> Sie müssen dieses Add-On nicht in Clustern von Amazon EKS Auto Mode installieren. Weitere Informationen finden Sie unter the section called “Überlegungen zu Amazon EKS Auto Mode”.

Der Name des Amazon-EKS-Add-Ons lautet `eks-node-monitoring-agent`.

Erforderliche IAM-Berechtigungen

Dieses Add-On erfordert keine zusätzlichen Berechtigungen.

Zusätzliche Informationen

Weitere Informationen finden Sie unter the section called “Knotenintegrität”.

AWS Distribution für OpenTelemetry

Das Add-on AWS Distro for OpenTelemetry Amazon EKS ist eine sichere, produktionsbereite und AWS unterstützte Distribution des Projekts. OpenTelemetry Weitere Informationen finden Sie unter AWS Distro for on. OpenTelemetry GitHub

Der Name des Amazon-EKS-Add-Ons lautet `adot`.

Erforderliche IAM-Berechtigungen

Für dieses Add-On sind nur dann IAM-Berechtigungen erforderlich, wenn Sie eine der vorkonfigurierten benutzerdefinierten Ressourcen verwenden, die über die erweiterte Konfiguration aktiviert werden können.

Zusätzliche Informationen

Weitere Informationen finden Sie in der Dokumentation unter Erste Schritte mit AWS Distro für die OpenTelemetry Verwendung von EKS-Add-Ons in der AWS Distro. OpenTelemetry

ADOT setzt voraus, dass das `cert-manager` Add-on auf dem Cluster bereitgestellt wird. Andernfalls funktioniert dieses Add-on nicht, wenn es direkt über https://registry.terraform bereitgestellt wird. io/modules/terraform-aws-modules/eks/aws/latest`cluster_addons`Eigentum. Weitere Anforderungen finden Sie in der Dokumentation unter Anforderungen für die ersten Schritte mit AWS Distro für die OpenTelemetry Verwendung von EKS-Add-Ons in der AWS Distro. OpenTelemetry

GuardDuty Amazon-Vertreter

Das Amazon EKS-Add-on für GuardDuty Agenten sammelt Laufzeitereignisse (Dateizugriff, Prozessausführung, Netzwerkverbindungen) von Ihren EKS-Clusterknoten zur Analyse durch GuardDuty Runtime Monitoring. GuardDuty selbst (nicht der Agent) ist der Sicherheitsüberwachungsdienst, der grundlegende Datenquellen wie AWS CloudTrail Verwaltungsereignisse und Amazon VPC-Flow-Logs sowie Funktionen wie Kubernetes-Auditprotokolle und Laufzeitüberwachung analysiert und verarbeitet.

Der Name des Amazon-EKS-Add-Ons lautet `aws-guardduty-agent`.

Erforderliche IAM-Berechtigungen

Dieses Add-On erfordert keine Berechtigungen.

Zusätzliche Informationen

Weitere Informationen finden Sie unter Laufzeitüberwachung für Amazon EKS-Cluster in Amazon GuardDuty.

- Um potenzielle Sicherheitsbedrohungen in Ihren Amazon EKS-Clustern zu erkennen, aktivieren Sie Amazon GuardDuty Runtime Monitoring und stellen Sie den GuardDuty Security Agent auf Ihren Amazon EKS-Clustern bereit.

Amazon CloudWatch Observability-Agent

Der Amazon CloudWatch Observability-Agent Amazon EKS erweitert den Überwachungs- und Observabilitätsservice von. AWS Dieses Add-on installiert den CloudWatch Agenten und aktiviert sowohl CloudWatch Application Signals als auch CloudWatch Container Insights mit verbesserter Observability für Amazon EKS. Weitere Informationen finden Sie unter Amazon CloudWatch Agent.

Der Name des Amazon-EKS-Add-Ons lautet `amazon-cloudwatch-observability`.

Erforderliche IAM-Berechtigungen

Dieses Add-on verwendet die IAM-Rollen für die Servicekontofunktion von Amazon EKS. Weitere Informationen finden Sie unter the section called “Anmeldeinformationen mit IRSA”. Die Berechtigungen in den AWSXrayWriteOnlyAccessund den CloudWatchAgentServerPolicy AWS verwalteten Richtlinien sind erforderlich. Sie können eine IAM-Rolle erstellen, ihr die verwalteten Richtlinien anfügen und das vom Add-On verwendete Kubernetes-Servicekonto mit dem folgenden Befehl annotieren. Ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters und *`AmazonEKS_Observability_role`* durch den Namen Ihrer Rolle. Dieser Befehl erfordert, dass Sie eksctl auf Ihrem Gerät installiert haben. Wenn Sie ein anderes Tool verwenden müssen, um die Rolle zu erstellen, ihr die Richtlinie zuzuordnen und das Kubernetes-Servicekonto mit Annotationen zu versehen, siehe the section called “IAM-Rolle zuweisen”.

```
eksctl create iamserviceaccount \
    --name cloudwatch-agent \
    --namespace amazon-cloudwatch \
    --cluster my-cluster \
    --role-name AmazonEKS_Observability_Role \
    --role-only \
    --attach-policy-arn arn:aws: iam::aws:policy/AWSXrayWriteOnlyAccess \
    --attach-policy-arn arn:aws: iam::aws:policy/CloudWatchAgentServerPolicy \
    --approve
```

Zusätzliche Informationen

Weitere Informationen finden Sie unter Den CloudWatch Agenten installieren.

AWS Privater CA-Connector für Kubernetes

Der AWS Private CA Connector for Kubernetes ist ein Add-on für Cert-Manager, mit dem Benutzer Zertifikate von einer AWS privaten Zertifizierungsstelle (Private CA) erhalten können.AWS

- Der Name des Amazon-EKS-Add-Ons lautet `aws-privateca-connector-for-kubernetes`.
- Der Add-On-Namespace lautet `aws-privateca-issuer`.

Dieses Add-On erfordert `cert-manager`. `cert-manager` ist in Amazon EKS als Community-Add-On verfügbar. Weitere Informationen zu diesem Add-On finden Sie unter the section called “Cert Manager”. Weitere Informationen zur Installation von Add-Ons finden Sie unter the section called “Add-On erstellen”.

Erforderliche IAM-Berechtigungen

Für dieses Add-On sind IAM-Berechtigungen erforderlich.

Verwenden Sie EKS Pod Identities, um die `AWSPrivateCAConnectorForKubernetesPolicy`-IAM-Richtlinie an das `aws-privateca-issuer`-Kubernetes-Servicekonto anzufügen. Weitere Informationen finden Sie unter the section called “Pod Identities verwenden”.

Informationen zu den erforderlichen Berechtigungen finden Sie AWSPrivateCAConnectorForKubernetesPolicyin der Managed Policy Reference. AWS

Zusätzliche Informationen

Weitere Informationen finden Sie im Repository AWS Private CA Issuer for Kubernetes GitHub .

Weitere Informationen zur Konfiguration des Add-ons finden Sie unter values.yaml im Repo. `aws-privateca-issuer` GitHub Bestätigen Sie, dass die Version von values.yaml mit der Version des in Ihrem Cluster installierten Add-Ons übereinstimmt.

Dieses Add-on toleriert den `CriticalAddonsOnly` Makel, der im automatischen Modus von EKS verwendet wird. `system` NodePool Weitere Informationen finden Sie unter the section called “Kritische Add-Ons ausführen”.

EKS Pod Identity Agent

Das Amazon EKS Pod Identity Agent Amazon EKS Add-on bietet die Möglichkeit, Anmeldeinformationen für Ihre Anwendungen zu verwalten, ähnlich wie EC2 Instance-Profile Anmeldeinformationen für EC2 Instances bereitstellen.

> **Note**
>
> Sie müssen dieses Add-On nicht in Clustern von Amazon EKS Auto Mode installieren. Amazon EKS Auto Mode ist in EKS Pod Identity integriert. Weitere Informationen finden Sie unter the section called “Überlegungen zu Amazon EKS Auto Mode”.

Der Name des Amazon-EKS-Add-Ons lautet `eks-pod-identity-agent`.

Erforderliche IAM-Berechtigungen

Das Pod-Identity-Agent-Add-On selbst erfordert keine IAM-Rolle. Es nutzt Berechtigungen der IAM-Rolle des Amazon-EKS-Knotens, benötigt jedoch keine dedizierte IAM-Rolle für das Add-On.

Aktualisieren der Informationen

Sie können jeweils nur eine Nebenversion aktualisieren. Wenn Ihre aktuelle Version beispielsweise `1.28.x-eksbuild.y` ist und Sie auf `1.30.x-eksbuild.y` aktualisieren möchten, müssen Sie zuerst ihre aktuelle Version auf `1.29.x-eksbuild.y` und dann auf `1.30.x-eksbuild.y` aktualisieren. Weitere Informationen zum Aktualisieren des Add-ons finden Sie unter the section called “Add-On aktualisieren”.

SR-IOV-Netzwerkmetrik-Exportprogramm

Das Amazon-EKS-Add-On für das SR-IOV-Netzwerkmetrik-Exportprogramm erfasst und stellt Metriken zu SR-IOV-Netzwerkgeräten im Prometheus-Format bereit. Es ermöglicht die Überwachung der SR-IOV-Netzwerkleistung in EKS-Bare-Metal-Knoten. Der Exporter läuft als DaemonSet On-Knoten mit SR-IOV-capable Netzwerkschnittstellen und exportiert Metriken, die von Prometheus gescrapt werden können.

> **Note**
>
> Für dieses Add-on sind Knoten mit Netzwerkschnittstellen erforderlich. SR-IOV-capable

Eigenschaft	Wert
Add-On-Name	`sriov-network-metrics-exporter`

Eigenschaft	Wert
Namespace	`monitoring`
Dokumentation	Das SR-IOV Network Metrics Exporter-Repository GitHub
Servicekonto-Name	Keine
Verwaltete IAM-Richtlinie	Keine
Benutzerdefinierte IAM-Berechtigungen	Keine

AWS Secrets Store CSI-Treiberanbieter

Der AWS Anbieter für den Secrets Store CSI Driver ist ein Add-on, mit dem AWS Secrets Manager und Parameter aus dem AWS Systems Manager Parameter Store abgerufen und als Dateien in Kubernetes-Pods bereitgestellt werden können.

Erforderliche IAM-Berechtigungen

Für das Add-on sind keine IAM-Berechtigungen erforderlich. Anwendungs-Pods benötigen jedoch IAM-Berechtigungen, um Secrets aus AWS Secrets Manager und Parameter aus dem AWS Systems Manager Parameter Store abzurufen. Nach der Installation des Add-ons muss der Zugriff über IAM Roles for Service Accounts (IRSA) oder EKS Pod Identity konfiguriert werden. Informationen zur Verwendung von IRSA finden Sie in der Secrets Manager IRSA-Setup-Dokumentation. Informationen zur Verwendung von EKS Pod Identity finden Sie in der Setup-Dokumentation für Secrets Manager Pod Identity.

AWS schlägt die `AWSSecretsManagerClientReadOnlyAccess` verwaltete Richtlinie vor.

Weitere Informationen zu den erforderlichen Berechtigungen finden Sie `AWSSecretsManagerClientReadOnlyAccess` in der Referenz zu AWS verwalteten Richtlinien.

Zusätzliche Informationen

Weitere Informationen finden Sie im secrets-store-csi-driver GitHub -provider-aws-Repository.

Weitere Informationen zum Add-on finden Sie in der AWS Secrets Manager Manager-Dokumentation für das Add-on.

Amazon SageMaker Spaces

Das Amazon SageMaker Spaces-Add-on ermöglicht die Ausführung IDEs von Notebooks auf EKS- oder HyperPod -EKS-Clustern. Administratoren können die EKS-Konsole verwenden, um das Add-on auf ihrem Cluster zu installieren und Standardspeicherkonfigurationen wie Bilder, Rechenressourcen, lokalen Speicher für Notebook-Einstellungen (zusätzlicher Speicher, der an ihre Spaces angehängt werden muss), Dateisysteme und Initialisierungsskripten zu definieren.

KI-Entwickler können kubectl verwenden, um Spaces zu erstellen, zu aktualisieren und zu löschen. Sie haben die Flexibilität, von Administratoren bereitgestellte Standardkonfigurationen zu verwenden oder Einstellungen anzupassen. KI-Entwickler können mithilfe ihres lokalen VS Codes, ihres Webbrowsers, der sie hostet, oder ihrer CodeEditor IDE auf einer benutzerdefinierten DNS-Domain IDEs, and/or die von ihren Administratoren konfiguriert wurde, auf ihre Bereiche in HyperPod EKS JupyterLab oder -EKS zugreifen. Sie können auch die Portweiterleitungsfunktion von Kubernetes verwenden, um auf Bereiche in ihren Webbrowsern zuzugreifen.

Der Name des Amazon-EKS-Add-Ons lautet `amazon-sagemaker-spaces`.

Erforderliche IAM-Berechtigungen

Für dieses Add-On sind IAM-Berechtigungen erforderlich. Weitere Informationen zur erforderlichen IAM-Einrichtung finden Sie unter Einrichtung von IAM-Berechtigungen im Amazon SageMaker Developer Guide.

Zusätzliche Informationen

Weitere Informationen über das Add-on und seine Funktionen finden Sie unter SageMaker AI Notebooks HyperPod im Amazon SageMaker Developer Guide.

Community-Erweiterungen

Sie können AWS APIs Community-Add-Ons wie den Kubernetes Metrics Server installieren. Sie haben die Möglichkeit, Community-Add-Ons als Amazon-EKS-Add-Ons zu installieren, um die Komplexität der Wartung der Software auf mehreren Clustern zu reduzieren.

Sie können beispielsweise die AWS API, CLI oder die Management Console verwenden, um Community-Add-Ons zu installieren. Sie können während der Cluster-Erstellung ein Community-Add-On installieren.

Sie verwalten Community-Add-Ons genauso wie vorhandene Amazon-EKS-Add-Ons. Community-Add-Ons unterscheiden sich von vorhandenen Add-Ons dadurch, dass sie einen einzigartigen Support-Umfang bieten.

> **Note**
>
> Die Verwendung von Community-Add-Ons liegt in Ihrem Ermessen. Im Rahmen des Modells der gemeinsamen Verantwortung zwischen Ihnen und Ihnen wird erwartet AWS, dass Sie wissen, was Sie für diese Drittanbieter-Plugins in Ihrem Cluster installieren. Sie sind auch dafür verantwortlich, dass die Community-Add-Ons Ihren Cluster-Sicherheitsanforderungen entsprechen. Weitere Informationen finden Sie unter the section called “Support für in EKS bereitgestellte Software”.

Community-Add-Ons wurden nicht von entwickelt AWS. AWS validiert nur Community-Add-Ons auf Versionskompatibilität. Wenn Sie beispielsweise ein Community-Add-on auf einem Cluster installieren, wird AWS geprüft, ob es mit der Kubernetes-Version Ihres Clusters kompatibel ist.

Wichtig ist, dass Community-Add-Ons AWS nicht vollständig unterstützt werden. AWS unterstützt nur Lebenszyklusvorgänge, die mithilfe von Add-Ons ausgeführt werden AWS APIs, z. B. das Installieren von Add-Ons oder das Löschen von Add-Ons

Wenn Sie Support für ein Community-Add-On benötigen, nutzen Sie die vorhandenen Projektressourcen. Sie können beispielsweise ein GitHub Problem im Repository für das Projekt erstellen.

Add-On-Typ ermitteln

Sie können die AWS CLI verwenden, um den Typ eines Amazon EKS-Add-ons zu bestimmen.

Verwenden Sie den folgenden CLI-Befehl, um Informationen zu einem Add-On abzurufen. Sie können `metrics-server` durch den Namen eines beliebigen Add-Ons ersetzen.

```
aws eks describe-addon-versions --addon-name metrics-server
```

Überprüfen Sie die CLI-Ausgabe für das `owner`-Feld.

```
{
    "addons": [
        {
```

```
            "addonName": "metrics-server",
            "type": "observability",
            "owner": "community",
            "addonVersions": [
```

Wenn der Wert von gleich `owner` ist`community`, handelt es sich bei dem Add-On um ein Community-Add-On. AWS bietet nur Unterstützung für die Installation, Aktualisierung und Deinstallation des Add-ons. Wenn Sie Fragen zur Funktionalität und zum Betrieb des Add-ons selbst haben, nutzen Sie Community-Ressourcen wie GitHub Issues.

Community-Add-On installieren oder aktualisieren

Sie installieren oder aktualisieren Community-Add-Ons auf dieselbe Weise wie andere Amazon-EKS-Add-Ons.

- the section called “Add-On erstellen”
- the section called “Add-On aktualisieren”
- the section called “Add-On entfernen”

Verfügbare Community-Add-Ons

Die folgenden Community-Add-Ons sind bei Amazon EKS verfügbar.

- the section called “Kubernetes-Metriken-Server”
- the section called “kube-state-metrics”
- the section called “Prometheus-Knoten-Exporter”
- the section called “Cert Manager”
- the section called “Externes DNS”
- the section called “Fluent Bit”

Kubernetes-Metriken-Server

Der Kubernetes-Metriken-Server ist eine skalierbare und effiziente Quelle für Container-Ressourcen-Metriken für die in Kubernetes integrierten Pipelines zur automatischen Skalierung. Es erfasst Ressourcenmetriken von Kubelets und stellt sie über die Metrik-API im Kubernetes-Apiserver zur Verfügung, damit sie vom Horizontal Pod Autoscaler und Vertical Pod Autoscaler genutzt werden können.

Eigenschaft	Wert
Add-On-Name	`metrics-server`
Namespace	`kube-system`
Dokumentation	GitHub Readme
Servicekonto-Name	Keine
Verwaltete IAM-Richtlinie	Keine
Benutzerdefinierte IAM-Berechtigungen	Keine

kube-state-metrics

Add-On-Agent zum Generieren und Bereitstellen von Metriken auf Cluster-Ebene.

Der Status von Kubernetes-Objekten in der Kubernetes-API kann als Metriken dargestellt werden. Ein sogenannter Add-On-Agent kube-state-metrics kann eine Verbindung zum Kubernetes-API-Server herstellen und einen HTTP-Endpunkt mit Metriken bereitstellen, die aus dem Status einzelner Objekte im Cluster generiert werden. Es werden verschiedene Informationen über den Status von Objekten angezeigt, wie Beschriftungen und Annotationen, Startup- und Beendigungszeiten, Status oder die Phase, in der sich das Objekt derzeit befindet.

Eigenschaft	Wert
Add-On-Name	`kube-state-metrics`
Namespace	`kube-state-metrics`
Dokumentation	Metriken für Kubernetes-Objektzustände in der Kubernetes-Dokumentation
Servicekonto-Name	Keine
Verwaltete IAM-Richtlinie	Keine
Benutzerdefinierte IAM-Berechtigungen	Keine

Prometheus-Knoten-Exporter

Prometheus-Exporter für Hardware- und Betriebssystemmetriken, die von *NIX-Kernels bereitgestellt werden, geschrieben in Go mit pluggbaren Metrik-Kollektoren. Der Prometheus-Knoten-Exporter stellt eine Vielzahl von hardware- und kernellastigen Metriken bereit.

Eigenschaft	Wert
Add-On-Name	`prometheus-node-exporter`
Namespace	`prometheus-node-exporter`
Dokumentation	Überwachung von Linux-Host-Metriken mit dem Knoten-Exporter in Prometheus-Dokumentation
Servicekonto-Name	Keine
Verwaltete IAM-Richtlinie	Keine
Benutzerdefinierte IAM-Berechtigungen	Keine

Cert Manager

Cert Manager kann zur Verwaltung der Erstellung und Erneuerung von Zertifikaten verwendet werden.

Eigenschaft	Wert
Add-On-Name	`cert-manager`
Namespace	`cert-manager`
Dokumentation	Cert-Manager-Dokumente
Servicekonto-Name	Keine
Verwaltete IAM-Richtlinie	Keine
Benutzerdefinierte IAM-Berechtigungen	Keine

Externes DNS

Mit dem Add-On Externes DNS EKS können Sie Route53-DNS-Einträge über Kubernetes-Ressourcen verwalten.

Externe DNS-Berechtigungen können für die gehosteten Zonen, die Sie verwalten möchten, auf `route53:ChangeResourceRecordSets`, `route53:ListHostedZones` und `route53:ListResourceRecordSets` reduziert werden.

Eigenschaft	Wert
Add-On-Name	`external-dns`
Namespace	`external-dns`
Dokumentation	GitHub Readme
Servicekonto-Name	`external-dns`
Verwaltete IAM-Richtlinie	`arn:aws: iam::aws:policy/AmazonRoute53FullAccess`
Benutzerdefinierte IAM-Berechtigungen	Keine

Fluent Bit

Fluent Bit ist ein unkomplizierter und leistungsstarker Protokollprozessor und -weiterleiter. Es ermöglicht Ihnen, Daten data/logs aus verschiedenen Quellen zu sammeln, zu vereinheitlichen und an mehrere Ziele zu senden, darunter Amazon CloudWatch Logs, Amazon S3 und Amazon Kinesis Data Firehose. Fluent Bit wurde im Hinblick auf Leistung und Ressourceneffizienz entwickelt und ist daher ideal für Kubernetes-Umgebungen.

Dieses Add-On erfordert in der Standardkonfiguration keine IAM-Berechtigungen. Möglicherweise müssen Sie diesem Add-on jedoch IAM-Berechtigungen erteilen, wenn Sie einen Ausgabespeicherort konfigurieren. AWS Weitere Informationen finden Sie unter the section called “Pod Identities verwenden”.

Eigenschaft	Wert
Add-On-Name	`fluent-bit`
Namespace	`fluent-bit`
Dokumentation	Fluent-Bit-Dokumentation
Servicekonto-Name	`fluent-bit`
Verwaltete IAM-Richtlinie	Keine
Benutzerdefinierte IAM-Berechtigungen	Keine

Zuordnungen anzeigen

Sie können die Open-Source-Zuordnungen und Lizenzinformationen für Community-Add-Ons herunterladen.

1. Ermitteln Sie den Namen und die Version des Add-Ons, für das Sie die Zuordnungen herunterladen möchten.
2. Aktualisieren Sie den folgenden Befehl mit dem Namen und der Version:

   ```
   curl -O https://amazon-eks-docs.s3.amazonaws.com/attributions/<add-on-name>/<add-on-version>/attributions.zip
   ```

   Beispiel:

   ```
   curl -O https://amazon-eks-docs.s3.amazonaws.com/attributions/kube-state-metrics/v2.14.0-eksbuild.1/attributions.zip
   ```

3. Verwenden Sie den Befehl, um die Datei herunterzuladen.

Verwenden Sie diese ZIP-Datei, um Informationen zu den Lizenz-Zuordnungen anzuzeigen.

AWS Marketplace-Add-ons

Zusätzlich zur vorherigen Liste der Amazon-EKS-Add-Ons können Sie auch eine große Auswahl an Amazon-EKS-Add-Ons für Betriebssoftware von unabhängigen Softwareanbietern hinzufügen. Wählen Sie ein Add-on aus, um mehr darüber und seine Installationsanforderungen zu erfahren.

Accuknox

Der Add-On-Name lautet `accuknox_kubearmor` und der Namespace ist `kubearmor`. Accuknox veröffentlicht das Add-On.

Informationen zum Add-on finden Sie KubeArmor in der KubeArmor Dokumentation unter Erste Schritte mit.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Akuity

Der Add-On-Name lautet `akuity_agent` und der Namespace ist `akuity`. Akuity veröffentlicht das Add-On.

Informationen zur Verwendung des Add-Ons finden Sie unter Installieren des Akuity-Agenten in Amazon EKS mit dem Akuity-EKS-Add-On in der Dokumentation zur Akuity-Plattform.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Calyptia

Der Add-On-Name lautet `calyptia_fluent-bit` und der Namespace ist `calytia-fluentbit`. Calyptia veröffentlicht das Add-On.

Informationen zum Add-on finden Sie unter Erste Schritte mit dem Calyptia-Haupt-Agent auf der Calyptia-Dokumentations-Website.

Servicekonto-Name

Der Name des Servicekontos lautet `clyptia-fluentbit`.

AWS verwaltete IAM-Richtlinie

Dieses Add-On verwendet die von `AWSMarketplaceMeteringRegisterUsage` verwaltete Richtlinie. Weitere Informationen finden Sie AWSMarketplaceMeteringRegisterUsageim Referenzhandbuch für AWS verwaltete Richtlinien.

Befehl zum Erstellen der erforderlichen IAM-Rolle

Für den folgenden Befehl ist es erforderlich, dass Sie über einen IAM OpenID Connect (OIDC)-Anbieter für Ihren Cluster verfügen. Um festzustellen, ob Sie über einen solchen verfügen oder um einen zu erstellen, lesen Sie the section called “IAM-OIDC-Anbieter”. Ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters und *`my-calyptia-role`* durch den Namen Ihrer Rolle. Dieser Befehl erfordert, dass Sie eksctl auf Ihrem Gerät installiert haben. Wenn Sie ein anderes Tool verwenden müssen, um die Rolle zu erstellen und das Kubernetes-Servicekonto zu annotieren, lesen Sie the section called “IAM-Rolle zuweisen”.

```
eksctl create iamserviceaccount --name service-account-name  --namespace calyptia-
fluentbit --cluster my-cluster --role-name my-calyptia-role \
    --role-only --attach-policy-arn arn:aws: iam::aws:policy/
AWSMarketplaceMeteringRegisterUsage --approve
```

Cisco-Beobachtbarkeits-Collector

Der Add-On-Name lautet `cisco_cisco-cloud-observability-collectors` und der Namespace ist `appdynamics`. Cisco veröffentlicht das Add-On.

Informationen zum Add-on finden Sie unter Verwenden der Cisco Cloud Observability AWS Marketplace-Add-Ons in der AppDynamics Cisco-Dokumentation.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Cisco-Beobachtbarkeits-Operator

Der Add-On-Name lautet `cisco_cisco-cloud-observability-operators` und der Namespace ist `appdynamics`. Cisco veröffentlicht das Add-On.

Informationen zum Add-on finden Sie unter Verwenden der Cisco Cloud Observability AWS Marketplace-Add-Ons in der AppDynamics Cisco-Dokumentation.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

CLOUDSOFT

Der Add-On-Name lautet `cloudsoft_cloudsoft-amp` und der Namespace ist `cloudsoft-amp`. CLOUDSOFT veröffentlicht das Add-On.

Informationen zum Add-On finden Sie unter Amazon-EKS-ADDON in der CLOUDSOFT-Dokumentation.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Cribl

Der Add-On-Name lautet `cribl_cribledge` und der Namespace ist `cribledge`. Cribl veröffentlicht das Add-On.

Informationen zum Add-On finden Sie unter Installieren des Cribl-Amazon-EKS-Add-Ons für Edge in der Cribl-Dokumentation.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Dynatrace

Der Add-On-Name lautet `dynatrace_dynatrace-operator` und der Namespace ist `dynatrace`. Dynatrace veröffentlicht das Add-On.

Informationen zum Add-On finden Sie unter Kubernetes-Überwachung in der Dynatrace-Dokumentation.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Datree

Der Add-On-Name lautet `datree_engine-pro` und der Namespace ist `datree`. Datree veröffentlicht das Add-On.

Informationen zum Add-On finden Sie unter Amazon-EKS-Integration in der Datree-Dokumentation.

Servicekonto-Name

Der Name des Dienstkontos lautet datree-webhook-server-awsmp.

AWS verwaltete IAM-Richtlinie

Die verwaltete Richtlinie ist AWSLicenseManagerConsumptionPolicy. Weitere Informationen finden Sie AWSLicenseManagerConsumptionPolicyim Referenzhandbuch für AWS verwaltete Richtlinien..

Befehl zum Erstellen der erforderlichen IAM-Rolle

Für den folgenden Befehl ist es erforderlich, dass Sie über einen IAM OpenID Connect (OIDC)-Anbieter für Ihren Cluster verfügen. Um festzustellen, ob Sie über einen solchen verfügen oder um einen zu erstellen, lesen Sie the section called “IAM-OIDC-Anbieter”. Ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters und *`my-datree-role`* durch den Namen Ihrer Rolle. Dieser Befehl erfordert, dass Sie eksctl auf Ihrem Gerät installiert haben. Wenn Sie ein anderes Tool verwenden müssen, um die Rolle zu erstellen und das Kubernetes-Servicekonto zu annotieren, lesen Sie the section called “IAM-Rolle zuweisen”.

```
eksctl create iamserviceaccount --name datree-webhook-server-awsmp --namespace datree
 --cluster my-cluster --role-name my-datree-role \
    --role-only --attach-policy-arn arn:aws: iam::aws:policy/service-role/
AWSLicenseManagerConsumptionPolicy --approve
```

Benutzerdefinierte Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Datadog

Der Add-On-Name lautet `datadog_operator` und der Namespace ist `datadog-agent`. Datadog veröffentlicht das Add-On.

Informationen zum Add-on finden Sie unter Installieren des Datadog-Agenten in Amazon EKS mit dem Datadog-Operator-Add-On in der Datadog-Dokumentation.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Groundcover

Der Add-On-Name lautet `groundcover_agent` und der Namespace lautet `groundcover`. Groundcover veröffentlicht das Add-On.

Informationen zum Add-on finden Sie unter Installieren des Groundcover-Amazon-EKS-Add-Ons in der Groundcover-Dokumentation.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

IBM Instana

Der Add-On-Name lautet `instana-agent` und der Namespace ist `instana-agent`. IBM veröffentlicht das Add-On.

Informationen zum Add-on finden Sie unter Implementieren von Observability für Amazon EKS-Workloads mithilfe des Instana Amazon EKS-Add-ons und Überwachen und Optimieren der Amazon EKS-Kosten mit IBM Instana und Kubecost im Blog. AWS

Instana-Beobachtbarkeit (Instana) bietet ein Amazon-EKS-Add-On, das Instana-Agenten in Amazon-EKS-Clustern bereitstellt. Kunden können dieses Add-On verwenden, um Leistungsdaten in Echtzeit zu erfassen und zu analysieren und so Einblicke in ihre containerisierten Anwendungen zu gewinnen. Das Amazon-EKS-Add-On von Instana bietet Sichtbarkeit über Ihre Kubernetes-Umgebungen hinweg. Nach der Bereitstellung erkennt der Instana-Agent automatisch Komponenten innerhalb Ihrer Amazon-EKS-Cluster, darunter Knoten, Namespaces, Bereitstellungen, Services und Pods.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Grafana Labs

Der Add-On-Name lautet `grafana-labs_kubernetes-monitoring` und der Namespace ist `monitoring`. Grafana Labs veröffentlicht das Add-On.

Informationen zum Add-On finden Sie unter Konfigurieren der Kubernetes-Überwachung als Add-On mit Amazon EKS in der Grafana-Labs-Dokumentation.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Guance

- Publisher – GUANCE
- Name (Name – `guance_datakit`
- Namespace – `datakit`
- Servicekonto-Name – Ein Servicekonto wird mit diesem Add-On nicht verwendet.
- Von AWS verwaltete IAM-Richtlinie – Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.
- Benutzerdefinierte IAM-Berechtigungen – Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.
- Anweisungen zur Einrichtung und Verwendung – Lesen Sie Verwenden des Amazon-EKS-Add-Ons in der Guance-Dokumentation.

HA Proxy

Der Name lautet `haproxy-technologies_kubernetes-ingress-ee` und der Namespace ist `haproxy-controller`. HA Proxy veröffentlicht das Add-On.

Informationen zum Add-On finden Sie unter Amazon-EKS-Integration in der Datree-Dokumentation.

Servicekonto-Name

Der Name des Servicekontos lautet `customer defined`.

AWS verwaltete IAM-Richtlinie

Die verwaltete Richtlinie ist AWSLicenseManagerConsumptionPolicy. Weitere Informationen finden Sie AWSLicenseManagerConsumptionPolicyim Referenzhandbuch für AWS verwaltete Richtlinien..

Befehl zum Erstellen der erforderlichen IAM-Rolle

Für den folgenden Befehl ist es erforderlich, dass Sie über einen IAM OpenID Connect (OIDC)-Anbieter für Ihren Cluster verfügen. Um festzustellen, ob Sie über einen solchen verfügen oder um einen zu erstellen, lesen Sie the section called “IAM-OIDC-Anbieter”. Ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters und *`my-haproxy-role`* durch den Namen Ihrer Rolle. Dieser Befehl erfordert, dass Sie eksctl auf Ihrem Gerät installiert haben. Wenn Sie ein anderes Tool verwenden müssen, um die Rolle zu erstellen und das Kubernetes-Servicekonto zu annotieren, lesen Sie the section called “IAM-Rolle zuweisen”.

```
eksctl create iamserviceaccount --name service-account-name  --namespace haproxy-
controller --cluster my-cluster --role-name my-haproxy-role \
    --role-only --attach-policy-arn arn:aws: iam::aws:policy/service-role/
AWSLicenseManagerConsumptionPolicy --approve
```

Benutzerdefinierte Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Kpow

Der Add-On-Name lautet `factorhouse_kpow` und der Namespace ist `factorhouse`. Factorhouse veröffentlicht das Add-On.

Informationen zum Add-On finden Sie unter AWS Marketplace LM in der Kpow-Dokumentation.

Servicekonto-Name

Der Name des Servicekontos lautet `kpow`.

AWS verwaltete IAM-Richtlinie

Die verwaltete Richtlinie ist AWSLicenseManagerConsumptionPolicy. Weitere Informationen finden Sie AWSLicenseManagerConsumptionPolicyim Referenzhandbuch für AWS verwaltete Richtlinien..

Befehl zum Erstellen der erforderlichen IAM-Rolle

Für den folgenden Befehl ist es erforderlich, dass Sie über einen IAM OpenID Connect (OIDC)-Anbieter für Ihren Cluster verfügen. Um festzustellen, ob Sie über einen solchen verfügen oder um einen zu erstellen, lesen Sie the section called “IAM-OIDC-Anbieter”. Ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters und *`my-kpow-role`* durch den Namen Ihrer Rolle. Dieser Befehl erfordert, dass Sie eksctl auf Ihrem Gerät installiert haben. Wenn Sie ein anderes Tool verwenden müssen, um die Rolle zu erstellen und das Kubernetes-Servicekonto zu annotieren, lesen Sie the section called “IAM-Rolle zuweisen”.

```
eksctl create iamserviceaccount --name kpow --namespace factorhouse --cluster my-
cluster --role-name my-kpow-role \
    --role-only --attach-policy-arn arn:aws: iam::aws:policy/service-role/
AWSLicenseManagerConsumptionPolicy --approve
```

Benutzerdefinierte Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Kubecost

Der Add-On-Name lautet `kubecost_kubecost` und der Namespace ist `kubecost`. Kubecost veröffentlicht das Add-On.

Informationen zum Add-On finden Sie unter Integration der AWS -Cloud- Fakturierung in der Kubecost-Dokumentation.

Sie müssen die Speichern von Kubernetes-Volumes mit Amazon EBS auf Ihrem Cluster installiert haben. Andernfalls erhalten Sie eine Fehlermeldung.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Kasten

Der Add-On-Name lautet `kasten_k10` und der Namespace ist `kasten-io`. Kasten von Veeam veröffentlicht das Add-On.

Informationen zum Add-on finden Sie unter Installation von K10 unter AWS Verwendung des Amazon EKS Add-ons in der Kasten-Dokumentation.

Sie müssen den Amazon-EBS-CSI-Treiber mit einem Standard-`StorageClass` auf Ihrem Cluster installiert haben.

Servicekonto-Name

Der Name des Servicekontos lautet `k10-k10`.

AWS verwaltete IAM-Richtlinie

Die verwaltete Richtlinie ist AWSLicenseManagerConsumptionPolicy. Weitere Informationen finden Sie AWSLicenseManagerConsumptionPolicyim Referenzhandbuch für AWS verwaltete Richtlinien..

Befehl zum Erstellen der erforderlichen IAM-Rolle

Für den folgenden Befehl ist es erforderlich, dass Sie über einen IAM OpenID Connect (OIDC)-Anbieter für Ihren Cluster verfügen. Um festzustellen, ob Sie über einen solchen verfügen oder um einen zu erstellen, lesen Sie the section called “IAM-OIDC-Anbieter”. Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters und *my-kasten-role* durch den Namen Ihrer Rolle. Dieser Befehl erfordert, dass Sie eksctl auf Ihrem Gerät installiert haben. Wenn Sie ein anderes Tool verwenden müssen, um die Rolle zu erstellen und das Kubernetes-Servicekonto zu annotieren, lesen Sie the section called “IAM-Rolle zuweisen”.

```
eksctl create iamserviceaccount --name k10-k10 --namespace kasten-io --cluster my-
cluster --role-name my-kasten-role \
    --role-only --attach-policy-arn arn:aws: iam::aws:policy/service-role/
AWSLicenseManagerConsumptionPolicy --approve
```

Benutzerdefinierte Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Kong

Der Add-On-Name lautet `kong_konnect-ri` und der Namespace ist `kong`. Kong veröffentlicht das Add-On.

Informationen zum Add-On finden Sie unter Installieren des EKS-Add-Ons für Kong-Gateway in der Kong-Dokumentation.

Sie müssen die Speichern von Kubernetes-Volumes mit Amazon EBS auf Ihrem Cluster installiert haben. Andernfalls erhalten Sie eine Fehlermeldung.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

LeakSignal

Der Name des Add-ons lautet `leaksignal_leakagent` und der Namespace ist`leakagent`. LeakSignal veröffentlicht das Add-on.

Informationen zum Add-on finden Sie in der Dokumentation unter https://www.leaksignal.com/docs/LeakAgent /Deployment/AWS%20EKS%20Addon/ [Das Add-on installieren] LeakAgent LeakSignal

Sie müssen die Speichern von Kubernetes-Volumes mit Amazon EBS auf Ihrem Cluster installiert haben. Andernfalls erhalten Sie eine Fehlermeldung.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

NetApp

Der Name des Add-ons lautet `netapp_trident-operator` und der Namespace ist`trident`. NetApp veröffentlicht das Add-on.

Informationen zum Add-on finden Sie in der NetApp Dokumentation unter Konfiguration des Trident EKS-Add-ons.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

New Relic

Der Add-On-Name lautet `new-relic_kubernetes-operator` und der Namespace ist `newrelic`. New Relic veröffentlicht das Add-On.

Informationen zum Add-on finden Sie unter Installieren des New-Relic-Add-Ons für EKS in der New-Relic-Dokumentation.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Rafay

Der Add-On-Name lautet `rafay-systems_rafay-operator` und der Namespace ist `rafay-system`. Rafay veröffentlicht das Add-On.

Informationen zum Add-On finden Sie unter Installieren des Rafay-Amazon-EKS-Add-Ons in der Rafay-Dokumentation.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Rad Security

- Publisher – RAD SECURITY
- Name (Name – `rad-security_rad-security`
- Namespace – `ksoc`
- Servicekonto-Name – Ein Servicekonto wird mit diesem Add-On nicht verwendet.
- Von AWS verwaltete IAM-Richtlinie – Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.
- Benutzerdefinierte IAM-Berechtigungen – Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.
- Anweisungen zur Einrichtung und Verwendung — Weitere Informationen finden Sie unter Installation von Rad Through The AWS Marketplace in der Rad Security-Dokumentation.

SolarWinds

- Publisher – SOLARWINDS
- Name (Name – `solarwinds_swo-k8s-collector-addon`
- Namespace – `solarwinds`
- Servicekonto-Name – Ein Servicekonto wird mit diesem Add-On nicht verwendet.
- Von AWS verwaltete IAM-Richtlinie – Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.
- Benutzerdefinierte IAM-Berechtigungen – Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.
- Anweisungen zur Einrichtung und Verwendung — siehe Überwachen eines Amazon EKS-Clusters in der SolarWinds Dokumentation.

Solo

Der Add-On-Name lautet `solo-io_istio-distro` und der Namespace ist `istio-system`. Solo veröffentlicht das Add-On.

Informationen zum Add-On finden Sie unter Installation von Istio in der Solo.io-Dokumentation.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Snyk

- Publisher – SNYK
- Name (Name – `snyk_runtime-sensor`
- Namespace – `snyk_runtime-sensor`
- Servicekonto-Name – Ein Servicekonto wird mit diesem Add-On nicht verwendet.
- Von AWS verwaltete IAM-Richtlinie – Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.
- Benutzerdefinierte IAM-Berechtigungen – Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.
- Anleitung zur Einrichtung und Verwendung – Beachten Sie den Snyk-Laufzeitsensor in der Snyk-Benutzerdokumentation.

Stormforge

Der Add-On-Name lautet `stormforge_optimize-Live` und der Namespace ist `stormforge-system`. Stormforge veröffentlicht das Add-On.

Informationen zum Add-on finden Sie in der StormForge Dokumentation unter Installation des StormForge Agenten.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

SUSE

- Publisher – SUSE
- Name (Name – `suse_observability-agent`
- Namespace – `suse-observability`
- Servicekonto-Name – Ein Servicekonto wird mit diesem Add-On nicht verwendet.
- Von AWS verwaltete IAM-Richtlinie – Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.
- Benutzerdefinierte IAM-Berechtigungen – Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.
- Anweisungen zur Einrichtung und Verwendung – Lesen Sie Schnellstart in der SUSE-Dokumentation.

Splunk

Der Add-On-Name lautet `splunk_splunk-otel-collector-chart` und der Namespace ist `splunk-monitoring`. Splunk veröffentlicht das Add-On.

Informationen zum Add-On finden Sie unter Installieren des Splunk-Add-Ons für Amazon EKS in der Splunk-Dokumentation

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Teleport

Der Add-On-Name lautet `teleport_teleport` und der Namespace ist `teleport`. Teleport veröffentlicht das Add-On.

Informationen zum Add-On finden Sie unter Funktionsweise von Teleport in der Teleport-Dokumentation.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Tetrate

Der Add-On-Name lautet `tetrate-io_istio-distro` und der Namespace ist `istio-system`. Tetrate Io veröffentlicht das Add-On.

Informationen zum Add-On finden Sie auf der Website von Tetrate Istio Distro.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Upbound Universal Crossplane

Der Add-On-Name lautet `upbound_universal-crossplane` und der Namespace ist `upbound-system`. Upbound veröffentlicht das Add-On.

Informationen zum Add-On finden Sie unter Upbound Universal Crossplane (UXP) in der Upbound-Dokumentation

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Upwind

Der Add-On-Name lautet `upwind` und der Namespace ist `upwind`. Upwind veröffentlicht das Add-On.

Informationen zum Add-On finden Sie in der Upwind-Dokumentation.

Servicekonto-Name

Ein Servicekonto wird mit diesem Add-On nicht verwendet.

AWS verwaltete IAM-Richtlinie

Eine verwaltete Richtlinie wird mit diesem Add-On nicht verwendet.

Benutzerdefinierte IAM-Berechtigungen

Benutzerdefinierte Berechtigungen werden mit diesem Add-On nicht verwendet.

Erstellung eines Amazon-EKS-Add-Ons

Amazon-EKS-Add-Ons sind Zusatzsoftware für Amazon-EKS-Cluster. Alle Amazon-EKS-Add-Ons:

- Sie enthalten die neuesten Sicherheits-Patches und Fehlerbehebungen.
- Sind für die Verwendung AWS mit Amazon EKS validiert.
- Sie reduzieren den Arbeitsaufwand für die Verwaltung der Add-On-Software.

Sie können ein Amazon EKS-Add-on mithilfe `eksctl` der AWS-Managementkonsole oder der AWS CLI erstellen. Wenn das Add-On eine IAM-Rolle erfordert, finden Sie weitere Informationen zum Erstellen der Rolle in den Details zum jeweiligen Add-On unter Amazon-EKS-Add-Ons.

Voraussetzungen

Führen Sie Folgendes aus, bevor Sie ein Add-On erstellen:

- Der Cluster muss vorhanden sein, bevor Sie ein Add-On dafür erstellen. Weitere Informationen finden Sie unter the section called “Erstellen eines Clusters”.
- Überprüfen Sie, ob Ihr Add-On eine IAM-Rolle erfordert. Weitere Informationen finden Sie unter the section called “Kompatibilität überprüfen”.
- Stellen Sie sicher, dass die Amazon-EKS-Add-On-Version mit Ihrem Cluster kompatibel ist. Weitere Informationen finden Sie unter the section called “Kompatibilität überprüfen”.
- Vergewissern Sie sich, dass Version 0.190.0 oder höher des `eksctl` Befehlszeilentools auf Ihrem Computer installiert ist oder. AWS CloudShell Weitere Informationen finden Sie unter Installation auf der `eksctl`-Website.

Verfahren

Sie können ein Amazon EKS-Add-on mithilfe `eksctl` der AWS-Managementkonsole oder der AWS CLI erstellen. Wenn das Add-On eine IAM-Rolle erfordert, finden Sie in den Details zum jeweiligen Add-On unter Verfügbare Amazon-EKS-Add-Ons von AWS Einzelheiten zum Erstellen der Rolle.

Add-On erstellen (eksctl)

1. Zeigen Sie die Namen der Add-Ons an, die für eine Cluster-Version verfügbar sind. Ersetzen Sie *`1.33`* durch die Version Ihres Clusters.

   ```
   eksctl utils describe-addon-versions --kubernetes-version 1.33 | grep AddonName
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   "AddonName": "aws-ebs-csi-driver",
                           "AddonName": "coredns",
                           "AddonName": "kube-proxy",
                           "AddonName": "vpc-cni",
                           "AddonName": "adot",
                           "AddonName": "dynatrace_dynatrace-operator",
   ```

```
                        "AddonName": "upbound_universal-crossplane",
                        "AddonName": "teleport_teleport",
                        "AddonName": "factorhouse_kpow",
                        [...]
```

2. Zeigen Sie die Versionen des Add-Ons an, die für das Add-On verfügbar sind, das Sie erstellen möchten. Ersetzen Sie *1.33* durch die Version Ihres Clusters. Ersetzen Sie *name-of-addon* mit dem Namen des Add-Ons, für das Sie die Versionen anzeigen möchten. Der Name muss einer der Namen sein, die in den vorherigen Schritten zurückgegeben wurden.

```
eksctl utils describe-addon-versions --kubernetes-version 1.33 --name name-of-addon |
 grep AddonVersion
```

   Die folgende Ausgabe ist ein Beispiel dafür, was für das Add-On mit dem Namen `vpc-cni` zurückgegeben wird. Sie können sehen, dass das Add-On in mehreren Versionen verfügbar ist.

```
"AddonVersions": [
    "AddonVersion": "v1.12.0-eksbuild.1",
    "AddonVersion": "v1.11.4-eksbuild.1",
    "AddonVersion": "v1.10.4-eksbuild.1",
    "AddonVersion": "v1.9.3-eksbuild.1",
```

   a. Stellen Sie fest, ob es sich bei dem Add-on, das Sie erstellen möchten, um ein Amazon EKS- oder AWS Marketplace-Add-On handelt. Der AWS Marketplace enthält Add-Ons von Drittanbietern, für die Sie zusätzliche Schritte ausführen müssen, um das Add-on zu erstellen.

```
eksctl utils describe-addon-versions --kubernetes-version 1.33 --name name-of-
addon | grep ProductUrl
```

      Wenn keine Ausgabe zurückgegeben wird, handelt es sich bei dem Add-On um ein Amazon EKS. Wenn die Ausgabe zurückgegeben wird, handelt es sich bei dem Add-On um ein AWS Marketplace-Add-On. Die folgende Ausgabe bezieht sich auf ein Add-On mit dem Namen `teleport_teleport`.

```
"ProductUrl": "https://aws.amazon.com/marketplace/pp?sku=3bda70bb-566f-4976-806c-
f96faef18b26"
```

      Sie können mehr über das Add-on im AWS Marketplace mit der zurückgegebenen URL erfahren. Wenn für das Add-on ein Abonnement erforderlich ist, können Sie das Add-on über

den AWS Marketplace abonnieren. Wenn Sie ein Add-on im AWS Marketplace erstellen möchten, muss der IAM-Principal, den Sie zum Erstellen des Add-ons verwenden, über die Berechtigung verfügen, die AWSServiceRoleForAWSLicenseManagerRoleserviceverknüpfte Rolle zu erstellen. Weitere Informationen zum Zuweisen von Berechtigungen zu einer IAM-Entität finden Sie unter Hinzufügen und Entfernen von IAM-Identitäts-Berechtigungen im IAM-Benutzerhandbuch.

3. Erstellen Sie ein Amazon-EKS-Add-On. Kopieren Sie den Befehl und ersetzen Sie ihn *user-data* wie folgt:
   - Ersetzen Sie *my-cluster* mit dem Namen Ihres Clusters.
   - Ersetzen Sie *name-of-addon* durch den Namen des Add-Ons, das Sie erstellen möchten.
   - Wenn Sie eine Version des Add-ons verwenden möchten, die älter als die neueste Version ist, *latest* ersetzen Sie sie durch die Versionsnummer, die in der Ausgabe eines vorherigen Schritts zurückgegeben wurde, den Sie verwenden möchten.
   - Wenn das Add-On eine Servicekonto-Rolle verwendet, ersetzen Sie *111122223333* durch Ihre Konto-ID und *role-name* durch den Namen der Rolle. Anweisungen zum Erstellen einer Rolle für Ihr Servicekonto finden Sie in der Dokumentation für das Add-On, das Sie erstellen. Eine Liste der Add-Ons finden Sie unter the section called “ AWS Add-Ons”. Die Angabe einer Servicekonto-Rolle setzt voraus, dass Sie über einen IAM OpenID Connect (OIDC)-Anbieter für Ihren Cluster verfügen. Um festzustellen, ob Sie über einen solchen für Ihren Cluster verfügen, oder um einen zu erstellen, lesen Sie the section called “IAM-OIDC-Anbieter”.

     Wenn das Add-On keine Servicekonto-Rolle verwendet, löschen Sie `--service-account-role-arn arn:aws: iam::111122223333:role/role-name`.
   - Dieser Beispielbefehl überschreibt die Konfiguration aller vorhandenen selbstverwalteten Versionen des Add-Ons, falls es eine gibt. Wenn Sie die Konfiguration eines vorhandenen selbstverwalteten Add-Ons nicht überschreiben möchten, entfernen Sie die *--force* Option. Wenn Sie die Option entfernen und das Amazon-EKS-Add-On die Konfiguration eines vorhandenen selbstverwalteten Add-Ons benötigt, scheitert die Erstellung des Amazon-EKS-Add-Ons mit einer Fehlermeldung, die Sie bei der Behebung des Konflikts unterstützt. Stellen Sie vor der Angabe dieser Option sicher, dass das Amazon-EKS-Add-On keine Einstellungen verwaltet, die Sie verwalten müssen, da diese Einstellungen mit dieser Option überschrieben werden.

```
eksctl create addon --cluster my-cluster --name name-of-addon --version latest \
    --service-account-role-arn arn:aws: iam::111122223333:role/role-name --force
```

Sie finden eine Liste mit allen verfügbaren Optionen für den Befehl.

```
eksctl create addon --help
```

Weitere Informationen zu verfügbaren Optionen finden Sie unter Add-Ons in der `eksctl`-Dokumentation.

Add-on erstellen (Konsole)AWS

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich Cluster aus.
3. Wählen Sie den Namen des Clusters, für den Sie das Add-On erstellen möchten.
4. Wählen Sie die Registerkarte Add-ons.
5. Wählen Sie Weitere Add-Ons erhalten.
6. Wählen Sie auf der Seite Select add-ons (Add-Ons auswählen) die Add-Ons aus, die Sie Ihrem Cluster hinzufügen möchten. Sie können so viele Amazon EKS-Add-Ons und AWS Marketplace-Add-Ons hinzufügen, wie Sie benötigen.

   Für AWS Marketplace-Add-Ons muss der IAM-Principal, den Sie zum Erstellen des Add-ons verwenden, über Berechtigungen zum Lesen von Berechtigungen für das Add-on aus dem verfügen. AWS LicenseManager AWS LicenseManager erfordert eine AWSServiceRoleForAWSLicenseManagerRoleserviceverknüpfte Rolle (SLR), die es AWS Ressourcen ermöglicht, Lizenzen in Ihrem Namen zu verwalten. Das SLR ist eine einmalige Anforderung pro Konto, und Sie müssen keine separaten SLRs für jedes Add-On oder jeden Cluster erstellen. Weitere Informationen zum Zuweisen von Berechtigungen zu einem IAM-Prinzipal finden Sie unter Hinzufügen und Entfernen von IAM-Identitäts-Berechtigungen im IAM-Benutzerhandbuch.

   Wenn die AWS Marketplace-Add-Ons, die Sie installieren möchten, nicht aufgeführt sind, können Sie auf die Seitennummerierung klicken, um zusätzliche Seitenergebnisse anzuzeigen, oder im Suchfeld suchen. In den Filtering options (Filteroptionen) können Sie auch nach category (Kategorie), vendor (Anbieter) oder pricing model (Preismodell) filtern und dann die Add-ons aus den Suchergebnissen auswählen. Nachdem Sie die Add-Ons ausgewählt haben, die Sie installieren möchten, wählen Sie Weiter.
7. Gehen Sie auf der Seite Konfigurieren ausgewählter Add-Ons-Einstellungen wie folgt vor:

a. Wählen Sie View subscription options (Abonnementoptionen anzeigen), um das Formular Subscription options (Abonnementoptionen) zu öffnen. Lesen Sie die Abschnitte Pricing details (Preisinformationen) und Legal (Rechtliche Hinweise) und klicken Sie dann auf Subscribe (Abonnieren), um fortzufahren.

b. Wählen Sie für Version die Version aus, welche Sie installieren möchten. Wir empfehlen die als neueste Version markierte Version, es sei denn, das einzelne Add-On, das Sie erstellen, empfiehlt eine andere Version. Um festzustellen, ob ein Add-On über eine empfohlene Version verfügt, lesen Sie die Dokumentation für das Add-On, das Sie erstellen. Eine Liste der Add-Ons finden Sie unter the section called “ AWS Add-Ons”.

c. Sie haben zwei Möglichkeiten, Rollen für Add-Ons zu konfigurieren: IAM-Rolle für EKS-Pod-Identitäten und IAM-Rollen für Servicekonten (IRSA). Befolgen Sie die entsprechenden nachfolgenden Schritte für die von Ihnen bevorzugte Option. Wenn für alle von Ihnen ausgewählten Add-Ons Abonnement erforderlich ist unter Status, wählen Sie Weiter aus. Sie können diese Add-Ons konfigurieren, wenn Sie sie nach der Erstellung Ihres Clusters abonniert haben. Für die Add-Ons, die kein Abonnement erforderlich unter Status haben, gehen Sie wie folgt vor:

   i. Für die IAM-Rolle Pod Identity für Servicekonto können Sie entweder eine vorhandene IAM-Rolle EKS Pod Identity verwenden oder mithilfe der Schaltfläche Empfohlene Rolle erstellen eine neue Rolle erstellen. In diesem Feld werden nur Optionen mit der entsprechenden Vertrauensrichtlinie bereitgestellt. Wenn keine Rolle zur Auswahl steht, verfügen Sie über keine vorhandene Rolle mit einer passenden Vertrauensrichtlinie. Um eine IAM-Rolle EKS Pod Identity für Servicekonten der ausgewählten Add-On zu konfigurieren, wählen Sie Empfohlene Rolle erstellen aus. Der Assistent zur Rollenerstellung wird in einem separaten Fenster geöffnet. Der Assistent füllt die Rolleninformationen automatisch wie folgt aus. Führen Sie für jedes Add-On, für das Sie die IAM-Rolle EKS Pod Identity erstellen möchten, die folgenden Schritte im IAM-Assistenten aus.

      - Im Schritt Vertrauenswürdige Entität auswählen sind die AWS Serviceoption für EKS und der Anwendungsfall für EKS — Pod Identity vorausgewählt, und die entsprechende Vertrauensrichtlinie wird automatisch für das Add-on ausgefüllt. Beispielsweise wird die Rolle mit der entsprechenden Vertrauensrichtlinie erstellt, die den IAM-Prinzipal pods.eks.amazonaws.com enthält, wie in the section called “Vorteile von EKS-Pod-Identitäten” beschrieben. Wählen Sie Weiter aus.
      - Im Schritt Berechtigungen hinzufügen wird die entsprechende verwaltete Richtlinie für die Rollenrichtlinie für das Add-On vorausgewählt. Beispielsweise wird für das Add-On für Amazon VPC CNI die Rolle mit der verwalteten Richtlinie `AmazonEKS_CNI_Policy`

erstellt, wie in the section called “Amazon-VPC-CNI-Plugin für Kubernetes” beschrieben. Wählen Sie Weiter aus.

- Im Schritt Name, überprüfen und erstellen wird unter Rollenname automatisch der Standardrollenname für das Add-On eingetragen. Für das Amazon VPC CNI-Add-on wird die Rolle beispielsweise mit dem Namen Amazon erstellt. EKSPod IdentityAmazon VPCCNIRole Unter Beschreibung wird die Standardbeschreibung automatisch mit der entsprechenden Beschreibung für das Add-On eingetragen. Für das Amazon VPC CNI-Add-on wird die Rolle beispielsweise mit der Beschreibung Erlaubt Pods, die im Amazon EKS-Cluster ausgeführt werden, den Zugriff auf AWS Ressourcen erstellt. Zeigen Sie unter Vertrauensrichtlinie die ausgefüllte Vertrauensrichtlinie für das Add-On an. Wählen Sie Rolle erstellen aus.

  HINWEIS: Durch Beibehalten des Standard-Rollennamens kann EKS die Rolle für Add-Ons in neuen Clustern oder beim Hinzufügen von Add-Ons zu vorhandenen Clustern vorab auswählen. Sie können diesen Namen weiterhin überschreiben, und die Rolle steht dann für das Add-On in Ihren Clustern zur Verfügung, muss jedoch manuell aus der Dropdown-Liste ausgewählt werden.

ii. Informationen zu Add-Ons, bei denen unter Status nicht die Option Abonnement erforderlich angezeigt wird und bei denen Sie Rollen mithilfe von IRSA konfigurieren möchten, finden Sie in der Dokumentation des Add-Ons, das Sie erstellen, um eine IAM-Richtlinie zu erstellen und sie einer Rolle zuzuordnen. Eine Liste der Add-Ons finden Sie unter the section called “ AWS Add-Ons”. Die Auswahl einer IAM-Rolle erfordert, dass Sie über einen IAM OpenID Connect (OIDC)-Anbieter für Ihren Cluster verfügen. Um festzustellen, ob Sie über einen solchen für Ihren Cluster verfügen, oder um einen zu erstellen, lesen Sie the section called “IAM-OIDC-Anbieter”.

iii. Wählen Sie Optional configuration settings (Optionale Konfigurationseinstellungen) aus.

iv. Wenn für das Add-On eine Konfiguration erforderlich ist, geben Sie diese in das Feld Configuration values (Konfigurationswerte) ein. Um festzustellen, ob für das Add-On Konfigurationsinformationen erforderlich sind, lesen Sie die Dokumentation für das Add-On, das Sie erstellen. Eine Liste der Add-Ons finden Sie unter the section called “ AWS Add-Ons”.

v. Wählen Sie eine der verfügbaren Optionen für die Konfliktlösungsmethode aus. Wenn Sie Überschreiben für Konfliktlösungsmethode auswählen, können eine oder mehrere der Einstellungen für das vorhandene Add-On mit den Einstellungen des Amazon-EKS-Add-Ons überschrieben werden. Wenn Sie diese Option nicht aktivieren und ein Konflikt mit Ihren vorhandenen Einstellungen vorliegt, schlägt der Vorgang fehl. Sie können die sich daraus

ergebende Fehlermeldung heranziehen, um den Konflikt zu beheben. Stellen Sie vor der Auswahl dieser Option sicher, dass das Amazon-EKS-Add-On keine Einstellungen verwaltet, die Sie selbst verwalten müssen.

vi. Wenn Sie das Add-On in einem bestimmten Namespace installieren möchten, geben Sie diesen in das Feld Namespace ein. Für Add-Ons AWS und Community-Erweiterungen können Sie einen benutzerdefinierten Kubernetes-Namespace definieren, in dem das Add-on installiert werden soll. Weitere Informationen finden Sie unter the section called “Benutzerdefinierter Namespace für Add-Ons”.

vii. Wählen Sie Weiter aus.

8. Wählen Sie auf der Seite Überprüfen und hinzufügen die Option Erstellen aus. Nachdem die Installation der Add-Ons abgeschlossen ist, werden Ihre installierten Add-Ons angezeigt.
9. Wenn für eines der von Ihnen installierten Add-Ons ein Abonnement erforderlich ist, gehen Sie wie folgt vor:
   a. Wählen Sie die Schaltfläche Subscribe (Abonnieren) in der unteren rechten Ecke für das Add-On. Sie werden auf die Seite für das Add-on im AWS Marketplace weitergeleitet. Lesen Sie die Informationen zum Add-On, z. B. die Product Overview (Produktübersicht) und die Pricing Information (Preisinformationen).
   b. Wählen Sie oben rechts auf der Add-on-Seite die Schaltfläche Continue to Subscribe (Mit dem Abonnement fortfahren) aus.
   c. Lesen Sie sich die (Allgemeinen Geschäftsbedingungen) durch. Wenn Sie damit einverstanden sind, wählen Sie Accept Terms (Bedingungen akzeptieren) aus. Die Bearbeitung des Abonnements kann mehrere Minuten dauern. Während das Abonnement bearbeitet wird, ist die Schaltfläche Return to Amazon EKS Console (Zurück zur Amazon-EKS-Konsole) ausgegraut.
   d. Sobald die Bearbeitung des Abonnements abgeschlossen ist, ist die Schaltfläche Return to Amazon EKS Console (Zurück zur Amazon-EKS-Konsole) nicht mehr ausgegraut. Wählen Sie die Schaltfläche aus, um zur Registerkarte Add-ons (Add-Ons) der Amazon-EKS-Konsole für Ihren Cluster zurückzukehren.
   e. Wählen Sie für das Add-On, das Sie abonniert haben, Remove and reinstall (Entfernen und erneut installieren) und dann Reinstall add-on (Add-On erneut installieren) aus. Die Installation des Add-Ons kann einige Minuten dauern. Wenn die Installation abgeschlossen ist, können Sie das Add-On konfigurieren.

Add-on erstellen (AWS CLI)

1. Sie müssen Version `2.12.3` oder höher oder Version `1.27.160` oder höher der AWS Befehlszeilenschnittstelle (AWS CLI) auf Ihrem Gerät installiert und konfiguriert haben oder AWS CloudShell. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Paketmanager wie `yum` `apt-get`, oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der AWS CLI. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Die AWS CLI-Version, in der installiert ist, AWS CloudShell kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zur Aktualisierung finden Sie im AWS CloudShell Benutzerhandbuch unter AWS CLI in Ihrem Home-Verzeichnis installieren.
2. Bestimmen Sie, welche Add-Ons verfügbar sind. Sie können alle verfügbaren Add-Ons, ihren Typ und ihren Publisher anzeigen. Sie können auch die URL für Add-Ons sehen, die über den AWS Marketplace erhältlich sind. Ersetzen Sie *`1.33`* durch die Version Ihres Clusters.

```
aws eks describe-addon-versions --kubernetes-version 1.33 \
    --query 'addons[].{MarketplaceProductUrl: marketplaceInformation.productUrl,
 Name: addonName, Owner: owner Publisher: publisher, Type: type}' --output table
```

Eine Beispielausgabe sieht wie folgt aus.

```
-------------------------------------------------------------------------------------------------
|
 DescribeAddonVersions
  |
+-----------------------------------------------------------
+-------------------------------+------------------+--------------
+---------------------+
|                     MarketplaceProductUrl                     |             Name
           |      Owner       |  Publisher   |        Type         |
+-----------------------------------------------------------
+-------------------------------+------------------+--------------
+---------------------+
|  None                                                         |  aws-ebs-csi-driver
           |  aws             |  eks         |  storage            |
|  None                                                         |  coredns
           |  aws             |  eks         |  networking         |
|  None                                                         |  kube-proxy
           |  aws             |  eks         |  networking         |
```

```
|  None                                                                 |  vpc-cni
           |  aws             |  eks         |  networking          |
|  None                                                                 |  adot
           |  aws             |  eks         |  observability       |
| https://aws.amazon.com/marketplace/pp/prodview-brb73nceicv7u |
 dynatrace_dynatrace-operator |  aws-marketplace |  dynatrace    |  monitoring
 |
| https://aws.amazon.com/marketplace/pp/prodview-uhc2iwi5xysoc |  upbound_universal-
crossplane |  aws-marketplace |  upbound     |  infra-management    |
| https://aws.amazon.com/marketplace/pp/prodview-hd2ydsrgqy4li |  teleport_teleport
          |  aws-marketplace |  teleport    |  policy-management   |
| https://aws.amazon.com/marketplace/pp/prodview-vgghgqdsplhvc |  factorhouse_kpow
          |  aws-marketplace |  factorhouse |  monitoring          |
|  [...]                                                                |  [...]
           |  [...]           |  [...]       |  [...]               |
+-----------------------------------------------------------------
+------------------------------+------------------+--------------
+---------------------+
```

Ihre Ausgabe ist möglicherweise anders. In dieser Beispielausgabe sind drei verschiedene Add-Ons vom Typ `networking` und fünf Add-Ons mit einem Publisher vom Typ `eks` verfügbar. Für die Add-Ons mit `aws-marketplace` in der `Owner`-Spalte ist möglicherweise ein Abonnement erforderlich, bevor Sie sie installieren können. Sie können die URL besuchen, um mehr über das Add-On zu erfahren und es zu abonnieren.

3. Sie können anzeigen, welche Versionen für jedes Add-On verfügbar sind. Ersetzen Sie *`1.33`* durch die Version Ihres Clusters und *`vpc-cni`* durch den Namen eines Add-Ons, das im vorherigen Schritt zurückgegeben wurde.

```
aws eks describe-addon-versions --kubernetes-version 1.33 --addon-name vpc-cni \
    --query 'addons[].addonVersions[].{Version: addonVersion, Defaultversion:
 compatibilities[0].defaultVersion}' --output table
```

Eine Beispielausgabe sieht wie folgt aus.

```
-----------------------------------------
|          DescribeAddonVersions         |
+-----------------+----------------------+
| Defaultversion  |       Version        |
+-----------------+----------------------+
|  False          |  v1.12.0-eksbuild.1  |
|  True           |  v1.11.4-eksbuild.1  |
```

```
|  False          |  v1.10.4-eksbuild.1  |
|  False          |  v1.9.3-eksbuild.1   |
+-----------------+----------------------+
```

Die Version `True` in der `Defaultversion`-Spalte ist standardmäßig die Version, mit der das Add-On erstellt wurde.

4. (Optional) Finden Sie die Konfigurationsoptionen für das von Ihnen gewählte Add-On, indem Sie den folgenden Befehl ausführen:

```
aws eks describe-addon-configuration --addon-name vpc-cni --addon-version v1.12.0-
eksbuild.1
```

```
{
    "addonName": "vpc-cni",
    "addonVersion": "v1.12.0-eksbuild.1",
    "configurationSchema": "{\"$ref\":\"#/definitions/VpcCni\",\"$schema\":\"http://
json-schema.org/draft-06/schema#\",\"definitions\":{\"Cri\":{\"additionalProperties
\":false,\"properties\":{\"hostPath\":{\"$ref\":\"#/definitions/HostPath\"}},\"title
\":\"Cri\",\"type\":\"object\"},\"Env\":{\"additionalProperties\":false,\"properties
\":{\"ADDITIONAL_ENI_TAGS\":{\"type\":\"string\"},\"AWS_VPC_CNI_NODE_PORT_SUPPORT
\":{\"format\":\"boolean\",\"type\":\"string\"},\"AWS_VPC_ENI_MTU\":{\"format\":
\"integer\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_CONFIGURE_RPFILTER\":{\"format
\":\"boolean\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG\":{\"format
\":\"boolean\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_EXTERNALSNAT\":{\"format\":
\"boolean\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_LOGLEVEL\":{\"type\":\"string\"},
\"AWS_VPC_K8S_CNI_LOG_FILE\":{\"type\":\"string\"},\"AWS_VPC_K8S_CNI_RANDOMIZESNAT
\":{\"type\":\"string\"},\"AWS_VPC_K8S_CNI_VETHPREFIX\":{\"type\":\"string\"},
\"AWS_VPC_K8S_PLUGIN_LOG_FILE\":{\"type\":\"string\"},\"AWS_VPC_K8S_PLUGIN_LOG_LEVEL
\":{\"type\":\"string\"},\"DISABLE_INTROSPECTION\":{\"format\":\"boolean\",\"type
\":\"string\"},\"DISABLE_METRICS\":{\"format\":\"boolean\",\"type\":\"string
\"},\"DISABLE_NETWORK_RESOURCE_PROVISIONING\":{\"format\":\"boolean\",\"type
\":\"string\"},\"ENABLE_POD_ENI\":{\"format\":\"boolean\",\"type\":\"string
\"},\"ENABLE_PREFIX_DELEGATION\":{\"format\":\"boolean\",\"type\":\"string\"},
\"WARM_ENI_TARGET\":{\"format\":\"integer\",\"type\":\"string\"},\"WARM_PREFIX_TARGET
\":{\"format\":\"integer\",\"type\":\"string\"}},\"title\":\"Env\",\"type\":
\"object\"},\"HostPath\":{\"additionalProperties\":false,\"properties\":{\"path
\":{\"type\":\"string\"}},\"title\":\"HostPath\",\"type\":\"object\"},\"Limits
\":{\"additionalProperties\":false,\"properties\":{\"cpu\":{\"type\":\"string
\"},\"memory\":{\"type\":\"string\"}},\"title\":\"Limits\",\"type\":\"object\"},
\"Resources\":{\"additionalProperties\":false,\"properties\":{\"limits\":{\"$ref\":
\"#/definitions/Limits\"},\"requests\":{\"$ref\":\"#/definitions/Limits\"}},\"title
```

```
\":\"Resources\",\"type\":\"object\"},\"VpcCni\":{\"additionalProperties\":false,
\"properties\":{\"cri\":{\"$ref\":\"#/definitions/Cri\"},\"env\":{\"$ref\":\"#/
definitions/Env\"},\"resources\":{\"$ref\":\"#/definitions/Resources\"}},\"title\":
\"VpcCni\",\"type\":\"object\"}}}"
}
```

Die Ausgabe ist ein standardmäßiges JSON-Schema.

Hier ist ein Beispiel für gültige Konfigurationswerte im JSON-Format, die mit dem obigen Schema funktionieren.

```
{
  "resources": {
    "limits": {
      "cpu": "100m"
    }
  }
}
```

Hier ist ein Beispiel für gültige Konfigurationswerte im YAML-Format, die mit dem obigen Schema funktionieren.

```
  resources:
    limits:
      cpu: 100m
```

5. Stellen Sie fest, ob das Add-On IAM-Berechtigungen erfordert. In diesem Fall müssen Sie (1) festlegen, ob Sie EKS Pod Identity oder IAM-Rollen für Servicekonten (IRSA) verwenden möchten, (2) die ARN der IAM-Rolle bestimmen, die mit dem Add-On verwendet werden soll, und (3) den Namen des Kubernetes-Servicekontos festlegen, das vom Add-On verwendet wird. Weitere Informationen finden Sie unter the section called “IAM-Informationen abrufen”.
   - Amazon EKS empfiehlt die Verwendung von EKS Pod Identities, sofern das Add-On dies unterstützt. Dies erfordert, dass der Pod Identity Agent auf Ihrem Cluster installiert ist. Weitere Informationen zur Verwendung von Pod Identities mit Add-Ons finden Sie unter the section called “IAM roles”.
   - Wenn das Add-On oder Ihr Cluster nicht für EKS Pod Identities eingerichtet ist, verwenden Sie IRSA. Bestätigen Sie, dass IRSA auf Ihrem Cluster eingerichtet ist.
   - Überprüfen Sie die Dokumentation zu Amazon-EKS-Add-Ons, um festzustellen, ob das Add-On IAM-Berechtigungen und den Namen des zugehörigen Kubernetes-Servicekontos erfordert.

a. Erstellen Sie ein Amazon-EKS-Add-On. Kopieren Sie den folgenden Befehl auf Ihr Gerät. Nehmen Sie nach Bedarf die folgenden Änderungen am Befehl vor und führen Sie anschließend den geänderten Befehl aus:

- Ersetzen Sie *my-cluster* mit dem Namen Ihres Clusters.
- Ersetzen Sie *vpc-cni* durch einen Add-On-Namen, der in der Ausgabe des vorherigen Schritts zurückgegeben wurde, den Sie erstellen möchten.
- Ersetzen Sie *version-number* durch die in der Ausgabe des vorherigen Schritts zurückgegebene Version, die Sie verwenden möchten.
- Wenn Sie das Add-On in einem benutzerdefinierten Kubernetes-Namespace installieren möchten, fügen Sie die `--namespace-config 'namespace=<my-namespace>`-Option hinzu. Diese Option ist nur für AWS Community-Add-Ons verfügbar. Weitere Informationen finden Sie unter the section called “Benutzerdefinierter Namespace für Add-Ons”.
- Wenn für das Add-on keine IAM-Berechtigungen erforderlich sind, löschen Sie *<service-account-configuration>* es.
- Führen Sie eine der folgenden Aktionen aus:
  - Wenn für das Add-on (1) IAM-Berechtigungen erforderlich sind und (2) Ihr Cluster EKS-Pod-Identitäten verwendet, *<service-account-configuration>* ersetzen Sie es durch die folgende Pod-Identitätszuordnung. Ersetzen Sie es *<service-account-name>* durch den Namen des Dienstkontos, der vom Add-on verwendet wird. Durch *<role-arn>* den ARN einer IAM-Rolle ersetzen. Die Rolle muss über die von EKS Pod Identities geforderte Vertrauensrichtlinie verfügen.

    ```
    --pod-identity-associations 'serviceAccount=<service-account-name>,roleArn=<role-arn>'
    ```

  - Wenn für das Add-on (1) IAM-Berechtigungen erforderlich sind und (2) Ihr Cluster IRSA verwendet, *<service-account-configuration>* ersetzen Sie es durch die folgende IRSA-Konfiguration. *111122223333*Ersetzen Sie es durch Ihre Konto-ID und *role-name* den Namen einer vorhandenen IAM-Rolle, die Sie erstellt haben. Anweisungen zum Erstellen der Rolle finden Sie in der Dokumentation für das Add-On, das Sie erstellen. Eine Liste der Add-Ons finden Sie unter the section called “ AWS Add-Ons”. Die Angabe einer Servicekonto-Rolle setzt voraus, dass Sie über einen IAM OpenID Connect (OIDC)-Anbieter für Ihren Cluster verfügen. Um festzustellen, ob Sie über einen solchen für Ihren Cluster verfügen, oder um einen zu erstellen, lesen Sie the section called “IAM-OIDC-Anbieter”.

```
--service-account-role-arn arn:aws::iam::111122223333:role/role-name
```

- Diese Beispielbefehle überschreiben die Option `--configuration-values` von gegebenenfalls vorhandenen selbstverwalteten Versionen des Add-Ons. Ersetzen Sie diese durch die gewünschten Konfigurationswerte, z. B. eine Zeichenfolge oder eine Dateieingabe. Wenn Sie keine Konfigurationswerte angeben möchten, löschen Sie die `--configuration-values`-Option. Wenn Sie nicht möchten, dass die AWS CLI die Konfiguration eines vorhandenen selbstverwalteten Add-Ons überschreibt, entfernen Sie die *`--resolve-conflicts OVERWRITE`* Option. Wenn Sie die Option entfernen und das Amazon-EKS-Add-On die Konfiguration eines vorhandenen selbstverwalteten Add-Ons benötigt, scheitert die Erstellung des Amazon-EKS-Add-Ons mit einer Fehlermeldung, die Sie bei der Behebung des Konflikts unterstützt. Stellen Sie vor der Angabe dieser Option sicher, dass das Amazon-EKS-Add-On keine Einstellungen verwaltet, die Sie verwalten müssen, da diese Einstellungen mit dieser Option überschrieben werden.

```
aws eks create-addon --cluster-name my-cluster --addon-name vpc-cni --addon-version
 version-number \
    <service-account-configuration> --configuration-values '{"resources":
{"limits":{"cpu":"100m"}}}' --resolve-conflicts OVERWRITE
```

```
aws eks create-addon --cluster-name my-cluster --addon-name vpc-cni --addon-version
 version-number \
   <service-account-configuration> --configuration-values 'file://example.yaml' --
resolve-conflicts OVERWRITE
```

Eine vollständige Liste der verfügbaren Optionen finden Sie unter `create-addon` in der Amazon-EKS-Befehlszeilenreferenz. Wenn für das von Ihnen erstellte Add-On `aws-marketplace` in der `Owner`-Spalte eines vorherigen Schritts aufgeführt ist, kann die Erstellung fehlschlagen und Sie erhalten eine Fehlermeldung ähnlich der folgenden.

```
{
    "addon": {
        "addonName": "addon-name",
        "clusterName": "my-cluster",
        "status": "CREATE_FAILED",
        "addonVersion": "version",
        "health": {
            "issues": [
```

```
                {
                    "code": "AddonSubscriptionNeeded",
                    "message": "You are currently not subscribed to this add-on. To
 subscribe, visit the AWS Marketplace console, agree to the seller EULA, select the
 pricing type if required, then re-install the add-on"
                }
            ]
        }
    }
}
```

Wenn Sie einen Fehler erhalten, der dem Fehler in der vorherigen Ausgabe ähnelt, besuchen Sie die URL in der Ausgabe eines vorherigen Schritts, um das Add-On zu abonnieren. Führen Sie den `create-addon`-Befehl nach dem Abonnieren erneut aus.

Aktualisierung eines Amazon-EKS-Add-Ons

Amazon EKS aktualisiert das Add-On nicht automatisch, wenn neue Versionen veröffentlicht werden oder nachdem Sie Ihren Cluster auf eine neue Kubernetes-Nebenversion aktualisiert haben. Um ein Add-On für einen vorhandenen Cluster zu aktualisieren, müssen Sie das Update initiieren. Nachdem Sie das Update initiiert haben, aktualisiert Amazon EKS das Add-On für Sie. Lesen Sie vor dem Aktualisieren eines Add-Ons die aktuelle Dokumentation für das Add-On. Eine Liste der verfügbaren Add-Ons finden Sie unter the section called “ AWS Add-Ons”. Wenn das Add-On eine IAM-Rolle erfordert, finden Sie in den Details zum jeweiligen Add-On unter Verfügbare Amazon-EKS-Add-Ons von AWS Einzelheiten zum Erstellen der Rolle.

Voraussetzungen

Führen Sie Folgendes aus, bevor Sie ein Add-On erstellen:

- Überprüfen Sie, ob Ihr Add-On eine IAM-Rolle erfordert. Weitere Informationen finden Sie unter the section called “Amazon-EKS-Add-ons”.
- Stellen Sie sicher, dass die Amazon-EKS-Add-On-Version mit Ihrem Cluster kompatibel ist. Weitere Informationen finden Sie unter the section called “Kompatibilität überprüfen”.

Verfahren

Sie können ein Amazon EKS-Add-on mit `eksctl` AWS-Managementkonsole, der oder der AWS CLI aktualisieren.

Add-On aktualisieren (eksctl)

1. Bestimmen Sie die aktuellen Add-Ons und Add-On-Versionen, die auf Ihrem Cluster installiert sind. Ersetzen Sie *my-cluster* mit dem Namen Ihres Clusters.

   ```
   eksctl get addon --cluster my-cluster
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   NAME        VERSION              STATUS  ISSUES  IAMROLE  UPDATE AVAILABLE
   coredns     v1.8.7-eksbuild.2    ACTIVE  0
   kube-proxy  v1.23.7-eksbuild.1   ACTIVE  0                v1.23.8-eksbuild.2
   vpc-cni     v1.10.4-eksbuild.1   ACTIVE  0                v1.12.0-eksbuild.1,v1.11.4-
   eksbuild.1,v1.11.3-eksbuild.1,v1.11.2-eksbuild.1,v1.11.0-eksbuild.1
   ```

   IIhre Ausgabe sieht möglicherweise anders aus, je nachdem, welche Add-Ons und Versionen Sie auf Ihrem Cluster haben. Sie können sehen, dass in der vorherigen Beispielausgabe für zwei vorhandene Add-Ons auf dem Cluster neuere Versionen in der `UPDATE AVAILABLE`-Spalte verfügbar sind.

2. Aktualisieren Sie das Add-On.

   a. Kopieren Sie den folgenden Befehl auf Ihr Gerät. Nehmen Sie nach Bedarf die folgenden Änderungen am Befehl vor:

   - Ersetzen Sie *my-cluster* mit dem Namen Ihres Clusters.
   - *region-code*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet.
   - Ersetzen Sie *vpc-cni* durch den Namen eines Add-Ons, das in der Ausgabe des vorherigen Schritts zurückgegeben wurde, das Sie aktualisieren möchten.
   - Wenn Sie eine ältere Version als die neueste verfügbare Version aktualisieren möchten, dann ersetzen Sie *latest* durch die Versionsnummer, die in der Ausgabe des vorherigen Schritts zurückgegeben wurde, die Sie verwenden möchten. Für einige Add-Ons gibt es empfohlene Versionen. Weitere Informationen finden Sie in der Dokumentation für das Add-On, das Sie aktualisieren. Eine Liste der Add-Ons finden Sie unterthe section called “ AWS Add-Ons”. * Wenn das Add-on ein Kubernetes-Dienstkonto und eine IAM-Rolle verwendet, *111122223333* ersetzen Sie es durch Ihre Konto-ID und *role-name* durch den Namen einer vorhandenen IAM-Rolle, die Sie erstellt haben. Anweisungen zum Erstellen der Rolle finden Sie in der Dokumentation für das Add-On, das Sie erstellen. Eine Liste der Add-Ons finden Sie unter the section called “ AWS Add-Ons”. Die Angabe einer Servicekonto-Rolle

setzt voraus, dass Sie über einen IAM OpenID Connect (OIDC)-Anbieter für Ihren Cluster verfügen. Um festzustellen, ob Sie über einen solchen für Ihren Cluster verfügen, oder um einen zu erstellen, lesen Sie the section called “IAM-OIDC-Anbieter”.

Wenn das Add-On kein Kubernetes-Servicekonto und keine IAM-Rolle verwendet, löschen Sie die Zeile `serviceAccountRoleARN: arn:aws: iam::`*`111122223333`*`:role/`*`role-name`* .

- Die *`preserve`*-Option behält die vorhandenen Werte für das Add-On bei. Wenn Sie benutzerdefinierte Werte für Add-On-Einstellungen festgelegt haben und diese Option nicht verwenden, überschreibt Amazon EKS Ihre Werte mit seinen Standardwerten. Wenn Sie diese Option verwenden, empfehlen wir, dass Sie alle Feld- und Wertänderungen auf einem Nicht-Produktionscluster testen, bevor Sie das Add-on auf Ihrem Produktionscluster aktualisieren. Wenn Sie diesen Wert auf `overwrite` ändern, werden alle Einstellungen auf die Amazon-EKS-Standardwerte geändert. Wenn Sie benutzerdefinierte Werte für Einstellungen festgelegt haben, werden diese möglicherweise mit den Amazon-EKS-Standardwerten überschrieben. Wenn Sie diesen Wert auf `none` ändern, ändert Amazon EKS den Wert der Einstellungen nicht, aber die Aktualisierung schlägt möglicherweise fehl. Wenn das Update fehlschlägt, erhalten Sie eine Fehlermeldung, die Sie bei der Behebung des Konflikts unterstützt.

```
cat >update-addon.yaml <<EOF
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: my-cluster
  region: region-code

addons:
- name: vpc-cni
  version: latest
  serviceAccountRoleARN: arn:aws: iam::111122223333:role/role-name
  resolveConflicts: preserve
EOF
```

b. Führen Sie den bearbeiteten Befehl aus, um die `update-addon.yaml`-Datei zu erstellen.

c. Wenden Sie die Konfigurationsdatei auf Ihren Cluster an.

```
eksctl update addon -f update-addon.yaml
```

Weitere Informationen zum Aktualisieren von Add-Ons finden Sie unter Aktualisierung von Add-Ons in der `eksctl`-Dokumentation.

Add-on aktualisieren (Konsole)AWS

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich Cluster aus.
3. Wählen Sie den Namen des Clusters, für den Sie das Add-On aktualisieren möchten.
4. Wählen Sie die Registerkarte Add-ons.
5. Wählen Sie das Add-On, das Sie aktualisieren möchten.
6. Wählen Sie Bearbeiten aus.
7. Gehen Sie auf der *`name of addon`* Seite „Konfigurieren“ wie folgt vor:
   a. Wählen Sie die Version aus, die Sie verwenden möchten. Das Add-On verfügt möglicherweise über eine empfohlene Version. Weitere Informationen finden Sie in der Dokumentation für das Add-On, das Sie aktualisieren. Eine Liste der Add-Ons finden Sie unter the section called “ AWS Add-Ons”.
   b. Sie haben zwei Möglichkeiten, Rollen für Add-Ons zu konfigurieren: IAM-Rolle für EKS-Pod-Identitäten und IAM-Rollen für Servicekonten (IRSA). Befolgen Sie die entsprechenden nachfolgenden Schritte für die von Ihnen bevorzugte Option. Wenn für alle von Ihnen ausgewählten Add-Ons Abonnement erforderlich ist unter Status, wählen Sie Weiter aus. Für die Add-Ons, die kein Abonnement erforderlich unter Status haben, gehen Sie wie folgt vor:
      i. Für die IAM-Rolle Pod Identity für Servicekonto können Sie entweder eine vorhandene IAM-Rolle EKS Pod Identity verwenden oder mithilfe der Schaltfläche Empfohlene Rolle erstellen eine neue Rolle erstellen. In diesem Feld werden nur Optionen mit der entsprechenden Vertrauensrichtlinie bereitgestellt. Wenn keine Rolle zur Auswahl steht, verfügen Sie über keine vorhandene Rolle mit einer passenden Vertrauensrichtlinie. Um eine IAM-Rolle EKS Pod Identity für Servicekonten der ausgewählten Add-On zu konfigurieren, wählen Sie Empfohlene Rolle erstellen aus. Der Assistent zur Rollenerstellung wird in einem separaten Fenster geöffnet. Der Assistent füllt die Rolleninformationen automatisch wie folgt aus. Führen Sie für jedes Add-On, für das Sie die IAM-Rolle EKS Pod Identity erstellen möchten, die folgenden Schritte im IAM-Assistenten aus.
         - Im Schritt Vertrauenswürdige Entität auswählen sind die AWS Serviceoption für EKS und der Anwendungsfall für EKS — Pod Identity vorausgewählt, und die entsprechende Vertrauensrichtlinie wird automatisch für das Add-on ausgefüllt. Beispielsweise wird

die Rolle mit der entsprechenden Vertrauensrichtlinie erstellt, die den IAM-Prinzipal pods.eks.amazonaws.com enthält, wie in the section called “Vorteile von EKS-Pod-Identitäten” beschrieben. Wählen Sie Weiter aus.

- Im Schritt Berechtigungen hinzufügen wird die entsprechende verwaltete Richtlinie für die Rollenrichtlinie für das Add-On vorausgewählt. Beispielsweise wird für das Add-On für Amazon VPC CNI die Rolle mit der verwalteten Richtlinie `AmazonEKS_CNI_Policy` erstellt, wie in the section called “Amazon-VPC-CNI-Plugin für Kubernetes” beschrieben. Wählen Sie Weiter aus.
- Im Schritt Name, überprüfen und erstellen wird unter Rollenname automatisch der Standardrollenname für das Add-On eingetragen. Für das Amazon VPC CNI-Add-on wird die Rolle beispielsweise mit dem Namen Amazon erstellt. EKSPod IdentityAmazon VPCCNIRole Unter Beschreibung wird die Standardbeschreibung automatisch mit der entsprechenden Beschreibung für das Add-On eingetragen. Für das Amazon VPC CNI-Add-on wird die Rolle beispielsweise mit der Beschreibung Erlaubt Pods, die im Amazon EKS-Cluster ausgeführt werden, den Zugriff auf AWS Ressourcen erstellt. Zeigen Sie unter Vertrauensrichtlinie die ausgefüllte Vertrauensrichtlinie für das Add-On an. Wählen Sie Rolle erstellen aus.

> **Note**
>
> Durch die Beibehaltung des Standardrollennamens kann EKS die Rolle für Add-Ons in neuen Clustern oder beim Hinzufügen von Add-Ons zu bestehenden Clustern vorab auswählen. Sie können diesen Namen weiterhin überschreiben, und die Rolle steht dann für das Add-On in Ihren Clustern zur Verfügung, muss jedoch manuell aus der Dropdown-Liste ausgewählt werden.

ii. Informationen zu Add-Ons, bei denen unter Status nicht die Option Abonnement erforderlich angezeigt wird und bei denen Sie Rollen mithilfe von IRSA konfigurieren möchten, finden Sie in der Dokumentation des Add-Ons, das Sie erstellen, um eine IAM-Richtlinie zu erstellen und sie einer Rolle zuzuordnen. Eine Liste der Add-Ons finden Sie unter the section called “ AWS Add-Ons”. Die Auswahl einer IAM-Rolle erfordert, dass Sie über einen IAM OpenID Connect (OIDC)-Anbieter für Ihren Cluster verfügen. Um festzustellen, ob Sie über einen solchen für Ihren Cluster verfügen, oder um einen zu erstellen, lesen Sie the section called “IAM-OIDC-Anbieter”.

c. Erweitern Sie Optionale Konfigurationseinstellungen.

d. Geben Sie unter Konfigurationswerte alle Add-On-spezifischen Konfigurationsinformationen ein. Weitere Informationen finden Sie in der Dokumentation für das Add-On, das Sie aktualisieren. Eine Liste der Add-Ons finden Sie unter the section called “ AWS Add-Ons” … Wählen Sie für die Konfliktlösungsmethode eine der Optionen aus. Wenn Sie benutzerdefinierte Werte für Add-On-Einstellungen festgelegt haben, empfehlen wir die Option Preserve (Beibehalten). Wenn Sie diese Option nicht auswählen, überschreibt Amazon EKS Ihre Werte mit seinen Standardwerten. Wenn Sie diese Option verwenden, empfehlen wir, dass Sie alle Feld- und Wertänderungen auf einem Nicht-Produktionscluster testen, bevor Sie das Add-on auf Ihrem Produktionscluster aktualisieren. Wenn Sie diesen Wert auf „Überschreiben“ ändern, werden alle Einstellungen auf die Amazon-EKS-Standardwerte geändert. Wenn Sie benutzerdefinierte Werte für Einstellungen festgelegt haben, werden diese möglicherweise mit den Amazon-EKS-Standardwerten überschrieben. Wenn Sie diesen Wert auf „Keine“ ändern, ändert Amazon EKS den Wert keiner der Einstellungen, aber die Aktualisierung schlägt möglicherweise fehl. Wenn das Update fehlschlägt, erhalten Sie eine Fehlermeldung, die Sie bei der Behebung des Konflikts unterstützt.

8. Wählen Sie Änderungen speichern aus.

Add-on aktualisieren (AWS CLI)

1. Sie müssen Version `2.12.3` oder höher oder Version `1.27.160` oder höher der AWS Befehlszeilenschnittstelle (AWS CLI) auf Ihrem Gerät installiert und konfiguriert haben oder AWS CloudShell. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Paketmanager wie `yum`apt-get, oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der AWS CLI. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Die AWS CLI-Version, in der installiert ist, AWS CloudShell kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zur Aktualisierung finden Sie im AWS CloudShell Benutzerhandbuch unter AWS CLI in Ihrem Home-Verzeichnis installieren.
2. Hier finden Sie eine Liste der installierten Add-Ons. Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters.

```
aws eks list-addons --cluster-name my-cluster
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "addons": [
        "coredns",
        "kube-proxy",
        "vpc-cni"
    ]
}
```

3. Zeigen Sie die aktuelle Version des Add-Ons an, die Sie aktualisieren möchten. Ersetzen Sie *my-cluster* mit Ihrem Clusternamen und *vpc-cni* mit dem Namen des Add-Ons, das Sie aktualisieren möchten.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 "addon.addonVersion" --output text
```

Eine Beispielausgabe sieht wie folgt aus.

```
v1.10.4-eksbuild.1
```

4. Bestimmen Sie, welche Versionen des Add-Ons für die Version Ihres Clusters verfügbar sind. *1.33*Ersetzen Sie es durch die Version Ihres Clusters und *vpc-cni* durch den Namen des Add-ons, das Sie aktualisieren möchten.

```
aws eks describe-addon-versions --kubernetes-version 1.33 --addon-name vpc-cni \
    --query 'addons[].addonVersions[].{Version: addonVersion, Defaultversion:
 compatibilities[0].defaultVersion}' --output table
```

Eine Beispielausgabe sieht wie folgt aus.

```
-----------------------------------------
|          DescribeAddonVersions         |
+-----------------+----------------------+
| Defaultversion  |       Version        |
+-----------------+----------------------+
|  False          |  v1.12.0-eksbuild.1  |
|  True           |  v1.11.4-eksbuild.1  |
|  False          |  v1.10.4-eksbuild.1  |
|  False          |  v1.9.3-eksbuild.1   |
+-----------------+----------------------+
```

Die Version `True` in der `Defaultversion`-Spalte ist standardmäßig die Version, mit der das Add-On erstellt wurde.

5. Aktualisieren Sie Ihr Add-on. Kopieren Sie den folgenden Befehl auf Ihr Gerät. Nehmen Sie bei Bedarf die folgenden Änderungen am Befehl vor, und führen Sie dann den geänderten Befehl aus. Weitere Informationen zu diesem Befehl finden Sie unter update-addon in der Amazon-EKS-Befehlszeilenreferenz.

   - Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters.
   - Ersetzen Sie *`vpc-cni`* durch den Namen des Add-Ons, das Sie aktualisieren möchten und das in der Ausgabe eines vorherigen Schritts zurückgegeben wurde.
   - Ersetzen Sie *`version-number`* durch die in der Ausgabe des vorherigen Schritts zurückgegebene Version, auf die Sie aktualisieren möchten. Für einige Add-Ons gibt es empfohlene Versionen. Weitere Informationen finden Sie in der Dokumentation für das Add-On, das Sie aktualisieren. Eine Liste der Add-Ons finden Sie unterthe section called “ AWS Add-Ons”. * Wenn das Add-on ein Kubernetes-Dienstkonto und eine IAM-Rolle verwendet, *`111122223333`* ersetzen Sie es durch Ihre Konto-ID und *`role-name`* durch den Namen einer vorhandenen IAM-Rolle, die Sie erstellt haben. Anweisungen zum Erstellen der Rolle finden Sie in der Dokumentation für das Add-On, das Sie erstellen. Eine Liste der Add-Ons finden Sie unter the section called “ AWS Add-Ons”. Die Angabe einer Servicekonto-Rolle setzt voraus, dass Sie über einen IAM OpenID Connect (OIDC)-Anbieter für Ihren Cluster verfügen. Um festzustellen, ob Sie über einen solchen für Ihren Cluster verfügen, oder um einen zu erstellen, lesen Sie the section called “IAM-OIDC-Anbieter”.

     Wenn das Add-On kein Kubernetes-Servicekonto und keine IAM-Rolle verwendet, löschen Sie die Zeile `serviceAccountRoleARN: arn:aws: iam::`*`111122223333`*`:role/`*`role-name`* .
   - Die `--resolve-conflicts PRESERVE`-Option behält die vorhandenen Werte für das Add-On bei. Wenn Sie benutzerdefinierte Werte für Add-On-Einstellungen festgelegt haben und diese Option nicht verwenden, überschreibt Amazon EKS Ihre Werte mit seinen Standardwerten. Wenn Sie diese Option verwenden, empfehlen wir, dass Sie alle Feld- und Wertänderungen auf einem Nicht-Produktionscluster testen, bevor Sie das Add-on auf Ihrem Produktionscluster aktualisieren. Wenn Sie diesen Wert auf `OVERWRITE` ändern, werden alle Einstellungen auf die Amazon-EKS-Standardwerte geändert. Wenn Sie benutzerdefinierte Werte für Einstellungen festgelegt haben, werden diese möglicherweise mit den Amazon-EKS-Standardwerten überschrieben. Wenn Sie diesen Wert auf `NONE` ändern, ändert Amazon EKS den Wert der

Einstellungen nicht, aber die Aktualisierung schlägt möglicherweise fehl. Wenn das Update fehlschlägt, erhalten Sie eine Fehlermeldung, die Sie bei der Behebung des Konflikts unterstützt.

- Wenn Sie die gesamte benutzerdefinierte Konfiguration entfernen möchten, führen Sie das Update mit der *--configuration-values '{}'*-Option durch. Dadurch werden alle benutzerdefinierten Konfigurationen auf die Standardwerte zurückgesetzt. Wenn Sie Ihre benutzerdefinierte Konfiguration nicht ändern möchten, geben Sie das Kennzeichen nicht an. *--configuration-values* Wenn Sie eine benutzerdefinierte Konfiguration anpassen möchten, ersetzen Sie *{}* durch die neuen Parameter.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-version
 version-number \
    --service-account-role-arn arn:aws: iam::111122223333:role/role-name --
configuration-values '{}' --resolve-conflicts PRESERVE
```

6. Überprüfen Sie den Status des Updates. *my-cluster*Ersetzen Sie es durch den Namen Ihres Clusters und *vpc-cni* durch den Namen des Add-ons, das Sie aktualisieren.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "addon": {
        "addonName": "vpc-cni",
        "clusterName": "my-cluster",
        "status": "UPDATING",
    }
}
```

Das Update ist abgeschlossen, wenn der Status `ACTIVE` ist.

Kompatibilität der Add-On-Version von Amazon EKS mit einem Cluster überprüfen

Bevor Sie ein Amazon-EKS-Add-On erstellen, müssen Sie sicherstellen, dass die Version des Amazon-EKS-Add-Ons mit Ihrem Cluster kompatibel ist.

Verwenden Sie die describe-addon-versions API, um die verfügbaren Versionen von EKS-Add-Ons aufzulisten und anzuzeigen, welche Kubernetes-Versionen die einzelnen Add-On-Versionen unterstützen.

1. Überprüfen Sie, ob die AWS-CLI installiert ist und mit `aws sts get-caller-identity` funktioniert. Sollte dieser Befehl nicht funktionieren, informieren Sie sich über die Ersten Schritte mit der AWS-CLI.
2. Bestimmen Sie den Namen des Add-Ons, für das Sie Informationen zur Versionskompatibilität abrufen möchten, z. B. `amazon-cloudwatch-observability`.
3. Bestimmen Sie die Kubernetes-Version Ihres Clusters, z. B. `1.33`.
4. Verwenden Sie die AWS CLI, um die Add-On-Versionen abzurufen, die mit der Kubernetes-Version Ihres Clusters kompatibel sind.

```
aws eks describe-addon-versions --addon-name amazon-cloudwatch-observability --
kubernetes-version 1.33
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "addons": [
        {
            "addonName": "amazon-cloudwatch-observability",
            "type": "observability",
            "addonVersions": [
                {
                    "addonVersion": "vX.X.X-eksbuild.X",
                    "architecture": [
                        "amd64",
                        "arm64"
                    ],
                    "computeTypes": [
                        "ec2",
                        "auto",
                        "hybrid"
                    ],
                    "compatibilities": [
                        {
                            "clusterVersion": "1.33",
                            "platformVersions": [
                                "*"
```

```
                            ],
                            "defaultVersion": true
                        }
                    ],
                }
            ]
        }
    ]
}
```

Diese Ausgabe zeigt, dass die Add-On-Version `vX.X.X-eksbuild.X` mit der Kubernetes-Clusterversion `1.33` kompatibel ist.

Kompatibilität von Add-Ons mit Datenverarbeitungstypen

Das `computeTypes`-Feld in der Ausgabe `describe-addon-versions` gibt die Kompatibilität eines Add-Ons mit EKS Auto Mode verwaltete Knoten oder Hybridknoten an. Mit `auto` gekennzeichnete Add-Ons funktionieren mit der cloudbasierten, AWS-verwalteten Infrastruktur von EKS Auto Mode, während mit `hybrid` gekennzeichnete Add-On in On-Premises-Knoten ausgeführt werden können, die mit der EKS-Cloud-Steuerebene verbunden sind.

Weitere Informationen finden Sie unter the section called “Überlegungen zu Amazon EKS Auto Mode”.

Entfernung eines Amazon-EKS-Add-Ons aus einem Cluster

Sie können ein Amazon EKS-Add-on mithilfe `eksctl` der oder der AWS-Managementkonsole AWS CLI aus Ihrem Cluster entfernen.

Wenn Sie ein Amazon-EKS-Add-On aus einem Cluster entfernen:

- Es gibt keine Ausfallzeit für die Funktionalität, die das Add-On bereitstellt.
- Wenn Sie IAM-Rollen für Servicekonten (IRSA) verwenden und das Add-On einer IAM-Rolle zugeordnet ist, wird die IAM-Rolle nicht entfernt.
- Wenn Sie Pod Identities verwenden, werden alle Pod-Identity-Zuordnungen, die dem Add-On gehören, entfernt. Wenn Sie die `--preserve` Option für die AWS CLI angeben, werden die Verknüpfungen beibehalten.
- Amazon EKS beendet die Verwaltung der Einstellungen für das Add-On.

- Die Konsole benachrichtigt Sie nicht mehr, wenn neue Versionen verfügbar sind.
- Sie können das Add-on nicht mit irgendwelchen AWS Tools oder aktualisieren APIs.
- Sie können die Add-On-Software auf Ihrem Cluster beibehalten, damit Sie sie selbst verwalten können, oder Sie können die Add-On-Software aus Ihrem Cluster entfernen. Sie sollten die Add-On-Software nur dann aus Ihrem Cluster entfernen, wenn keine Ressourcen in Ihrem Cluster von der Funktionalität abhängen, die das Add-On bereitstellt.

Voraussetzungen

Führen Sie Folgendes aus, bevor Sie ein Add-On erstellen:

- Ein vorhandener Amazon-EKS-Cluster. Informationen zum Bereitstellen finden Sie unter Erste Schritte.
- Überprüfen Sie, ob Ihr Add-On eine IAM-Rolle erfordert. Die Ausgabe des obigen Befehls sieht in etwa folgendermaßen aus (JSON format).
- Version `0.215.0` oder höher des `eksctl`-Befehlszeilen-Tools, das auf Ihrem Computer oder in der AWS CloudShell installiert ist. Informationen zum Installieren und Aktualisieren von `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.

Verfahren

Beim Entfernen eines Amazon-EKS-Add-Ons stehen Ihnen zwei Optionen zur Verfügung:

- Beibehalten von Add-on-Software auf Ihrem Cluster – Diese Option entfernt die Amazon-EKS-Verwaltung aller Einstellungen. Amazon EKS wird außerdem die Möglichkeit aufgehoben, Sie über Updates zu informieren und das Amazon-EKS-Add-on automatisch zu aktualisieren, nachdem Sie ein Update eingeleitet haben. Es behält jedoch die Add-on-Software auf Ihrem Cluster bei. Diese Option macht das Add-on zu einem selbstverwalteten Add-on statt einem Amazon-EKS-Add-on. Bei dieser Option gibt es keine Ausfallzeiten für das Add-On.
- Entfernen Sie die Add-on-Software vollständig aus Ihrem Cluster – Wir empfehlen, dass Sie das Amazon-EKS-Add-on nur aus Ihrem Cluster entfernen, wenn auf Ihrem Cluster keine Ressourcen vorhanden sind, die davon abhängig sind.

Sie können ein Amazon EKS-Add-on mithilfe `eksctl` der AWS-Managementkonsole oder der AWS CLI entfernen.

Add-on entfernen (eksctl)

1. Bestimmen Sie die aktuell auf Ihrem Cluster installierten Add-Ons. Ersetzen Sie *my-cluster* mit dem Namen Ihres Clusters.

   ```
   eksctl get addon --cluster my-cluster
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   NAME        VERSION              STATUS  ISSUES  IAMROLE  UPDATE AVAILABLE
   coredns     v1.8.7-eksbuild.2    ACTIVE  0
   kube-proxy  v1.23.7-eksbuild.1   ACTIVE  0
   vpc-cni     v1.10.4-eksbuild.1   ACTIVE  0
   [...]
   ```

   IIhre Ausgabe sieht möglicherweise anders aus, je nachdem, welche Add-Ons und Versionen Sie auf Ihrem Cluster haben.

2. Entfernen Sie das Add-On. Ersetzen Sie *my-cluster* mit dem Namen Ihres Clusters und *name-of-add-on* mit dem Namen des Add-Ons, das in der Ausgabe des vorherigen Schritts zurückgegeben wurde. Wenn Sie die *--preserve* Option entfernen, wird die Zusatzsoftware nicht mehr von Amazon EKS verwaltet, sondern auch aus Ihrem Cluster gelöscht.

   ```
   eksctl delete addon --cluster my-cluster --name name-of-addon --preserve
   ```

   Weitere Informationen zum Entfernen von Add-Ons finden Sie unter Entfernung von Add-Ons in der `eksctl`-Dokumentation.

Add-on entfernen (AWS Konsole)

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich Cluster aus.
3. Wählen Sie den Namen des Clusters aus, für den Sie das Amazon-EKS-Add-On entfernen möchten.
4. Wählen Sie die Registerkarte Add-ons.
5. Wählen Sie das Add-On aus, das Sie entfernen möchten.
6. Wählen Sie Remove (Entfernen) aus.

7. Gehen Sie im ***name of addon*** Bestätigungsdialogfeld Entfernen: wie folgt vor:
   a. Wenn Amazon EKS die Verwaltung von Einstellungen für das Add-on einstellen soll, wählen Sie Auf dem Cluster beibehalten. Tun Sie dies, wenn Sie die Add-on-Software auf Ihrem Cluster behalten möchten. Auf diese Weise können Sie alle Einstellungen des Add-ons selbst verwalten.
   b. Geben Sie den Add-On-Namen ein.
   c. Wählen Sie Remove (Entfernen) aus.

Add-on entfernen (AWS CLI)

1. Sie benötigen eine Version `0.215.0` oder eine neuere Version des `eksctl` Befehlszeilentools, das auf Ihrem Gerät installiert ist oder AWS CloudShell. Informationen zum Installieren und Aktualisieren von `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.
2. Hier finden Sie eine Liste der installierten Add-Ons. Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters.

   ```
   aws eks list-addons --cluster-name my-cluster
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   {
       "addons": [
           "coredns",
           "kube-proxy",
           "vpc-cni",
           "name-of-addon"
       ]
   }
   ```

3. Entfernen Sie das installierte Add-On. Ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters und *`name-of-add-on`* durch den Namen des Add-Ons, das Sie entfernen. Durch das Entfernen *`--preserve`* wird die Zusatzsoftware aus Ihrem Cluster gelöscht.

   ```
   aws eks delete-addon --cluster-name my-cluster --addon-name name-of-addon --preserve
   ```

   Die gekürzte Beispielausgabe lautet wie folgt.

```
{
    "addon": {
        "addonName": "name-of-add-on",
        "clusterName": "my-cluster",
        "status": "DELETING",
    }
}
```

4. Überprüfen Sie den Status der Entfernung. *my-cluster*Ersetzen Sie es durch den Namen Ihres Clusters und *name-of-addon* durch den Namen des Add-ons, das Sie entfernen möchten.

```
aws eks describe-addon --cluster-name my-cluster --addon-name name-of-addon
```

Nachdem das Add-On entfernt wurde, sieht die Beispielausgabe wie folgt aus.

```
An error occurred (ResourceNotFoundException) when calling the DescribeAddon
 operation: No addon: name-of-addon found in cluster: my-cluster
```

IAM-Rollen für Amazon-EKS-Add-Ons

Bestimmte Amazon-EKS-Add-Ons erfordern IAM-Rollen und -Berechtigungen, um AWS-APIs aufzurufen. Das Add-On für Amazon-VPC-CNI ruft beispielsweise bestimmte AWS-APIs auf, um die Netzwerkressourcen in Ihrem Konto zu konfigurieren. Für diese Add-Ons muss über IAM eine Berechtigung gewährt werden. Genauer gesagt muss das Servicekonto des Pods, in dem das Add-On ausgeführt wird, einer IAM-Rolle mit einer bestimmten IAM-Richtlinie zugeordnet werden.

Die empfohlene Methode zum Gewähren von AWS-Berechtigungen für Cluster-Workloads ist die Verwendung des Amazon-EKS-Features für Pod Identities. Sie können eine Pod-Identity-Zuordnung verwenden, um das Servicekonto eines Add-Ons einer IAM-Rolle zuzuordnen. Wenn ein Pod ein Servicekonto mit einer Zuordnung verwendet, legt Amazon EKS Umgebungsvariablen in den Containern des Pods fest. Die Umgebungsvariablen konfigurieren die AWS-SDKs, einschließlich der AWS-CLI, für die Verwendung der Anmeldeinformationen für EKS Pod Identity. Weitere Informationen finden Sie unter the section called “Pod Identity”.

Amazon-EKS-Add-Ons können Sie bei der Verwaltung des Lebenszyklus von Pod-Identity-Zuordnungen unterstützen, die dem Add-On entsprechen. Beispielsweise können Sie ein Amazon-EKS-Add-On und die erforderliche Pod-Identity-Zuordnung mit einem einzigen API-Aufruf erstellen

oder aktualisieren. Amazon EKS stellt auch eine API zum Abrufen empfohlener IAM-Richtlinien zur Verfügung.

1. Überprüfen Sie, ob der Amazon EKS Pod Identity Agent in Ihrem Cluster eingerichtet ist.
2. Ermitteln Sie mithilfe der `describe-addon-versions` AWS-CLI-Operation, ob das Add-On, das Sie installieren möchten, IAM-Berechtigungen erfordert. Wenn es sich beim `requiresIamPermissions`-Flag um `true` handelt, sollten Sie die `describe-addon-configurations`-Operation verwenden, um die vom Add-On benötigten Berechtigungen zu ermitteln. Die Antwort enthält eine Liste mit vorgeschlagenen verwalteten IAM-Richtlinien.
3. Rufen Sie den Namen des Kubernetes- Servicekontos und die IAM-Richtlinie mithilfe der `describe-addon-configuration`-CLI-Operation ab. Bewerten Sie den Umfang der vorgeschlagenen Richtlinie anhand Ihrer Sicherheitsanforderungen.
4. Erstellen Sie eine IAM-Rolle mithilfe der vorgeschlagenen Berechtigungsrichtlinie und der von Pod Identity benötigten Vertrauensrichtlinie. Weitere Informationen finden Sie unter the section called “Erstellen Sie eine Pod Identity-Zuordnung (AWS Konsole)”.
5. Erstellen oder aktualisieren Sie ein Amazon-EKS-Add-On mithilfe der CLI. Geben Sie mindestens eine Pod-Identity-Zuordnung an. Eine Pod-Identity-Zuordnung besteht aus dem Namen eines Kubernetes-Servicekontos und der ARN der IAM-Rolle.
   - Pod-Identitätszuordnungen, die mithilfe der Add-On-APIs erstellt wurden, sind Eigentum des jeweiligen Add-Ons. Wenn Sie das Add-On löschen, wird auch die Pod-Identity-Zuordnung gelöscht. Sie können diese kaskadenartige Löschung verhindern, indem Sie beim Löschen eines Add-Ons über die AWS-CLI oder API die entsprechende `preserve`-Option verwenden. Bei Bedarf können Sie die Pod-Identity-Zuordnung auch direkt aktualisieren oder löschen. Add-Ons können keine vorhandenen Pod-Identity-Zuordnungen übernehmen. Sie müssen die vorhandene Zuordnung löschen und sie mithilfe eines Vorgangs zu Add-On-Erstellung oder -Aktualisierung neu erstellen.
   - Amazon EKS empfiehlt die Verwendung von Pod-Identity-Zuordnungen zur Verwaltung von IAM-Berechtigungen für Add-Ons. Die bisherige Methode, IAM-Rollen für Servicekonten (IRSA), wird weiterhin unterstützt. Sie können sowohl eine IRSA-`serviceAccountRoleArn` als auch eine Pod-Identity-Zuordnung für ein Add-On festlegen. Wenn der EKS-Pod-Identity-Agent auf dem Cluster installiert ist, wird die `serviceAccountRoleArn` ignoriert, und EKS verwendet die bereitgestellte Pod-Identity-Zuordnung. Wenn die Pod Identity nicht aktiviert ist, wird die `serviceAccountRoleArn` verwendet.
   - Wenn Sie die Pod-Identity-Zuordnungen für ein vorhandenes Add-On aktualisieren, initiiert Amazon EKS einen fortlaufenden Neustart der Add-On-Pods.

Abrufen von IAM-Informationen zu einem Amazon-EKS-Add-On

Bevor Sie ein Add-On erstellen, ermitteln Sie mithilfe der AWS-CLI:

- Ob das Add-On IAM-Berechtigungen benötigt
- Die empfohlene zu verwendende IAM-Richtlinie

Verfahren

1. Bestimmen Sie den Namen des Add-ons, das Sie installieren möchten, und die Kubernetes-Version Ihres Clusters. Weitere Informationen zu Add-ons finden Sie unter the section called “Amazon-EKS-Add-ons”.
2. Verwenden Sie die AWS-CLI, um zu bestimmen, ob das AddO-n IAM-Berechtigungen erfordert.

   ```
   aws eks describe-addon-versions \
   --addon-name <addon-name> \
   --kubernetes-version <kubernetes-version>
   ```

   Zum Beispiel:

   ```
   aws eks describe-addon-versions \
   --addon-name aws-ebs-csi-driver \
   --kubernetes-version 1.30
   ```

   Sehen Sie sich die folgende Beispielausgabe an. Beachten Sie, dass `requiresIamPermissions` `true` ist und die Standard-Add-On-Version darstellt. Sie müssen die Add-On-Version angeben, wenn Sie die empfohlene IAM-Richtlinie abrufen.

   ```
   {
       "addons": [
           {
               "addonName": "aws-ebs-csi-driver",
               "type": "storage",
               "addonVersions": [
                   {
                       "addonVersion": "v1.31.0-eksbuild.1",
                       "architecture": [
                           "amd64",
                           "arm64"
   ```

```
                    ],
                    "compatibilities": [
                        {
                            "clusterVersion": "1.30",
                            "platformVersions": [
                                "*"
                            ],
                            "defaultVersion": true
                        }
                    ],
                    "requiresConfiguration": false,
                    "requiresIamPermissions": true
                },
[...]
```

3. Wenn das Add-On IAM-Berechtigungen erfordert, verwenden Sie die AWS-CLI, um eine empfohlene IAM-Richtlinie abzurufen.

```
aws eks describe-addon-configuration \
--query podIdentityConfiguration \
--addon-name <addon-name> \
--addon-version <addon-version>
```

Zum Beispiel:

```
aws eks describe-addon-configuration \
--query podIdentityConfiguration \
--addon-name aws-ebs-csi-driver \
--addon-version v1.31.0-eksbuild.1
```

Überprüfen Sie die folgende Ausgabe. Beachten Sie den `recommendedManagedPolicies`.

```
[
    {
        "serviceAccount": "ebs-csi-controller-sa",
        "recommendedManagedPolicies": [
            "arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy"
        ]
    }
]
```

4. Erstellen Sie eine IAM-Rolle und fügen Sie die empfohlene verwaltete Richtlinie an. Alternativ können Sie die verwaltete Richtlinie überprüfen und die Berechtigungen entsprechend einschränken. Weitere Informationen finden Sie unter the section called “Erstellen Sie eine Pod Identity-Zuordnung (AWS Konsole)”.

Unterstützungsreferenz für Pod Identity

Die folgende Tabelle gibt an, ob bestimmte Amazon-EKS-Add-Ons EKS Pod Identity unterstützen.

Add-On-Name	Unterstützung für Pod Identity	Erforderliche Mindestversion
Amazon-EBS-CSI-Treiber	Ja	v1.26.0-eksbuild.1
Amazon VPC CNI	Ja	v1.15.5-eksbuild.1
Amazon EFS-CSI-Treiber	Ja	v2.0.5-eksbuild.1
AWS Distro for OpenTelemetry	Ja	v0.94.1-eksbuild.1
CSI-Treiber für Mountpoint für Amazon S3	Nein	N/A
Amazon-CloudWatch-Beobachtbarkeits-Agent	Ja	v3.1.0-eksbuild.1

Diese Tabelle wurde zuletzt am 28. Oktober 2024 aktualisiert.

Verwendung von Pod Identities zum Zuweisen einer IAM-Rolle zu einem Amazon-EKS-Add-On

Bestimmte Amazon-EKS-Add-Ons erfordern IAM-Rollen und -Berechtigungen. Bevor Sie ein Amazon-EKS-Add-On aktualisieren, um eine Pod-Identity-Zuordnung zu verwenden, überprüfen Sie die zu verwendende Rolle und Richtlinie. Weitere Informationen finden Sie unter the section called “IAM-Informationen abrufen”.

1. Bestimmen Sie:
   - `cluster-name` – Den Namen des Clusters, auf dem das Add-On installiert werden soll.

- `addon-name` – Den Namen des zu installierenden Add-ons.
- `service-account-name` – Den Namen des vom Add-On verwendeten Kubernetes-Servicekontos.
- `iam-role-arn` – Die ARN einer IAM-Rolle mit ausreichenden Berechtigungen für das Add-On. Die Rolle muss über die erforderliche Vertrauensrichtlinie für die EKS Pod Identity verfügen. Weitere Informationen finden Sie unter the section called “Erstellen Sie eine Pod Identity-Zuordnung (AWS Konsole)”.

2. Aktualisieren Sie das Add-on mithilfe der AWS-CLI. Sie können beim Erstellen eines Add-Ons mit derselben `--pod-identity-assocations`-Syntax auch Pod-Identity-Zuordnungen angeben. Beachten Sie, dass beim Angeben von Pod-Identity-Zuordnungen beim Aktualisieren eines Add-Ons alle vorherigen Pod-Identity-Zuordnungen überschrieben werden.

```
aws eks update-addon --cluster-name <cluster-name> \
--addon-name <addon-name> \
--pod-identity-associations 'serviceAccount=<service-account-name>,roleArn=<role-
arn>'
```

   Zum Beispiel:

```
aws eks update-addon --cluster-name mycluster \
--addon-name aws-ebs-csi-driver \
--pod-identity-associations 'serviceAccount=ebs-csi-controller-
sa,roleArn=arn:aws:iam::123456789012:role/StorageDriver'
```

3. Überprüfen Sie, ob die Pod-Identity-Zuordnung erstellt wurde:

```
aws eks list-pod-identity-associations --cluster-name <cluster-name>
```

   Wenn Sie erfolgreich waren, sollte die Ausgabe folgendermaßen oder ähnlich aussehen. Notieren Sie sich die OwnerARN des EKS-Add-Ons.

```
{
    "associations": [
        {
            "clusterName": "mycluster",
            "namespace": "kube-system",
            "serviceAccount": "ebs-csi-controller-sa",
            "associationArn": "arn:aws:eks:us-
west-2:123456789012:podidentityassociation/mycluster/a-4wvljrezsukshq1bv",
```

```
            "associationId": "a-4wvljrezsukshq1bv",
            "ownerArn": "arn:aws:eks:us-west-2:123456789012:addon/mycluster/aws-ebs-
csi-driver/9cc7ce8c-2e15-b0a7-f311-426691cd8546"
        }
    ]
}
```

Entfernung von Pod-Identity-Zuordnungen aus einem Amazon-EKS-Add-On

Entfernen Sie die Pod-Identity-Zuordnungen aus einem Amazon-EKS-Add-On.

1. Bestimmen Sie:
   - `cluster-name` – Den Namen des EKS-Clusters, auf dem das Add-On installiert werden soll.
   - `addon-name` – Den Namen des zu installierenden Amazon-EKS-Add-Ons.
2. Aktualisieren Sie das Add-On, um ein leeres Array mit Pod-Identity-Zuordnungen anzugeben.

```
aws eks update-addon --cluster-name <cluster-name> \
--addon-name <addon-name> \
--pod-identity-associations "[]"
```

Fehlerbehebung bei Pod Identities für EKS-Add-Ons

Wenn bei Ihren Add-Ons beim Ausführen von AWS-API-, SDK- oder CLI-Operationen Fehler auftreten, überprüfen Sie Folgendes:

- Der Pod Identity Agent ist in Ihrem Cluster installiert.
  - Informationen zur Installation des Pod Identity Agent finden Sie unter the section called “Agent einrichten”.
- Das Add-On verfügt über eine gültige Pod-Identity-Zuordnung.
  - Verwenden Sie die AWS-CLI, um die Zuordnungen für den vom AddO-n verwendeten Servicekonto-Namen abzurufen.

    ```
    aws eks list-pod-identity-associations --cluster-name <cluster-name>
    ```
- Die IAM-Rolle verfügt über die erforderliche Vertrauensrichtlinie für Pod Identities.
  - Verwenden Sie die AWS-CLI, um die Vertrauensrichtlinie für ein Add-On abzurufen.

```
aws iam get-role --role-name <role-name> --query Role.AssumeRolePolicyDocument
```

- Die IAM-Rolle verfügt über die erforderlichen Berechtigungen für das Add-On.
  - Verwenden Sie AWS CloudTrail, um `AccessDenied` oder `UnauthorizedOperation`-Ereignisse zu überprüfen.
- Der Name des Servicekontos in der Pod-Identity-Zuordnung entspricht dem vom Add-On verwendeten Namen des Servicekontos.
  - Informationen zu den verfügbaren Add-Ons finden Sie unter the section called “ AWS Add-Ons”.
- Konfiguration von MutatingWebhookConfiguration mit dem Namen `pod-identity-webhook` überprüfen
  - `admissionReviewVersions` des Webhooks muss `v1beta1` sein und ist mit `v1` nicht kompatibel.

Felder für die Anpassung von Amazon-EKS-Add-On festlegen

Amazon-EKS-Add-ons werden in Ihrem Cluster unter Verwendung von bewährten Standardkonfigurationen installiert. Weitere Informationen zum Hinzufügen eines Amazon-EKS-Add-ons zu Ihrem Cluster finden Sie unter the section called “Amazon-EKS-Add-ons”.

Möglicherweise möchten Sie die Konfiguration eines Amazon-EKS-Add-ons anpassen, um erweiterte Funktionen zu aktivieren. Amazon EKS verwendet das serverseitige Feature Kubernetes, um die Verwaltung eines Add-Ons durch Amazon EKS zu ermöglichen, ohne Ihre Einstellungen zu überschreiben, die nicht von Amazon EKS verwaltet werden. Weitere Informationen dazu finden Sie unter Serverseitiges Anwenden in der Kubernetes-Dokumentation. Um dies zu erreichen, verwaltet Amazon EKS einen Mindestsatz von Feldern für jedes Add-on, das installiert wird. Sie können alle Felder problemlos ändern, die nicht von Amazon EKS oder einem anderen Prozess der Kubernetes-Steuerebene verwaltet werden, z. B. `kube-controller-manager`.

> **Important**
>
> Das Ändern eines von Amazon EKS verwalteten Felds verhindert, dass Amazon EKS das Add-on verwaltet und kann dazu führen, dass Ihre Änderungen überschrieben werden, wenn ein Add-on aktualisiert wird.

Syntax für Feldmanagement

Wenn Sie Details für ein Kubernetes-Objekt anzeigen, werden sowohl verwaltete als auch nicht verwaltete Felder in der Ausgabe zurückgegeben. Verwaltete Felder können einer der folgenden Typen sein:

- Vollständig verwaltet – Alle Schlüssel für das Feld werden von Amazon EKS verwaltet. Änderungen an einem beliebigen Wert verursachen einen Konflikt.
- Teilweise verwaltet – Einige Schlüssel für das Feld werden von Amazon EKS verwaltet. Nur Änderungen an den Schlüsseln, die explizit von Amazon EKS verwaltet werden, verursachen einen Konflikt.

Beide Arten von Feldern sind gekennzeichnet mit `manager: eks`.

Jeder Schlüssel ist entweder ein `.`, welches das Feld darstellt, das immer einem leeren Satz zugeordnet wird, oder eine Zeichenfolge, die ein Unterfeld oder Element darstellt. Die Ausgabe für die Feldverwaltung besteht aus folgenden Angabetypen:

- `f:`*`name`* , wobei *`name`* der Name eines Feldes in einer Liste ist.
- `k:`*`keys`* , wobei *`keys`* eine Karte der Felder eines Listenelements ist.
- `v:`*`value`* , wobei *`value`* der exakte JSON-formatierte Wert eines Listenelements ist.
- `i:`*`index`* , wobei *`index`* die Position eines Elements in der Liste ist.

Die folgenden Ausgabeteile für das CoreDNS Add-on veranschaulichen die vorherigen Angaben:

- Vollständig verwaltete Felder – Wenn ein verwaltetes Feld ein `f:` (Feld) festgelegt hat, aber keinen `k:` (Schlüssel), dann wird das gesamte Feld verwaltet. Änderungen an Werten in diesem Feld verursachen einen Konflikt.

  In der folgenden Ausgabe sehen Sie, dass der Container mit dem Namen `coredns` verwaltet von `eks`. Die`args`, `image` und `imagePullPolicy` Unterfelder werden auch verwaltet von `eks`. Änderungen an Werten in diesen Feldern verursachen einen Konflikt.

```
[...]
f:containers:
  k:{"name":"coredns"}:
  .: {}
  f:args: {}
```

```
  f:image: {}
  f:imagePullPolicy: {}
[...]
manager: eks
[...]
```

- Teilweise verwaltete Felder – Wenn für einen verwalteten Schlüssel ein Wert angegeben ist, werden die angegebenen Schlüssel für dieses Feld verwaltet. Das Ändern der angegebenen Schlüssel führt zu einem Konflikt.

  In der folgenden Ausgabe sehen Sie, dass `eks` die `config-volume` und `tmp` Volumes verwaltet, welche mit dem `name`-Schlüssel eingestellt wurden.

```
[...]
f:volumes:
  k:{"name":"config-volume"}:
    .: {}
    f:configMap:
      f:items: {}
      f:name: {}
    f:name: {}
  k:{"name":"tmp"}:
    .: {}
    f:name: {}
[...]
manager: eks
[...]
```

- Hinzufügen von Schlüsseln zu teilweise verwalteten Feldern – Wenn nur ein bestimmter Schlüsselwert verwaltet wird, können Sie sicher zusätzliche Schlüssel wie Argumente zu einem Feld hinzufügen, ohne einen Konflikt zu verursachen. Wenn Sie zusätzliche Schlüssel hinzufügen, stellen Sie sicher, dass das Feld nicht zuerst verwaltet wird. Das Hinzufügen oder Ändern eines verwalteten Werts verursacht einen Konflikt.

  In der folgenden Ausgabe sehen Sie, dass sowohl der Schlüssel `name` als auch das Feld `name` verwaltet werden. Das Hinzufügen oder Ändern eines Containernamens verursacht einen Konflikt mit diesem verwalteten Schlüssel.

```
[...]
f:containers:
  k:{"name":"coredns"}:
[...]
```

```
    f:name: {}
[...]
manager: eks
[...]
```

Verfahren

Sie können `kubectl` verwenden, um zu sehen, welche Felder von Amazon EKS für ein Amazon EKS Add-on verwaltet werden.

Sie können alle Felder problemlos ändern, die nicht von Amazon EKS oder einem anderen Prozess der Kubernetes-Steuerebene verwaltet werden, z. B. `kube-controller-manager`.

1. Bestimmen Sie das Add-on, das Sie überprüfen möchten. Informationen zu allen `deployments` und DaemonSets die in Ihrem Cluster bereitgestellt werden, finden Sie unter the section called “Zugriff auf Cluster-Ressourcen”.
2. Führen Sie den folgenden Befehl aus, um die verwalteten Felder für ein Add-on anzuzeigen:

   ```
   kubectl get type/add-on-name -n add-on-namespace -o yaml
   ```

   Sie können beispielsweise die verwalteten Felder für das CoreDNS Add-on mit dem folgenden Befehl sehen.

   ```
   kubectl get deployment/coredns -n kube-system -o yaml
   ```

   Die Feldverwaltung wird im folgenden Abschnitt in der zurückgegebenen Ausgabe aufgeführt.

   ```
   [...]
   managedFields:
     - apiVersion: apps/v1
       fieldsType: FieldsV1
       fieldsV1:
   [...]
   ```

   Note

   Wenn `managedFields` in der Ausgabe nicht angezeigt wird, fügen Sie dem Befehl `--show-managed-fields` hinzu, und führen Sie ihn erneut aus. Die Version von `kubectl`,

die Sie verwenden, bestimmt, ob verwaltete Felder standardmäßig zurückgegeben werden.

Nächste Schritte

Passen Sie die Felder, die nicht zu AWS gehören, für Ihr Add-On an.

Validierung von Container-Image-Signaturen während der Bereitstellung

Wenn Sie AWS Signer nutzen und zum Zeitpunkt der Bereitstellung signierte Container-Images verifizieren möchten, können Sie auf eine der folgenden Lösungen zurückgreifen:

- Gatekeeper und Ratify – Verwenden Sie Gatekeeper als Zugangs-Controller und Ratify (mit einem konfigurierten AWS-Signer-Plugin) als Webhook für die Validierung von Signaturen.
- Kyverno – Eine Kubernetes-Richtlinien-Engine, die mit einem AWS-Signer-Plugin zur Validierung von Signaturen konfiguriert ist.

> **Note**
>
> Bevor Sie die Signaturen von Container-Images verifizieren, müssen Sie den Trust Store und die Vertrauensrichtlinie für die Notation entsprechend den Anforderungen des ausgewählten Zugangscontrollers konfigurieren.

EKS-Fähigkeiten

> **Tip**
>
> Erste Schritte: ACK-Fähigkeit erstellen | Argo-CD-Funktion erstellen | Kro-Fähigkeit erstellen

Amazon EKS Capabilities ist ein mehrschichtiger Satz vollständig verwalteter Cluster-Funktionen, mit denen Entwickler schneller arbeiten und die Komplexität der Erstellung und Skalierung mit Kubernetes verringern können. EKS-Funktionen sind native Kubernetes-Funktionen für die deklarative kontinuierliche Bereitstellung, das AWS Ressourcenmanagement sowie die Erstellung und Orchestrierung von Kubernetes-Ressourcen, die alle vollständig von verwaltet werden. AWS Mit EKS Capabilities können Sie sich mehr auf den Aufbau und die Skalierung Ihrer Workloads konzentrieren und die betriebliche Belastung dieser grundlegenden Plattformdienste auf diese abwälzen. AWS Diese Funktionen werden innerhalb von EKS und nicht in Ihren Clustern ausgeführt, sodass Sie wichtige Plattformkomponenten auf Ihren Worker-Knoten nicht mehr installieren, warten und skalieren müssen.

Zu Beginn können Sie eine oder mehrere EKS-Funktionen auf einem neuen oder vorhandenen EKS-Cluster erstellen. Dazu können Sie die AWS CLI, EKS AWS-Managementkonsole APIs, eksctl oder Ihre bevorzugten infrastructure-as-code Tools verwenden. EKS-Funktionen sind zwar so konzipiert, dass sie zusammenarbeiten, es handelt sich jedoch um unabhängige Cloud-Ressourcen, die Sie je nach Anwendungsfall und Anforderungen auswählen können.

Alle von EKS unterstützten Kubernetes-Versionen werden für EKS-Funktionen unterstützt.

> **Note**
>
> EKS-Funktionen sind in allen AWS Handelsregionen verfügbar, in denen Amazon EKS verfügbar ist. Eine Liste der unterstützten Regionen finden Sie unter Amazon EKS-Endpunkte und Kontingente in der AWS Allgemeinen Referenz.

Verfügbare Funktionen

AWS Controller für Kubernetes (ACK)

ACK ermöglicht die Verwaltung von AWS Ressourcen mithilfe von Kubernetes APIs, sodass Sie S3-Buckets, RDS-Datenbanken, IAM-Rollen und andere AWS Ressourcen mithilfe von benutzerdefinierten Kubernetes-Ressourcen erstellen und verwalten können. ACK gleicht Ihren gewünschten Status kontinuierlich mit dem aktuellen Status ab und korrigiert alle Abweichungen im Laufe der Zeit AWS, um sicherzustellen, dass Ihr System einwandfrei funktioniert und Ihre Ressourcen wie angegeben konfiguriert sind. Sie können AWS Ressourcen zusammen mit Ihren Kubernetes-Workloads mithilfe derselben Tools und Workflows verwalten, wobei mehr als 50 AWS Dienste unterstützt werden, darunter S3, RDS, DynamoDB und Lambda. ACK unterstützt konto- und regionsübergreifendes Ressourcenmanagement und ermöglicht so komplexe Systemverwaltungsarchitekturen mit mehreren Konten und Clustern. ACK unterstützt schreibgeschützte Ressourcen und die Einführung von schreibgeschützten Ressourcen und erleichtert so die Migration von anderen Infrastruktur-as-Code-Tools zu Ihren Kubernetes-basierten Systemen.

Erfahren Sie mehr über ACK →

Argo CD

Argo CD implementiert eine GitOps basierte kontinuierliche Bereitstellung für Ihre Anwendungen und verwendet Git-Repositorys als Informationsquelle für Ihre Workloads und den Systemstatus. Argo CD synchronisiert automatisch Anwendungsressourcen aus Ihren Git-Repositorys mit Ihren Clustern und erkennt und behebt Abweichungen, um sicherzustellen, dass Ihre bereitgestellten Anwendungen dem gewünschten Status entsprechen. Sie können Anwendungen in mehreren Clustern von einer einzigen Argo-CD-Instanz aus bereitstellen und verwalten, wobei die automatische Bereitstellung aus Git-Repositorys erfolgt, wann immer Änderungen festgeschrieben werden. Die gemeinsame Verwendung von Argo CD und ACK bietet ein grundlegendes GitOps System, das die Verwaltung der Abhängigkeit von Arbeitslasten vereinfacht und Systemdesigns unterstützt, einschließlich Cluster- und Infrastrukturmanagement im großen Maßstab. Argo CD lässt sich zur Authentifizierung und Autorisierung in AWS Identity Center integrieren und bietet eine gehostete Argo-Benutzeroberfläche zur Visualisierung des Anwendungszustands und des Bereitstellungsstatus.

Erfahren Sie mehr über Argo CD →

kro (Kube Resource Orchestrator)

Mit kro können Sie benutzerdefinierte Kubernetes erstellen, die mehrere Ressourcen zu Abstraktionen auf höherer Ebene zusammensetzen APIs , sodass Plattformteams wiederverwendbare Muster für gängige Ressourcenkombinationen — Cloud-Bausteine — definieren können. Mit kro können Sie sowohl Kubernetes als auch AWS Ressourcen zu einheitlichen Abstraktionen zusammenfassen und dabei eine einfache Syntax verwenden, um dynamische Konfigurationen und bedingte Logik zu ermöglichen. kro ermöglicht es Plattformteams, Self-Service-Funktionen mit geeigneten Leitplanken bereitzustellen, sodass Entwickler komplexe Infrastrukturen mithilfe einfacher, speziell entwickelter Systeme bereitstellen und APIs gleichzeitig organisatorische Standards und Best Practices einhalten können. KRO-Ressourcen sind einfach Kubernetes-Ressourcen und werden in Kubernetes-Manifesten spezifiziert, die in Git gespeichert oder per Push übertragen werden können zu OCI- kompatible Register wie Amazon ECR für eine breite organisatorische Verteilung.

Erfahren Sie mehr über Kro →

Vorteile der EKS-Fähigkeiten

Die Funktionen von EKS werden vollständig von verwaltet AWS, sodass keine Installation, Wartung und Skalierung grundlegender Cluster-Services erforderlich sind. AWS kümmert sich um Sicherheitspatches, Updates und Betriebsmanagement, sodass sich Ihre Teams auf die Entwicklung und AWS nicht auf den Clusterbetrieb konzentrieren können. Im Gegensatz zu herkömmlichen Kubernetes-Add-Ons, die Cluster-Ressourcen verbrauchen, werden Funktionen in EKS und nicht auf Ihren Worker-Knoten ausgeführt. Dadurch werden Clusterkapazität und Ressourcen für Ihre Workloads freigegeben und gleichzeitig der Betriebsaufwand für die Verwaltung von Cluster-Controllern und anderen Plattformkomponenten minimiert.

Mit EKS-Funktionen können Sie Bereitstellungen, Ressourcen, benutzerdefinierte AWS Kubernetes-Ressourcen und Kompositionen mithilfe von nativem Kubernetes und Tools wie verwalten. APIs `kubectl` Alle Funktionen funktionieren im Kontext Ihrer Cluster und erkennen und korrigieren automatisch Konfigurationsabweichungen sowohl bei Anwendungs- als auch bei Cloud-Infrastrukturressourcen. Sie können Ressourcen für mehrere Cluster, AWS Konten und Regionen von einem einzigen Kontrollpunkt aus bereitstellen und verwalten und so den Betrieb in komplexen, verteilten Umgebungen vereinfachen.

Die EKS-Funktionen sind für GitOps Workflows konzipiert und bieten deklaratives, versionskontrolliertes Infrastruktur- und Anwendungsmanagement. Änderungen fließen von Git durch

das System und bieten Audit-Trails, Rollback-Funktionen und Workflows für die Zusammenarbeit, die sich in Ihre bestehenden Entwicklungspraktiken integrieren lassen. Dieser Kubernetes-native Ansatz bedeutet, dass Sie nicht mehrere Tools verwenden oder infrastructure-as-code Systeme außerhalb Ihrer Cluster verwalten müssen, und es gibt eine einzige Informationsquelle, auf die Sie sich verlassen können. Ihr gewünschter Status, der in der deklarativen Konfiguration von Kubernetes mit Versionskontrolle definiert ist, wird in Ihrer gesamten Umgebung kontinuierlich durchgesetzt.

Preisgestaltung

Mit EKS-Funktionen gibt es keine Vorabverpflichtungen oder Mindestgebühren. Ihnen wird jede Capability-Ressource für jede Stunde, in der sie in Ihrem Amazon EKS-Cluster aktiv ist, in Rechnung gestellt. Bestimmte Kubernetes-Ressourcen, die von EKS Capabilities verwaltet werden, werden ebenfalls nach einem Stundensatz abgerechnet.

Aktuelle Preisinformationen finden Sie auf der Amazon EKS-Preisseite.

> **Tip**
>
> Sie können den AWS Cost Explorer und die Kosten- und Nutzungsberichte verwenden, um die Kapazitätskosten getrennt von anderen EKS-Gebühren zu verfolgen. Sie können Ihre Kapazitäten mit Clusternamen, Fähigkeitstyp und anderen Details zur Kostenzuweisung kennzeichnen.

So funktionieren die Funktionen von EKS

Jede Funktion ist eine AWS Ressource, die Sie auf Ihrem EKS-Cluster erstellen. Nach der Erstellung wird die Funktion in EKS ausgeführt und vollständig von verwaltet AWS.

> **Note**
>
> Sie können für jeden Typ (Argo CD, ACK und Kro) eine Capability-Ressource für einen bestimmten Cluster erstellen. Sie können nicht mehrere Capability-Ressourcen desselben Typs auf demselben Cluster erstellen.

Sie interagieren mit den Funktionen in Ihrem Cluster mithilfe von Standard-Kubernetes APIs und Tools:

- Wird verwendet`kubectl`, um benutzerdefinierte Kubernetes-Ressourcen anzuwenden
- Verwenden Sie Git-Repositorys als Informationsquelle für Workflows GitOps

Für einige Funktionen werden zusätzliche Tools unterstützt. Beispiel:

- Verwenden Sie die Argo-CD-CLI, um Repositorys und Cluster in Ihrer Argo-CD-Funktion zu konfigurieren und zu verwalten
- Verwenden Sie die Argo-CD-Benutzeroberfläche, um Anwendungen zu visualisieren und zu verwalten, die von Ihrer Argo-CD-Funktion verwaltet werden

Die Funktionen sind so konzipiert, dass sie zusammenarbeiten, aber unabhängig voneinander und vollständig aktiviert sind. Sie können je nach Bedarf eine, zwei oder alle drei Funktionen aktivieren und Ihre Konfiguration aktualisieren, wenn sich Ihre Anforderungen weiterentwickeln.

Alle EKS-Compute-Typen werden für die Verwendung mit EKS-Funktionen unterstützt. Weitere Informationen finden Sie unter Verwaltung von Rechenleistung.

Informationen zur Sicherheitskonfiguration und Einzelheiten zu IAM-Rollen finden Sie unterthe section called “Überlegungen zu den Funktionen von EKS”. Architekturmuster für mehrere Cluster finden Sie unter. the section called “Überlegungen”

Häufige Anwendungsfälle

GitOps für Anwendungen und Infrastruktur

Verwenden Sie Argo CD, um Anwendungen und Betriebskomponenten bereitzustellen, und ACK, um Clusterkonfigurationen zu verwalten und Infrastruktur bereitzustellen, beide aus Git-Repositorys. Ihr gesamter Stack — Anwendungen, Datenbanken, Speicher und Netzwerke — wird als Code definiert und automatisch bereitgestellt.

Beispiel: Ein Entwicklungsteam überträgt Änderungen an Git. Argo CD stellt die aktualisierte Anwendung bereit und ACK stellt eine neue RDS-Datenbank mit der richtigen Konfiguration bereit. Alle Änderungen sind überprüfbar, reversibel und in allen Umgebungen konsistent.

Plattformentwicklung mit Self-Service

Verwenden Sie kro, um benutzerdefinierte APIs ACK- und Kubernetes-Ressourcen zu erstellen. Plattformteams definieren genehmigte Muster anhand von Leitplanken. Anwendungsteams verwenden einfache, übergeordnete Methoden, APIs um komplette Stacks bereitzustellen.

Beispiel: Ein Plattformteam erstellt eine "WebApplication" API, die einen Deployment-, Service-, Ingress- und S3-Bucket bereitstellt. Entwickler verwenden diese API, ohne die zugrunde liegende Komplexität oder AWS die Berechtigungen verstehen zu müssen.

Anwendungsmanagement mit mehreren Clustern

Verwenden Sie Argo CD, um Anwendungen auf mehreren EKS-Clustern in verschiedenen Regionen oder Konten bereitzustellen. Verwalten Sie alle Bereitstellungen von einer einzigen Argo CD-Instanz aus mit konsistenten Richtlinien und Workflows.

Beispiel: Stellen Sie dieselbe Anwendung für Entwicklungs-, Staging- und Produktionscluster in mehreren Regionen bereit. Argo CD stellt sicher, dass jede Umgebung mit dem entsprechenden Git-Zweig synchron bleibt.

Verwaltung mehrerer Cluster

Verwenden Sie ACK, um EKS-Cluster zu definieren und bereitzustellen, Kro, um Cluster-Konfigurationen an organisatorische Standards anzupassen, und Argo CD, um den Cluster-Lebenszyklus und die Konfiguration zu verwalten. Dies ermöglicht die end-to-end Clusterverwaltung von der Erstellung bis zum laufenden Betrieb.

Beispiel: Definieren Sie EKS-Cluster mithilfe von ACK und Kro zur Bereitstellung und Verwaltung der Cluster-Infrastruktur und definieren Sie organisatorische Standards für Netzwerke, Sicherheitsrichtlinien, Add-Ons und andere Konfigurationen. Verwenden Sie Argo CD, um Cluster, Konfigurationen und Kubernetes-Versionsupdates für Ihre gesamte Flotte zu erstellen und kontinuierlich zu verwalten. Dabei profitieren Sie von konsistenten Standards und automatisiertem Lebenszyklusmanagement.

Migrationen und Modernisierung

Vereinfachen Sie die Migration zu EKS mit nativer Cloud-Ressourcenbereitstellung und Workflows. GitOps Verwenden Sie ACK, um vorhandene AWS Ressourcen zu übernehmen, ohne sie neu zu erstellen, und Argo CD, um Workload-Bereitstellungen von Git aus zu operationalisieren.

Beispiel: Ein Team, das von EC2 zu EKS migriert, übernimmt seine vorhandenen RDS-Datenbanken und S3-Buckets mithilfe von ACK und verwendet dann Argo CD, um containerisierte Anwendungen von Git bereitzustellen. Der Migrationspfad ist klar und die Abläufe sind vom ersten Tag an standardisiert.

Konto und regionales Bootstrapping

Automatisieren Sie die Implementierung der Infrastruktur über Konten und Regionen hinweg, indem Sie Argo CD und ACK zusammen verwenden. Definieren Sie Ihre Infrastruktur als Code in Git und lassen Sie Capabilities die Bereitstellung und Verwaltung übernehmen.

Beispiel: Ein Plattformteam verwaltet Git-Repositorys, in denen StandardkontenkonfigurationenVPCs, IAM-Rollen, RDS-Instanzen und Monitoring-Stacks definiert werden. Argo CD stellt diese Konfigurationen automatisch für neue Konten und Regionen bereit, wodurch Konsistenz gewährleistet und die Zeit für die manuelle Einrichtung von Tagen auf Minuten reduziert wird.

Mit Capability-Ressourcen arbeiten

In diesem Thema werden allgemeine Verfahren zur Verwaltung von Fähigkeitsressourcen für alle Fähigkeitstypen beschrieben.

Ressourcen für EKS-Fähigkeiten

EKS-Funktionen sind AWS Ressourcen, die verwaltete Funktionen in Ihrem Amazon EKS-Cluster ermöglichen. Die Funktionen werden in EKS ausgeführt, sodass Sie keine Controller und andere Betriebskomponenten auf Ihren Worker-Knoten installieren und warten müssen. Funktionen werden für einen bestimmten EKS-Cluster erstellt und bleiben während ihres gesamten Lebenszyklus diesem Cluster zugeordnet.

Jede Capability-Ressource hat:

- Ein eindeutiger Name innerhalb Ihres Clusters
- Ein Fähigkeitstyp (ACK, ARGOCD oder KRO)
- Ein Amazon-Ressourcenname (ARN), der sowohl Name als auch Typ angibt
- Eine IAM-Rolle für Fähigkeiten
- Ein Status, der den aktuellen Status angibt
- Konfiguration, sowohl generisch als auch spezifisch für den Funktionstyp

Den Status der Fähigkeit verstehen

Fähigkeitsressourcen haben einen Status, der ihren aktuellen Status angibt. Sie können den Status und den Zustand der Funktionen in der EKS-Konsole oder über die AWS CLI anzeigen.

Konsole:

1. Öffnen Sie die Amazon EKS-Konsole unter https://console.aws.amazon.com/eks/ home#/clusters.
2. Wählen Sie Ihren Clusternamen aus.
3. Wählen Sie die Registerkarte Funktionen, um den Status aller Funktionen anzuzeigen.
4. Für detaillierte Gesundheitsinformationen wählen Sie die Registerkarte Observability, dann Monitor cluster und dann die Registerkarte Capabilities.

AWS CLI:

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-capability-name
```

Status der Fähigkeiten

ERSTELLUNG: Die Fähigkeit wird eingerichtet. Sie können von der Konsole weg navigieren — die Funktion erstellt weiterhin im Hintergrund.

AKTIV: Die Funktion wird ausgeführt und ist einsatzbereit. Wenn Ressourcen nicht wie erwartet funktionieren, überprüfen Sie den Ressourcenstatus und die IAM-Berechtigungen. Anleitungen finden Sie unter the section called “Problembehebung bei Funktionen”.

AKTUALISIERUNG: Die Konfigurationsänderungen werden übernommen. Warten Sie, bis der Status wieder angezeigt wird`ACTIVE`.

LÖSCHEN: Die Fähigkeit wird aus dem Cluster entfernt.

CREATE_FAILED: Beim Setup ist ein Fehler aufgetreten. Zu den häufigsten Ursachen gehören:

- Die Vertrauensrichtlinie für IAM-Rollen ist falsch oder fehlt
- Die IAM-Rolle ist nicht vorhanden oder es kann nicht darauf zugegriffen werden
- Probleme mit dem Clusterzugriff
- Ungültige Konfigurationsparameter

Spezifische Fehlerdetails finden Sie im Abschnitt zur Funktionsintegrität.

UPDATE_FAILED: Das Konfigurationsupdate ist fehlgeschlagen. Einzelheiten finden Sie im Abschnitt zur Funktionsintegrität und überprüfen Sie die IAM-Berechtigungen.

> **Tip**
>
> Eine ausführliche Anleitung zur Fehlerbehebung finden Sie unter:
>
> - the section called “Problembehebung bei Funktionen”- Allgemeine Problembehebung bei Funktionen
> - the section called “Fehlerbehebung”- ACK-spezifische Probleme
> - the section called “Fehlerbehebung”- CD-spezifische Probleme mit Argo
> - the section called “Fehlerbehebung”- Kro-spezifische Probleme

Fähigkeiten schaffen

Informationen zum Erstellen einer Funktion in Ihrem Cluster finden Sie in den folgenden Themen:

- the section called “ACK-Fähigkeit erstellen”— Erstellen Sie eine ACK-Fähigkeit zur Verwaltung von AWS Ressourcen mithilfe von Kubernetes APIs
- the section called “Erstellen der Argo CD-Funktion”— Erstellen Sie eine Argo-CD-Funktion für kontinuierliche Bereitstellung GitOps
- the section called “Erstellen Sie eine Kro-Fähigkeit”— Schaffen Sie eine Kro-Funktion für die Zusammenstellung und Orchestrierung von Ressourcen

Funktionen auflisten

Sie können alle Capability-Ressourcen in einem Cluster auflisten.

Konsole

1. Öffnen Sie die Amazon EKS-Konsole unter https://console.aws.amazon.com/eks/ home#/clusters.
2. Wählen Sie Ihren Clusternamen aus, um die Cluster-Detailseite zu öffnen.
3. Wählen Sie die Registerkarte Funktionen.
4. Sehen Sie sich die Ressourcen zu den Funktionen unter Verwaltete Funktionen an.

AWS CLI

Verwenden Sie den `list-capabilities` Befehl, um alle Funktionen Ihres Clusters anzuzeigen. *`region-code`*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet, und *`my-cluster`* ersetzen Sie es durch den Namen Ihres Clusters.

```
aws eks list-capabilities \
  --region region-code \
  --cluster-name my-cluster
```

```
{
    "capabilities": [
        {
            "capabilityName": "my-ack",
            "arn": "arn:aws:eks:us-west-2:111122223333:capability/my-cluster/ack/my-
ack/abc123",
            "type": "ACK",
            "status": "ACTIVE",
            "createdAt": "2025-11-02T10:30:00.000000-07:00",
            "modifiedAt": "2025-11-02T10:32:15.000000-07:00",
        },
        {
            "capabilityName": "my-kro",
            "arn": "arn:aws:eks:us-west-2:111122223333:capability/my-cluster/kro/my-
kro/abc123",
            "type": "KRO",
            "status": "ACTIVE",
            "version": "v0.6.3",
            "createdAt": "2025-11-02T10:30:00.000000-07:00",
            "modifiedAt": "2025-11-02T10:32:15.000000-07:00",
        },
        {
            "capabilityName": "my-argocd",
            "arn": "arn:aws:eks:us-west-2:111122223333:capability/my-cluster/argocd/my-
argocd/abc123",
            "type": "ARGOCD",
            "status": "ACTIVE",
            "version": "3.1.8-eks-1",
            "createdAt": "2025-11-21T08:22:28.486000-05:00",
            "modifiedAt": "2025-11-21T08:22:28.486000-05:00"
        }
```

```
    ]
}
```

Beschreiben Sie eine Fähigkeit

Erhalten Sie detaillierte Informationen zu einer bestimmten Funktion, einschließlich ihrer Konfiguration und ihres Status.

Konsole

1. Öffnen Sie die Amazon EKS-Konsole unter https://console.aws.amazon.com/eks/ home#/clusters.
2. Wählen Sie Ihren Clusternamen aus, um die Cluster-Detailseite zu öffnen.
3. Wählen Sie die Registerkarte Funktionen.
4. Wählen Sie unter Verwaltete Funktionen die Funktion aus, die Sie anzeigen möchten.
5. Sehen Sie sich die Funktionsdetails an, einschließlich Status, Konfiguration und Erstellungszeit.

AWS CLI

Verwenden Sie den `describe-capability` Befehl, um detaillierte Informationen anzuzeigen. *`region-code`*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet, *`my-cluster`* ersetzen Sie es durch den Namen Ihres Clusters und *`capability-name`* ersetzen Sie es durch den Namen der Fähigkeit (ack, argocd oder kro).

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name capability-name
```

Beispielausgabe:

```
{
  "capability": {
    "capabilityName": "my-ack",
    "capabilityArn": "arn:aws:eks:us-west-2:111122223333:capability/my-cluster/ack/my-
ack/abc123",
    "clusterName": "my-cluster",
```

```
    "type": "ACK",
    "roleArn": "arn:aws:iam::111122223333:role/AmazonEKSCapabilityACKRole",
    "status": "ACTIVE",
    "configuration": {},
    "tags": {},
    "health": {
      "issues": []
    },
    "createdAt": "2025-11-19T17:11:30.242000-05:00",
    "modifiedAt": "2025-11-19T17:11:30.242000-05:00",
    "deletePropagationPolicy": "RETAIN"
  }
}
```

Aktualisieren Sie die Konfiguration einer Funktion

Sie können bestimmte Aspekte der Konfiguration einer Funktion nach der Erstellung aktualisieren. Die spezifischen Konfigurationsoptionen variieren je nach Funktionstyp.

> **Note**
>
> Die Ressourcen der EKS-Funktionen werden vollständig verwaltet, einschließlich Patches und Versionsupdates. Die Aktualisierung einer Funktion aktualisiert die Ressourcenkonfiguration und führt nicht zu Versionsupdates der verwalteten Funktionskomponenten.

AWS CLI

Verwenden Sie den `update-capability` Befehl, um eine Funktion zu ändern:

```
aws eks update-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name capability-name \
  --role-arn arn:aws:iam::[.replaceable]111122223333:role/NewCapabilityRole
```

> **Note**
>
> Nicht alle Funktionseigenschaften können nach der Erstellung aktualisiert werden. Einzelheiten dazu, was geändert werden kann, finden Sie in der funktionsspezifischen Dokumentation.

Löschen Sie eine Funktion

Wenn Sie eine Funktion in Ihrem Cluster nicht mehr benötigen, können Sie die Fähigkeitsressource löschen.

> **Important**
>
> Löschen Sie die Clusterressourcen, bevor Sie die Funktion löschen.
> Durch das Löschen einer Funktionsressource werden nicht automatisch Ressourcen gelöscht, die mit dieser Fähigkeit erstellt wurden:
>
> - Alle benutzerdefinierten Kubernetes-Ressourcendefinitionen (CRDs) bleiben in Ihrem Cluster installiert.
> - ACK-Ressourcen verbleiben in Ihrem Cluster und die entsprechenden AWS Ressourcen verbleiben in Ihrem Konto
> - Argo CD-Anwendungen und ihre Kubernetes-Ressourcen verbleiben in Ihrem Cluster
> - Kro ResourceGraphDefinitions und Instanzen verbleiben in Ihrem Cluster
>
> Sie sollten diese Ressourcen löschen, bevor Sie die Funktion löschen, um verwaiste Ressourcen zu vermeiden.
> Sie können sich optional dafür entscheiden, AWS Ressourcen beizubehalten, die mit ACK Kubernetes-Ressourcen verknüpft sind. Siehe Überlegungen zu ACK

Konsole

1. Öffnen Sie die Amazon EKS-Konsole unter https://console.aws.amazon.com/eks/ home#/clusters.
2. Wählen Sie Ihren Clusternamen aus, um die Cluster-Detailseite zu öffnen.
3. Wählen Sie die Registerkarte Funktionen.

4. Wählen Sie die Funktion, die Sie löschen möchten, aus der Liste der verwalteten Funktionen aus.
5. Wählen Sie „Funktion löschen“.
6. Geben Sie im Bestätigungsdialogfeld den Namen der Funktion ein, um das Löschen zu bestätigen.
7. Wählen Sie Löschen aus.

AWS CLI

Verwenden Sie den `delete-capability` Befehl, um eine Capability-Ressource zu löschen:

*region-code*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet, *my-cluster* ersetzen Sie es durch den Namen Ihres Clusters und *capability-name* ersetzen Sie es durch den Namen der zu löschenden Funktion.

```
aws eks delete-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name capability-name
```

Nächste Schritte

- the section called “Kubernetes-Ressourcen”— Erfahren Sie mehr über die Kubernetes-Ressourcen, die von den einzelnen Funktionstypen bereitgestellt werden
- the section called “ACK-Konzepte”— Verstehen Sie die ACK-Konzepte und den Ressourcenlebenszyklus
- the section called “Arbeiten mit Argo CD”— Arbeiten mit den Funktionen von Argo CD für Workflows GitOps
- the section called “Kro-Konzepte”— Verstehen Sie die KRO-Konzepte und die Zusammensetzung der Ressourcen

Fähigkeiten — Kubernetes-Ressourcen

Nachdem Sie eine Funktion in Ihrem Cluster aktiviert haben, interagieren Sie meistens mit ihr, indem Sie benutzerdefinierte Kubernetes-Ressourcen in Ihrem Cluster erstellen und verwalten. Jede Funktion bietet ihre eigenen benutzerdefinierten Ressourcendefinitionen (CRDs), die die Kubernetes-API um funktionsspezifische Funktionen erweitern.

Ressourcen von Argo CD

Wenn Sie die Argo-CD-Funktion aktivieren, können Sie die folgenden Kubernetes-Ressourcen erstellen und verwalten:

Anwendung

Definiert eine Bereitstellung von einem Git-Repository in einem Zielcluster. `Application`Ressourcen spezifizieren das Quell-Repository, den Ziel-Namespace und die Synchronisierungsrichtlinie. Sie können bis zu 1000 `Application` Ressourcen pro Argo-CD-Capability-Instance erstellen.

ApplicationSet

Generiert mehrere `Application` Ressourcen aus Vorlagen und ermöglicht so Bereitstellungen in mehreren Clustern und Umgebungen. `ApplicationSet`Ressourcen verwenden Generatoren, um `Application` Ressourcen dynamisch auf der Grundlage von Clusterlisten, Git-Verzeichnissen oder anderen Quellen zu erstellen.

AppProject

Bietet logische Gruppierung und Zugriffskontrolle für `Application` Ressourcen. `AppProject`Ressourcen definieren, welche Repositorys, Cluster und Namespaces `Application` Ressourcen verwenden können, wodurch Mehrmandantenfähigkeit und Sicherheitsgrenzen ermöglicht werden.

Beispielressource: `Application`

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: my-app
  namespace: argocd
spec:
  project: default
  source:
    repoURL: https://github.com/org/repo
    targetRevision: main
    path: manifests
  destination:
    server: https://kubernetes.default.svc
```

```
    namespace: production
```

Weitere Informationen zu den Ressourcen und Konzepten von Argo CD finden Sie unterthe section called “Argo-CD-Konzepte”.

Kro-Ressourcen

Wenn Sie die Kro-Funktion aktivieren, können Sie die folgenden Kubernetes-Ressourcen erstellen und verwalten:

ResourceGraphDefinition (RGD)

Definiert eine benutzerdefinierte API, die mehrere Kubernetes und AWS Ressourcen zu einer Abstraktion auf höherer Ebene zusammenfasst. Plattformteams erstellen `ResourceGraphDefinition` Ressourcen, um wiederverwendbare Muster mit Leitplanken bereitzustellen.

Benutzerdefinierte Ressourceninstanzen

Nachdem Sie eine `ResourceGraphDefinition` Ressource erstellt haben, können Sie Instanzen der von der definierten benutzerdefinierten API erstellen`ResourceGraphDefinition`. kro erstellt und verwaltet automatisch die in der `ResourceGraphDefinition` angegebenen Ressourcen.

`ResourceGraphDefinition`Beispielressource:

```
apiVersion: kro.run/v1alpha1
kind: ResourceGraphDefinition
metadata:
  name: web-application
spec:
  schema:
    apiVersion: v1alpha1
    kind: WebApplication
    spec:
      name: string
      replicas: integer
  resources:
    - id: deployment
      template:
        apiVersion: apps/v1
        kind: Deployment
```

```
        # ... deployment spec
    - id: service
      template:
        apiVersion: v1
        kind: Service
        # ... service spec
```

`WebApplication`Beispielinstanz:

```
apiVersion: v1alpha1
kind: WebApplication
metadata:
  name: my-web-app
  namespace: default
spec:
  name: my-web-app
  replicas: 3
```

Wenn Sie diese Instanz anwenden, erstellt kro automatisch die `Deployment` in der `ResourceGraphDefinition` definierten `Service` Ressourcen.

Weitere Informationen zu den Ressourcen und Konzepten von Kro finden Sie unter. the section called “Kro-Konzepte”

ACK-Ressourcen

Wenn Sie die ACK-Funktion aktivieren, können Sie AWS Ressourcen mithilfe von benutzerdefinierten Kubernetes-Ressourcen erstellen und verwalten. ACK bietet über 200 Dienste CRDs für mehr als 50 AWS Dienste, sodass Sie AWS Ressourcen neben Ihren Kubernetes-Workloads definieren und dedizierte AWS Infrastrukturressourcen mit Kubernetes verwalten können.

Beispiele für ACK-Ressourcen:

S3 Bucket (S3-Bucket)

> `Bucket`Ressourcen erstellen und verwalten Amazon S3 S3-Buckets mit Versionierungs-, Verschlüsselungs- und Lebenszyklusrichtlinien.

RDS DBInstance

> `DBInstance`Bereitstellung und Verwaltung von Amazon RDS-Datenbank-Instances mit automatisierten Backups und Wartungsfenstern.

DynamoDB-Tabelle

`Table`Ressourcen erstellen und verwalten DynamoDB-Tabellen mit bereitgestellter oder bedarfsgesteuerter Kapazität.

IAM Role (IAM-Rolle)

`Role`Ressourcen definieren IAM-Rollen mit Vertrauensrichtlinien und Berechtigungsrichtlinien für den Servicezugriff. AWS

Lambda-Funktion

`Function`Ressourcen erstellen und verwalten Lambda-Funktionen mit Code-, Laufzeit- und Ausführungsrollenkonfiguration.

Beispiel für die Spezifikation einer `Bucket` Ressource:

```
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: my-app-bucket
spec:
  name: my-unique-bucket-name-12345
  versioning:
    status: Enabled
  encryption:
    rules:
      - applyServerSideEncryptionByDefault:
          sseAlgorithm: AES256
```

Weitere Informationen zu ACK-Ressourcen und Konzepten finden Sie unterthe section called “ACK-Konzepte”.

Ressourcenlimits

Für EKS-Funktionen gelten die folgenden Ressourcengrenzen:

Nutzungsbeschränkungen für Argo CD:

- Maximal 1000 `Application` Ressourcen pro Argo-CD-Capability-Instanz
- Pro Argo-CD-Capability-Instanz sind maximal 100 Remote-Cluster konfiguriert

Grenzwerte für die Ressourcenkonfiguration:

- Maximal 64 Kubernetes-Ressourcen pro `Application` Ressource in Argo CD
- Maximal 64 Kubernetes-Ressourcen pro in Kro `ResourceGraphDefinition`

> **Note**
>
> Diese Grenzwerte gelten für die Anzahl der Ressourcen, die von jeder Capability-Instanz verwaltet werden. Wenn Sie höhere Grenzwerte benötigen, können Sie Funktionen auf mehreren Clustern bereitstellen.

Nächste Schritte

Informationen zu funktionsspezifischen Aufgaben und zur erweiterten Konfiguration finden Sie in den folgenden Themen:

- the section called “ACK-Konzepte”— Verstehen Sie die ACK-Konzepte und den Ressourcenlebenszyklus
- the section called “Arbeiten mit Argo CD”— Arbeiten mit den Funktionen von Argo CD für Workflows GitOps
- the section called “Kro-Konzepte”— Verstehen Sie die KRO-Konzepte und die Zusammensetzung der Ressourcen

Überlegungen zu den EKS-Fähigkeiten

In diesem Thema werden wichtige Überlegungen zur Nutzung von EKS-Funktionen behandelt, darunter das Design der Zugriffskontrolle, die Wahl zwischen EKS-Funktionen und selbstverwalteten Lösungen, Architekturmuster für Bereitstellungen mit mehreren Clustern und bewährte Verfahren für den Betrieb.

Capability: IAM-Rollen und Kubernetes RBAC

Jede EKS-Capability-Ressource hat eine konfigurierte Capability-IAM-Rolle. Die Capability-Rolle wird verwendet, um AWS Dienstberechtigungen für EKS-Funktionen zu erteilen, damit sie in Ihrem Namen agieren können. Um beispielsweise die EKS-Funktion für ACK zur Verwaltung von Amazon S3 S3-

Buckets zu verwenden, gewähren Sie S3 Bucket Administratorberechtigungen für die Funktion, sodass sie Buckets erstellen und verwalten kann.

Sobald die Funktion konfiguriert ist, können S3-Ressourcen in AWS mit benutzerdefinierten Kubernetes-Ressourcen in Ihrem Cluster erstellt und verwaltet werden. Kubernetes RBAC ist der Cluster-interne Zugriffskontrollmechanismus, mit dem bestimmt wird, welche Benutzer und Gruppen diese benutzerdefinierten Ressourcen erstellen und verwalten können. Erteilen Sie beispielsweise bestimmten Kubernetes RBAC-Benutzern und -Gruppen Berechtigungen zum Erstellen und Verwalten von Ressourcen in Namespaces Ihrer Wahl. `Bucket`

Auf diese Weise sind IAM und Kubernetes RBAC zwei Hälften des Zugriffskontrollsystems, das die Berechtigungen im Zusammenhang mit den end-to-end Funktionen und Ressourcen von EKS regelt. Es ist wichtig, die richtige Kombination aus IAM-Berechtigungen und RBAC-Zugriffsrichtlinien für Ihren Anwendungsfall zu entwerfen.

Weitere Informationen zu Capability IAM-Rollen und Kubernetes-Berechtigungen finden Sie unter. the section called “Überlegungen zu den Funktionen von EKS”

Architekturmuster mit mehreren Clustern

Bei der Bereitstellung von Funktionen in mehreren Clustern sollten Sie die folgenden gängigen Architekturmuster berücksichtigen:

Hub and Spoke mit zentralisierter Verwaltung

Führen Sie alle drei Funktionen in einem zentral verwalteten Cluster aus, um Workloads zu orchestrieren und die Cloud-Infrastruktur über mehrere Workload-Cluster hinweg zu verwalten.

- Argo CD auf dem Management-Cluster stellt Anwendungen für Workload-Cluster in verschiedenen Regionen oder Konten bereit
- ACK auf dem Management-Cluster stellt AWS Ressourcen (RDS, S3, IAM) für alle Cluster bereit
- kro auf dem Management-Cluster erstellt tragbare Plattformabstraktionen, die in allen Clustern funktionieren

Dieses Muster zentralisiert das Workload- und Cloud-Infrastrukturmanagement und kann den Betrieb für Unternehmen, die viele Cluster verwalten, vereinfachen.

Dezentralisiert GitOps

Workloads und Cloud-Infrastruktur werden über Funktionen auf demselben Cluster verwaltet, auf dem die Workloads ausgeführt werden.

- Argo CD verwaltet Anwendungsressourcen auf dem lokalen Cluster.
- ACK-Ressourcen werden für Cluster- und Workload-Anforderungen verwendet.
- Die Abstraktionen der Kro-Plattform werden installiert und orchestrieren lokale Ressourcen.

Dieses Muster dezentralisiert den Betrieb, wobei die Teams ihre eigenen dedizierten Plattformdienste in einem oder mehreren Clustern verwalten.

Hub-and-Spoke-Lösung mit hybrider ACK-Bereitstellung

Kombinieren Sie zentralisierte und dezentrale Modelle mit zentralisierter Anwendungsbereitstellung und Ressourcenverwaltung auf der Grundlage von Umfang und Eigentümerschaft.

- Hub-Cluster:
  - Argo CD verwaltet GitOps Bereitstellungen auf dem lokalen Cluster und allen Remote-Workload-Clustern
  - ACK wird auf dem Management-Cluster für Ressourcen im Administratorbereich (Produktionsdatenbanken, IAM-Rollen,) verwendet VPCs
  - kro wird auf dem Management-Cluster für wiederverwendbare Plattformabstraktionen verwendet
- Spoke-Cluster:
  - Workloads werden über Argo CD auf dem zentralisierten Hub-Cluster verwaltet
  - ACK wird lokal für Ressourcen im Workloadbereich (S3-Buckets, Instanzen, SQS-Warteschlangen) verwendet ElastiCache
  - kro wird lokal für die Zusammenstellung von Ressourcen und für Bausteinmuster verwendet

Dieses Muster unterscheidet die einzelnen Anliegen: Plattformteams verwalten kritische Infrastrukturen zentral auf Managementclustern, optional einschließlich Workload-Clustern, während Anwendungsteams Cloud-Ressourcen zusammen mit Workloads spezifizieren und verwalten.

Ein Muster auswählen

Berücksichtigen Sie bei der Auswahl einer Architektur die folgenden Faktoren:

- Organisationsstruktur: Zentralisierte Plattformteams bevorzugen Hub-Modelle; dezentrale Teams bevorzugen möglicherweise Funktionen pro Cluster

- Umfang der Ressourcen: Ressourcen mit Administratorrechten (Datenbanken, IAM) profitieren häufig von einer zentralen Verwaltung; Workload-Ressourcen (Buckets, Warteschlangen) können lokal verwaltet werden
- Self-Service: Zentralisierte Plattformteams können präskriptive benutzerdefinierte Ressourcen erstellen und verteilen, um Cloud-Ressourcen für allgemeine Workload-Anforderungen sicher selbst zu verwalten
- Cluster-Flottenmanagement: Zentralisierte Management-Cluster bieten eine kundeneigene Steuerungsebene für das EKS-Cluster-Flottenmanagement, zusammen mit anderen administrativen Ressourcen
- Compliance-Anforderungen: Einige Unternehmen benötigen eine zentrale Steuerung für Audit und Governance
- Operative Komplexität: Weniger Funktionsinstanzen vereinfachen den Betrieb, können aber zu Engpässen führen

> **Note**
>
> Sie können mit einem Muster beginnen und sich mit zunehmender Reife Ihrer Plattform zu einem anderen weiterentwickeln. Die Funktionen sind unabhängig — Sie können sie je nach Bedarf in den Clustern unterschiedlich einsetzen.

Vergleich der EKS-Funktionen mit selbstverwalteten Lösungen

EKS-Funktionen bieten vollständig verwaltete Funktionen für beliebte Kubernetes-Tools und -Controller, die in EKS ausgeführt werden. Dies unterscheidet sich von selbstverwalteten Lösungen, die Sie in Ihrem Cluster installieren und betreiben.

Die wichtigsten Unterschiede

Bereitstellung und Verwaltung

AWS verwaltet die EKS-Funktionen vollständig, ohne dass eine Installation, Konfiguration oder Wartung der Komponentensoftware erforderlich ist. AWS installiert und verwaltet automatisch alle erforderlichen benutzerdefinierten Kubernetes-Ressourcendefinitionen (CRDs) im Cluster.

Bei selbstverwalteten Lösungen installieren und konfigurieren Sie Clustersoftware mithilfe von Helm Charts, Kubectl oder anderen Operatoren. Sie haben die volle Kontrolle über den

Softwarelebenszyklus und die Laufzeitkonfiguration Ihrer selbstverwalteten Lösungen und können Anpassungen auf jeder Ebene der Lösung vornehmen.

Betrieb und Wartung

AWS verwaltet Patching- und andere Softwarelebenszyklusvorgänge für EKS-Funktionen mit automatischen Updates und Sicherheitspatches. Die Funktionen von EKS sind in AWS Funktionen für optimierte Konfigurationen integriert, bieten integrierte Hochverfügbarkeit und Fehlertoleranz und machen die Behebung von Controller-Workloads im Cluster überflüssig.

Bei selbstverwalteten Lösungen müssen Sie den Zustand und die Protokolle der Komponenten überwachen, Sicherheitspatches und Versionsupdates anwenden, Hochverfügbarkeit mit mehreren Replikaten und Budgets für Pod-Unterbrechungen konfigurieren, Probleme mit der Controller-Arbeitslast beheben und beheben sowie Releases und Versionen verwalten. Sie haben die volle Kontrolle über Ihre Bereitstellungen, aber dafür sind häufig maßgeschneiderte Lösungen für den privaten Clusterzugriff und andere Integrationen erforderlich, die den Unternehmensstandards und den Sicherheitsbestimmungen entsprechen müssen.

Ressourcenverbrauch

EKS-Funktionen werden in EKS und außerhalb Ihrer Cluster ausgeführt, wodurch Knoten- und Clusterressourcen freigesetzt werden. Capabilities nutzen keine Cluster-Workload-Ressourcen, verbrauchen weder CPU noch Arbeitsspeicher auf Ihren Worker-Knoten, skalieren automatisch und haben nur minimale Auswirkungen auf die Cluster-Kapazitätsplanung.

Selbstverwaltete Lösungen führen Controller und andere Komponenten auf Ihren Worker-Knoten aus und verbrauchen dabei direkt Worker-Knotenressourcen IPs, Cluster und andere Clusterressourcen. Die Verwaltung von Clusterdiensten erfordert eine Kapazitätsplanung für deren Workloads sowie die Planung und Konfiguration von Ressourcenanforderungen und -limits, um Skalierungs- und Hochverfügbarkeitsanforderungen zu bewältigen.

Unterstützung von Funktionen

Als vollständig verwaltete Servicefunktionen sind EKS Capabilities von Natur aus eigensinnig im Vergleich zu selbstverwalteten Lösungen. Die Funktionen werden zwar die meisten Funktionen und Anwendungsfälle unterstützen, allerdings wird es im Vergleich zu selbstverwalteten Lösungen einen Unterschied in der Abdeckung geben.

Bei selbstverwalteten Lösungen haben Sie die vollständige Kontrolle über die Konfiguration, optionale Funktionen und andere Aspekte der Funktionalität Ihrer Software. Sie können sich dafür entscheiden,

Ihre eigenen benutzerdefinierten Images auszuführen, alle Aspekte der Konfiguration anzupassen und die Funktionen Ihrer selbstverwalteten Lösung vollständig zu kontrollieren.

Überlegungen zu den Kosten

Für jede EKS-Funktionsressource fallen Kosten pro Stunde an, die je nach Fähigkeitstyp unterschiedlich sind. Für Clusterressourcen, die von der Funktion verwaltet werden, fallen auch stündliche Kosten mit eigenen Preisen an. Weitere Informationen finden Sie unter Amazon EKS – Preise.

Bei selbstverwalteten Lösungen fallen keine direkten Kosten im Zusammenhang mit AWS Gebühren an. Sie zahlen jedoch für Cluster-Rechenressourcen, die von Controllern genutzt werden, und für die damit verbundenen Workloads. Neben dem Ressourcenverbrauch von Knoten und Clustern umfassen die Gesamtbetriebskosten bei selbstverwalteten Lösungen auch die Betriebskosten sowie die Kosten für Wartung, Fehlerbehebung und Support.

Sie haben die Wahl zwischen EKS-Funktionen und selbstverwalteten Lösungen

EKS-Funktionen Ziehen Sie diese Wahl in Betracht, wenn Sie den betrieblichen Aufwand reduzieren und sich auf den Mehrwert Ihrer Software und Systeme konzentrieren möchten, anstatt den Betrieb der Clusterplattform für grundlegende Anforderungen zu bündeln. Verwenden Sie EKS Capabilities, wenn Sie den betrieblichen Aufwand durch Sicherheitspatches und Software-Lebenszyklusmanagement minimieren, Knoten- und Clusterressourcen für Anwendungsworkloads freigeben, die Konfiguration und das Sicherheitsmanagement vereinfachen und vom AWS Support profitieren möchten. Die EKS-Funktionen eignen sich ideal für die meisten Anwendungsfälle in der Produktion und sind der empfohlene Ansatz für neue Implementierungen.

Selbstverwaltete Lösungen Ziehen Sie diese Wahl in Betracht, wenn Sie bestimmte Versionen der Kubernetes-Ressourcen-API oder benutzerdefinierte Controller-Builds benötigen, bereits vorhandene Automatisierungs- und Tools auf selbstverwalteten Bereitstellungen aufbauen möchten oder wenn Sie umfassende Anpassungen der Controller-Laufzeitkonfigurationen benötigen. Selbstverwaltete Lösungen bieten Flexibilität für spezielle Anwendungsfälle, und Sie haben die vollständige Kontrolle über Ihre Bereitstellung und Laufzeitkonfiguration.

> **Note**
>
> EKS-Funktionen können in Ihrem Cluster mit selbstverwalteten Lösungen koexistieren, und schrittweise Migrationen sind möglich.

Fähigkeitsspezifische Vergleiche

Detaillierte Vergleiche, einschließlich funktionsspezifischer Funktionen, Upstream-Unterschiede und Migrationspfade, finden Sie unter:

- the section called “Vergleich mit selbstverwaltetem System”
- the section called “Vergleich mit selbstverwalteter Software”
- the section called “Vergleich mit selbstverwaltetem System”

Stellen Sie AWS Ressourcen aus Kubernetes mit AWS Controllers for Kubernetes (ACK) bereit

AWS Mit Controllers for Kubernetes (ACK) können Sie Serviceressourcen direkt von Kubernetes aus definieren und verwalten. AWS Mit AWS Controllers for Kubernetes (ACK) können Sie Workload-Ressourcen und Cloud-Infrastruktur mithilfe von benutzerdefinierten Kubernetes-Ressourcen direkt neben Ihren Anwendungs-Workloads mit vertrauten Kubernetes und Tools verwalten. APIs

Mit EKS-Funktionen wird ACK vollständig verwaltet AWS, sodass Sie keine ACK-Controller auf Ihren Clustern installieren, warten und skalieren müssen.

Wie funktioniert ACK

ACK übersetzt benutzerdefinierte Kubernetes-Ressourcenspezifikationen in AWS API-Aufrufe. Wenn Sie eine benutzerdefinierte Kubernetes-Ressource, die eine AWS Dienstressource darstellt, erstellen, aktualisieren oder löschen, führt ACK die erforderlichen AWS API-Aufrufe durch, um die Ressource zu erstellen, zu aktualisieren oder zu löschen. AWS

Jede von ACK unterstützte AWS Ressource hat ihre eigene benutzerdefinierte Ressourcendefinition (CRD), die das Kubernetes-API-Schema für die Angabe ihrer Konfiguration definiert. ACK bietet CRDs beispielsweise S3 einschließlich Buckets, Bucket-Richtlinien und andere S3-Ressourcen.

ACK gleicht den Status Ihrer AWS Ressourcen kontinuierlich mit dem gewünschten Status ab, der in Ihren benutzerdefinierten Kubernetes-Ressourcen definiert ist. Wenn eine Ressource von ihrem gewünschten Zustand abweicht, erkennt ACK dies und ergreift Korrekturmaßnahmen, um sie wieder in Einklang zu bringen. Änderungen an Kubernetes-Ressourcen wirken sich sofort auf den AWS Ressourcenstatus aus, während die passive Drift-Erkennung und Behebung von AWS Upstream-Ressourcenänderungen bis zu 10 Stunden dauern kann (die Resynchronisierungszeit), in der Regel jedoch viel früher erfolgt.

Beispiel für ein S3-Bucket-Ressourcenmanifest

```
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: my-ack-bucket
spec:
  name: my-unique-bucket-name
```

Wenn Sie diese benutzerdefinierte Ressource auf Ihren Cluster anwenden, erstellt ACK einen Amazon S3 S3-Bucket in Ihrem Konto, falls dieser noch nicht vorhanden ist. Spätere Änderungen an dieser Ressource, z. B. die Angabe einer nicht standardmäßigen Speicherstufe oder das Hinzufügen einer Richtlinie, werden auf die S3-Ressource in AWS angewendet. Wenn diese Ressource aus dem Cluster gelöscht wird, wird der AWS darin enthaltene S3-Bucket standardmäßig gelöscht.

Vorteile von ACK

ACK bietet Kubernetes-natives AWS Ressourcenmanagement, sodass Sie AWS Ressourcen mit demselben Kubernetes APIs und denselben Tools verwalten können, die Sie für Ihre Anwendungen verwenden. Dieser einheitliche Ansatz vereinfacht Ihren Infrastrukturmanagement-Workflow, da Sie nicht mehr zwischen verschiedenen Tools wechseln oder separate Systeme erlernen müssen. infrastructure-as-code Sie definieren Ihre AWS Ressourcen deklarativ in Kubernetes-Manifesten und ermöglichen so GitOps Workflows und Infrastruktur als Code-Praktiken, die sich nahtlos in Ihre bestehenden Entwicklungsprozesse integrieren lassen.

ACK gleicht kontinuierlich den gewünschten Zustand Ihrer AWS Ressourcen mit ihrem tatsächlichen Zustand ab, korrigiert Abweichungen und sorgt für Konsistenz in Ihrer gesamten Infrastruktur. Dieser kontinuierliche Abgleich bedeutet, dass wichtige out-of-band Änderungen an AWS Ressourcen automatisch rückgängig gemacht werden, sodass sie Ihrer deklarierten Konfiguration entsprechen, sodass die Integrität Ihrer Infrastruktur als Code gewahrt bleibt. Sie können ACK so konfigurieren, dass Ressourcen über mehrere AWS Konten und Regionen hinweg verwaltet werden, sodass komplexe Architekturen mit mehreren Konten ohne zusätzliche Tools ermöglicht werden.

Für Unternehmen, die von anderen Infrastrukturmanagement-Tools migrieren, unterstützt ACK die Einführung von Ressourcen, sodass Sie vorhandene AWS Ressourcen unter das ACK-Management stellen können, ohne sie neu erstellen zu müssen. ACK bietet außerdem schreibgeschützte Ressourcen für die AWS Ressourcenbeobachtung ohne Änderungszugriff sowie Anmerkungen, um

AWS Ressourcen optional auch dann beizubehalten, wenn die Kubernetes-Ressource aus dem Cluster gelöscht wird.

Weitere Informationen und erste Schritte mit der EKS-Funktion für ACK finden Sie unter und. the section called “ACK-Konzepte” the section called “Überlegungen”

Unterstützte AWS Dienste

ACK unterstützt eine breite Palette von AWS Diensten, einschließlich, aber nicht beschränkt auf:

- Amazon EC2
- Amazon S3
- Amazon RDS
- Amazon DynamoDB
- Amazon ElastiCache
- Amazon EKS
- Amazon SQS
- Amazon SNS
- AWS Lambda
- AWS ICH BIN

Alle AWS Dienste, die im Upstream-Bereich als allgemein verfügbar aufgeführt sind, werden von der EKS-Funktion für ACK unterstützt. Einzelheiten finden Sie in der vollständigen Liste der unterstützten AWS Dienste.

Integration mit anderen von EKS verwalteten Funktionen

ACK lässt sich in andere von EKS verwaltete Funktionen integrieren.

- Argo CD: Verwenden Sie Argo CD, um die Bereitstellung von ACK-Ressourcen in mehreren Clustern zu verwalten und so GitOps Workflows für Ihre AWS Infrastruktur zu ermöglichen.
  - ACK erweitert die Vorteile von GitOps , wenn es mit ArgoCD kombiniert wird, aber ACK erfordert keine Integration mit Git.
- kro (Kube Resource Orchestrator): Verwenden Sie Kro, um komplexe Ressourcen aus ACK-Ressourcen zusammenzustellen und so Abstraktionen auf höherer Ebene zu erstellen, die das Ressourcenmanagement vereinfachen.

- Sie können mit kro zusammengesetzte benutzerdefinierte Ressourcen erstellen, die sowohl Kubernetes-Ressourcen als auch Ressourcen definieren. AWS Teammitglieder können diese benutzerdefinierten Ressourcen verwenden, um komplexe Anwendungen schnell bereitzustellen.

Erste Schritte mit ACK

Erste Schritte mit der EKS-Funktion für ACK:

1. Erstellen und konfigurieren Sie eine IAM-Capability-Rolle mit den erforderlichen Berechtigungen, damit ACK AWS Ressourcen in Ihrem Namen verwalten kann.
2. Erstellen Sie eine ACK-Fähigkeitsressource auf Ihrem EKS-Cluster über die AWS Konsole, AWS CLI oder Ihr bevorzugtes Infrastructure-as-Code-Tool.
3. Wenden Sie benutzerdefinierte Kubernetes-Ressourcen auf Ihren Cluster an, um mit der Verwaltung Ihrer AWS Ressourcen in Kubernetes zu beginnen.

Erstellen Sie eine ACK-Fähigkeit

In diesem Kapitel wird erklärt, wie Sie eine ACK-Fähigkeit in Ihrem Amazon EKS-Cluster erstellen.

Voraussetzungen

Bevor Sie eine ACK-Fähigkeit erstellen, stellen Sie sicher, dass Sie über Folgendes verfügen:

- Ein Amazon-EKS-Cluster
- Eine IAM-Fähigkeitsrolle mit Berechtigungen für ACK zur Verwaltung von Ressourcen AWS
- Ausreichende IAM-Berechtigungen zum Erstellen von Funktionsressourcen auf EKS-Clustern
- Das entsprechende installierte und konfigurierte CLI-Tool oder Zugriff auf die EKS-Konsole

Anweisungen zur Erstellung der IAM-Capability-Rolle finden Sie unterthe section called “Funktion: IAM-Rolle”.

> **Important**
>
> ACK ist eine Infrastrukturverwaltungsfunktion, die das Erstellen, Ändern und Löschen von AWS Ressourcen ermöglicht. Dabei handelt es sich um eine Funktion für Administratoren, die sorgfältig kontrolliert werden sollte. Jeder, der die Erlaubnis hat, Kubernetes-Ressourcen

in Ihrem Cluster zu erstellen, kann Ressourcen effektiv über ACK erstellen, sofern die Berechtigungen für die AWS IAM-Capability Role erfüllt sind. Die von Ihnen bereitgestellte IAM-Capability-Rolle bestimmt, welche AWS Ressourcen ACK erstellen und verwalten kann. Anleitungen zum Erstellen einer geeigneten Rolle mit den geringsten Rechten finden Sie unter und. the section called “Funktion: IAM-Rolle” the section called “Überlegungen zu den Funktionen von EKS”

Wählen Sie Ihr Tool

Sie können eine ACK-Fähigkeit mit der AWS-Managementkonsole AWS CLI oder eksctl erstellen:

- the section called “Konsole”- Verwenden Sie die Konsole für ein geführtes Erlebnis
- the section called “ AWS CLI”- Verwenden Sie die AWS CLI für Scripting und Automatisierung
- the section called “eksctl”- Verwenden Sie eksctl für ein Kubernetes-natives Erlebnis

Was passiert, wenn Sie eine ACK-Fähigkeit erstellen

Wenn Sie eine ACK-Fähigkeit erstellen:

1. EKS erstellt den ACK-Fähigkeitsdienst und konfiguriert ihn für die Überwachung und Verwaltung der Ressourcen in Ihrem Cluster
2. Benutzerdefinierte Ressourcendefinitionen (CRDs) sind in Ihrem Cluster installiert
3. Die Funktion setzt die von Ihnen bereitgestellte IAM-Fähigkeitsrolle voraus
4. ACK beginnt, in Ihrem Cluster nach seinen benutzerdefinierten Ressourcen Ausschau zu halten
5. Der Funktionsstatus ändert sich von `CREATING` zu `ACTIVE`

Sobald sie aktiv sind, können Sie benutzerdefinierte ACK-Ressourcen in Ihrem Cluster erstellen, um AWS Ressourcen zu verwalten.

Nächste Schritte

Nach dem Erstellen der ACK-Fähigkeit:

- the section called “ACK-Konzepte”- Verstehen Sie die ACK-Konzepte und beginnen Sie mit AWS Ressourcen

- the section called “ACK-Konzepte”- Erfahren Sie mehr über Abstimmungen, Feldexporte und Muster bei der Nutzung von Ressourcen
- the section called “Konfigurieren von -Berechtigungen”- Konfigurieren Sie IAM-Berechtigungen und Muster für mehrere Konten

Erstellen Sie eine ACK-Fähigkeit mithilfe der Konsole

In diesem Thema wird beschrieben, wie Sie eine AWS Controllers for Kubernetes (ACK) -Funktion mithilfe von erstellen. AWS-Managementkonsole

Erstellen Sie die ACK-Fähigkeit

1. Öffnen Sie die Amazon EKS-Konsole unter https://console.aws.amazon.com/eks/ home#/clusters.
2. Wählen Sie Ihren Clusternamen aus, um die Cluster-Detailseite zu öffnen.
3. Wählen Sie die Registerkarte Funktionen.
4. Wählen Sie in der linken Navigationsleiste AWS Controllers for Kubernetes (ACK) aus.
5. Wählen Sie Create AWS Controllers for Kubernetes Capability aus.
6. Für die IAM-Fähigkeitsrolle:
   - Wenn Sie bereits über eine IAM-Fähigkeitsrolle verfügen, wählen Sie sie aus der Dropdownliste aus
   - Wenn Sie eine Rolle erstellen müssen, wählen Sie Administratorrolle erstellen

     Dadurch wird die IAM-Konsole auf einer neuen Registerkarte mit vorab ausgefüllten Vertrauensrichtlinien und der `AdministratorAccess` verwalteten Richtlinie geöffnet. Sie können die Auswahl dieser Richtlinie aufheben und bei Bedarf weitere Berechtigungen hinzufügen.

     Kehren Sie nach dem Erstellen der Rolle zur EKS-Konsole zurück. Die Rolle wird automatisch ausgewählt.

     > **Important**
     >
     > Die vorgeschlagene `AdministratorAccess` Richtlinie gewährt umfassende Berechtigungen und soll den Einstieg vereinfachen. Für den produktiven Einsatz sollten Sie diese Richtlinie durch eine benutzerdefinierte Richtlinie ersetzen, die nur die Berechtigungen gewährt, die für die spezifischen AWS Dienste erforderlich sind,

die Sie mit ACK verwalten möchten. Anleitungen zur Erstellung von Richtlinien mit den geringsten Rechten finden Sie unter und. the section called “Konfigurieren von - Berechtigungen” the section called “Überlegungen zu den Funktionen von EKS”

7. Wählen Sie Erstellen aus.

Der Prozess zur Schaffung von Fähigkeiten beginnt.

Stellen Sie sicher, dass die Fähigkeit aktiv ist

1. Sehen Sie sich auf der Registerkarte Funktionen den Status der ACK-Fähigkeit an.
2. Warten Sie, bis sich der Status von `CREATING` zu ändert`ACTIVE`.
3. Sobald die Funktion aktiv ist, ist sie einsatzbereit.

Informationen zum Status der Funktionen und zur Problembehandlung finden Sie unterthe section called “Mit Fähigkeiten arbeiten”.

Stellen Sie sicher, dass benutzerdefinierte Ressourcen verfügbar sind

Stellen Sie nach der Aktivierung der Funktion sicher, dass benutzerdefinierte ACK-Ressourcen in Ihrem Cluster verfügbar sind.

Verwenden der Konsole

1. Navigieren Sie in der Amazon EKS-Konsole zu Ihrem Cluster.
2. Wählen Sie die Registerkarte Ressourcen
3. Wählen Sie Erweiterungen
4. Wählen Sie CustomResourceDefinitions aus.

Sie sollten eine Reihe von AWS Ressourcen CRDs auflisten sehen.

Verwenden Sie kubectl

```
kubectl api-resources | grep services.k8s.aws
```

Sie sollten eine Reihe von Ressourcen APIs aufgelistet sehen. AWS

> **Note**
>
> Die Funktion für AWS Controller for Kubernetes wird eine Reihe von CRDs für eine Vielzahl von AWS Ressourcen installieren.

Nächste Schritte

- the section called “ACK-Konzepte”- Verstehen Sie die ACK-Konzepte und legen Sie los
- the section called “Konfigurieren von -Berechtigungen”- Konfigurieren Sie IAM-Berechtigungen für andere Dienste AWS
- the section called “Mit Fähigkeiten arbeiten”- Verwalten Sie Ihre ACK-Fähigkeitsressource

Erstellen Sie eine ACK-Fähigkeit mit der AWS CLI

In diesem Thema wird beschrieben, wie Sie mithilfe der AWS CLI eine Controllers for Kubernetes-Funktion (ACK) erstellen. AWS

Voraussetzungen

- AWS CLI — Version `2.12.3` oder höher. Führen Sie den Befehl aus, um Ihre Version zu überprüfen`aws --version`. Weitere Informationen finden Sie im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle unter Installation.
- **`kubectl`** – Ein Befehlszeilentool für die Arbeit mit Kubernetes-Clustern. Weitere Informationen finden Sie unter the section called “kubectl und eksctl einrichten”.

Schritt 1: Erstellen Sie eine IAM-Capability-Rolle

Erstellen Sie eine Vertrauensrichtliniendatei:

```
cat > ack-trust-policy.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "capabilities.eks.amazonaws.com"
```

```
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
EOF
```

Erstellen Sie die IAM-Rolle:

```
aws iam create-role \
  --role-name ACKCapabilityRole \
  --assume-role-policy-document file://ack-trust-policy.json
```

Hängen Sie die `AdministratorAccess` verwaltete Richtlinie an die Rolle an:

```
aws iam attach-role-policy \
  --role-name ACKCapabilityRole \
  --policy-arn arn:aws:iam::aws:policy/AdministratorAccess
```

> **Important**
>
> Die vorgeschlagene `AdministratorAccess` Richtlinie gewährt umfassende Berechtigungen und soll den Einstieg vereinfachen. Für den produktiven Einsatz sollten Sie diese Richtlinie durch eine benutzerdefinierte Richtlinie ersetzen, die nur die Berechtigungen gewährt, die für die spezifischen AWS Dienste erforderlich sind, die Sie mit ACK verwalten möchten. Anleitungen zur Erstellung von Richtlinien mit den geringsten Rechten finden Sie unter und. the section called “Konfigurieren von -Berechtigungen” the section called “Überlegungen zu den Funktionen von EKS”

Schritt 2: Erstellen Sie die ACK-Fähigkeit

Erstellen Sie die ACK-Fähigkeitsressource auf Ihrem Cluster. *`region-code`*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet, und *`my-cluster`* ersetzen Sie es durch den Namen Ihres Clusters.

```
aws eks create-capability \
```

```
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-ack \
  --type ACK \
  --role-arn arn:aws:iam::$(aws sts get-caller-identity --query Account --output
text):role/ACKCapabilityRole \
  --delete-propagation-policy RETAIN
```

Der Befehl wird sofort zurückgegeben, aber es dauert einige Zeit, bis die Funktion aktiv wird, da EKS die erforderliche Funktionsinfrastruktur und die erforderlichen Komponenten erstellt. EKS installiert die benutzerdefinierten Kubernetes-Ressourcendefinitionen, die sich auf diese Funktion beziehen, in Ihrem Cluster, während dieser erstellt wird.

> **Note**
>
> Wenn Sie eine Fehlermeldung erhalten, dass der Cluster nicht existiert oder Sie keine Berechtigungen haben, überprüfen Sie Folgendes:
>
> - Der Clustername ist korrekt
> - Ihre AWS CLI ist für die richtige Region konfiguriert
> - Sie verfügen über die erforderlichen IAM-Berechtigungen

Schritt 3: Stellen Sie sicher, dass die Funktion aktiv ist

Warten Sie, bis die Funktion aktiv wird. *region-code*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet, und *my-cluster* ersetzen Sie es durch den Namen Ihres Clusters.

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-ack \
  --query 'capability.status' \
  --output text
```

Die Funktion ist bereit, wenn der Status angezeigt wird`ACTIVE`. Fahren Sie erst mit dem nächsten Schritt fort, wenn der Status lautet`ACTIVE`.

Sie können sich auch die vollständigen Funktionsdetails ansehen:

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-ack
```

Schritt 4: Stellen Sie sicher, dass benutzerdefinierte Ressourcen verfügbar sind

Nachdem die Funktion aktiviert ist, stellen Sie sicher, dass benutzerdefinierte ACK-Ressourcen in Ihrem Cluster verfügbar sind:

```
kubectl api-resources | grep services.k8s.aws
```

Sie sollten eine Reihe von AWS Ressourcen in der APIs Liste sehen.

> **Note**
>
> Die Funktion für AWS Controller for Kubernetes wird eine Reihe von CRDs für eine Vielzahl von AWS Ressourcen installieren.

Nächste Schritte

- the section called “ACK-Konzepte”- Verstehen Sie die ACK-Konzepte und legen Sie los
- the section called “Konfigurieren von -Berechtigungen”- Konfigurieren Sie IAM-Berechtigungen für andere Dienste AWS
- the section called “Mit Fähigkeiten arbeiten”- Verwalten Sie Ihre ACK-Fähigkeitsressource

Erstellen Sie eine ACK-Fähigkeit mit eksctl

In diesem Thema wird beschrieben, wie Sie mit eksctl eine AWS Controllers for Kubernetes-Funktion (ACK) erstellen.

> **Note**
>
> Für die folgenden Schritte ist die Version eksctl oder höher erforderlich. `0.220.0` Führen Sie den Befehl aus, um Ihre Version zu überprüfen. `eksctl version`

Schritt 1: Erstellen Sie eine IAM-Capability-Rolle

Erstellen Sie eine Vertrauensrichtliniendatei:

```
cat > ack-trust-policy.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "capabilities.eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
EOF
```

Erstellen Sie die IAM-Rolle:

```
aws iam create-role \
  --role-name ACKCapabilityRole \
  --assume-role-policy-document file://ack-trust-policy.json
```

Hängen Sie die `AdministratorAccess` verwaltete Richtlinie an die Rolle an:

```
aws iam attach-role-policy \
  --role-name ACKCapabilityRole \
  --policy-arn arn:aws:iam::aws:policy/AdministratorAccess
```

> **Important**
>
> Die vorgeschlagene `AdministratorAccess` Richtlinie gewährt umfassende Berechtigungen und soll den Einstieg vereinfachen. Für den produktiven Einsatz sollten Sie diese Richtlinie durch eine benutzerdefinierte Richtlinie ersetzen, die nur die Berechtigungen gewährt, die für die spezifischen AWS Dienste erforderlich sind, die Sie mit ACK verwalten

möchten. Anleitungen zur Erstellung von Richtlinien mit den geringsten Rechten finden Sie unter und. the section called “Konfigurieren von -Berechtigungen” the section called “Überlegungen zu den Funktionen von EKS”

Important

Diese Richtlinie gewährt Berechtigungen für die S3-Bucketverwaltung mit`"Resource": "*"`, wodurch Operationen auf allen S3-Buckets möglich sind.
Für Produktionszwecke: * Beschränken Sie das `Resource` Feld auf bestimmte Bucket ARNs - oder Namensmuster * Verwenden Sie IAM-Bedingungsschlüssel, um den Zugriff anhand von Ressourcen-Tags einzuschränken * Gewähren Sie nur die für Ihren Anwendungsfall erforderlichen Mindestberechtigungen
Weitere AWS Dienste finden Sie unterthe section called “Konfigurieren von -Berechtigungen”.

Fügen Sie die Richtlinie an die Rolle an:

```
aws iam attach-role-policy \
  --role-name ACKCapabilityRole \
  --policy-arn arn:aws:iam::$(aws sts get-caller-identity --query Account --output
 text):policy/ACKS3Policy
```

Schritt 2: Erstellen Sie die ACK-Fähigkeit

Erstellen Sie die ACK-Fähigkeit mit eksctl. *`region-code`*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet, und *`my-cluster`* ersetzen Sie es durch den Namen Ihres Clusters.

```
eksctl create capability \
  --cluster [.replaceable]`my-cluster` \
  --region [.replaceable]`region-code` \
  --name ack \
  --type ACK \
  --role-arn arn:aws:iam::$(aws sts get-caller-identity --query Account --output
 text):role/ACKCapabilityRole \
  --ack-service-controllers s3
```

> **Note**
>
> Die `--ack-service-controllers` Flagge ist optional. Wenn es weggelassen wird, aktiviert ACK alle verfügbaren Controller. Für eine bessere Leistung und Sicherheit sollten Sie erwägen, nur die Controller zu aktivieren, die Sie benötigen. Sie können mehrere Controller angeben: `--ack-service-controllers s3,rds,dynamodb`

Der Befehl kehrt sofort zurück, aber es dauert einige Zeit, bis die Funktion aktiv wird.

Schritt 3: Stellen Sie sicher, dass die Funktion aktiv ist

Überprüfen Sie den Status der Fähigkeit:

```
eksctl get capability \
  --cluster [.replaceable]`my-cluster` \
  --region [.replaceable]`region-code` \
  --name ack
```

Die Fähigkeit ist bereit, wenn der Status angezeigt wird`ACTIVE`.

Schritt 4: Stellen Sie sicher, dass benutzerdefinierte Ressourcen verfügbar sind

Nachdem die Funktion aktiviert ist, stellen Sie sicher, dass benutzerdefinierte ACK-Ressourcen in Ihrem Cluster verfügbar sind:

```
kubectl api-resources | grep services.k8s.aws
```

Sie sollten eine Reihe von AWS Ressourcen in der APIs Liste sehen.

> **Note**
>
> Die Funktion für AWS Controller for Kubernetes wird eine Reihe von CRDs für eine Vielzahl von AWS Ressourcen installieren.

Nächste Schritte

- the section called “ACK-Konzepte”- Verstehen Sie die ACK-Konzepte und legen Sie los

- the section called “Konfigurieren von -Berechtigungen”- Konfigurieren Sie IAM-Berechtigungen für andere Dienste AWS
- the section called “Mit Fähigkeiten arbeiten”- Verwalten Sie Ihre ACK-Fähigkeitsressource

ACK-Konzepte

ACK verwaltet AWS Ressourcen über Kubernetes, APIs indem es kontinuierlich den gewünschten Status in Ihren Manifesten mit dem tatsächlichen Status in abgleicht. AWS Wenn Sie eine benutzerdefinierte Kubernetes-Ressource erstellen oder aktualisieren, führt ACK die erforderlichen AWS API-Aufrufe durch, um die entsprechende AWS Ressource zu erstellen oder zu ändern, überwacht sie dann auf Abweichungen und aktualisiert den Kubernetes-Status, um den aktuellen Status widerzuspiegeln. Mit diesem Ansatz können Sie die Infrastruktur mithilfe vertrauter Kubernetes-Tools und -Workflows verwalten und gleichzeitig die Konsistenz zwischen Ihrem Cluster und aufrechterhalten. AWS

In diesem Thema werden die grundlegenden Konzepte erläutert, die der Verwaltung von AWS Ressourcen durch ACK über Kubernetes zugrunde liegen. APIs

Erste Schritte mit ACK

Nachdem Sie die ACK-Fähigkeit erstellt haben (siehethe section called “ACK-Fähigkeit erstellen”), können Sie mit der Verwaltung von AWS Ressourcen mithilfe von Kubernetes-Manifesten in Ihrem Cluster beginnen.

Erstellen Sie beispielsweise dieses S3-Bucket-Manifest in und wählen Sie `bucket.yaml` dabei Ihren eigenen eindeutigen Bucket-Namen.

```
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: my-test-bucket
  namespace: default
spec:
  name: my-unique-bucket-name-12345
```

Wenden Sie das Manifest an:

```
kubectl apply -f bucket.yaml
```

Überprüfen Sie den Status:

```
kubectl get bucket my-test-bucket
kubectl describe bucket my-test-bucket
```

Stellen Sie sicher, dass der Bucket erstellt wurde in AWS:

```
aws s3 ls | grep my-unique-bucket-name-12345
```

Löschen Sie die Kubernetes-Ressource:

```
kubectl delete bucket my-test-bucket
```

Stellen Sie sicher, dass der Bucket gelöscht wurde aus: AWS

```
aws s3 ls | grep my-unique-bucket-name-12345
```

Der Bucket sollte nicht mehr in der Liste erscheinen, was zeigt, dass ACK den gesamten Lebenszyklus von AWS Ressourcen verwaltet.

Weitere Informationen zu den ersten Schritten mit ACK finden Sie unter Erste Schritte mit ACK.

Lebenszyklus und Abstimmung von Ressourcen

ACK verwendet eine kontinuierliche Abstimmungsschleife, um sicherzustellen, dass Ihre AWS Ressourcen dem gewünschten Status entsprechen, der in Ihren Kubernetes-Manifesten definiert ist.

So funktioniert der Abgleich:

1. Sie erstellen oder aktualisieren eine benutzerdefinierte Kubernetes-Ressource (z. B. einen S3-Bucket)
2. ACK erkennt die Änderung und vergleicht den gewünschten Zustand mit dem tatsächlichen Zustand in AWS
3. Wenn sie sich unterscheiden, führt ACK AWS API-Aufrufe durch, um den Unterschied auszugleichen
4. ACK aktualisiert den Ressourcenstatus in Kubernetes, um den aktuellen Status widerzuspiegeln

5. Die Schleife wiederholt sich kontinuierlich, in der Regel alle paar Stunden

Der Abgleich wird ausgelöst, wenn Sie eine neue Kubernetes-Ressource erstellen, die einer vorhandenen Ressource aktualisieren oder wenn ACK feststelltspec, dass aufgrund manueller Änderungen, die außerhalb AWS von ACK vorgenommen wurden, Abweichungen auftreten. Darüber hinaus führt ACK einen regelmäßigen Abgleich mit einer Resynchronisierungszeit von 10 Stunden durch. Änderungen an Kubernetes-Ressourcen lösen einen sofortigen Abgleich aus, während die passive Drift-Erkennung von AWS Upstream-Ressourcenänderungen während der regelmäßigen Neusynchronisierung erfolgt.

Bei der Bearbeitung des obigen Beispiels für die ersten Schritte führt ACK die folgenden Schritte aus:

1. Prüft, ob ein Bucket existiert in AWS
2. Wenn nicht, ruft `s3:CreateBucket`
3. Aktualisiert den Kubernetes-Status mit dem Bucket ARN und dem Status
4. Setzt die Überwachung auf Drift fort

Weitere Informationen zur Funktionsweise von ACK finden Sie unter ACK Reconciliation.

Statusbedingungen

ACK-Ressourcen verwenden Statusbedingungen, um ihren Status zu kommunizieren. Wenn Sie diese Bedingungen verstehen, können Sie Probleme beheben und den Zustand der Ressourcen besser verstehen.

- Bereit: Zeigt an, dass die Ressource bereit ist, genutzt zu werden (standardisierter Kubernetes-Zustand).
- ZURÜCK. ResourceSynced: Zeigt an, dass die Ressourcenspezifikation dem AWS Ressourcenstatus entspricht.
- ack.Terminal: Zeigt an, dass ein nicht behebbarer Fehler aufgetreten ist.
- ack.Adopted: Zeigt an, dass die Ressource aus einer vorhandenen Ressource übernommen und nicht neu erstellt wurde. AWS
- Ack.Recoverable: Weist auf einen behebbaren Fehler hin, der ohne Aktualisierung der Spezifikation behoben werden kann.
- ACK.Advisory: Stellt Beratungsinformationen zu der Ressource bereit.
- ACK. LateInitialized: Zeigt an, ob die späte Initialisierung der Felder abgeschlossen ist.

- ZURÜCK. ReferencesResolved: Gibt an, ob alle `AWSResourceReference` Felder aufgelöst wurden.
- ZURÜCK. IAMRoleAusgewählt: Gibt an, ob ein IAMRole Selektor für die Verwaltung dieser Ressource ausgewählt wurde.

Überprüfen Sie den Ressourcenstatus:

```
# Check if resource is ready
kubectl get bucket my-bucket -o jsonpath='{.status.conditions[?
(@.type=="Ready")].status}'

# Check for terminal errors
kubectl get bucket my-bucket -o jsonpath='{.status.conditions[?
(@.type=="ACK.Terminal")]}'
```

Beispiel für einen Status:

```
status:
  conditions:
  - type: Ready
    status: "True"
    lastTransitionTime: "2024-01-15T10:30:00Z"
  - type: ACK.ResourceSynced
    status: "True"
    lastTransitionTime: "2024-01-15T10:30:00Z"
  - type: ACK.Terminal
    status: "True"
  ackResourceMetadata:
    arn: arn:aws:s3:::my-unique-bucket-name
    ownerAccountID: "111122223333"
    region: us-west-2
```

Weitere Informationen zum ACK-Status und zu den ACK-Bedingungen finden Sie unter ACK-Bedingungen.

Richtlinien zum Löschen

Die Löschrichtlinie von ACK steuert, was mit AWS Ressourcen passiert, wenn Sie die Kubernetes-Ressource löschen.

Löschen (Standard)

Die AWS Ressource wird gelöscht, wenn Sie die Kubernetes-Ressource löschen: Dies ist das Standardverhalten.

```
# No annotation needed - this is the default
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: temp-bucket
spec:
  name: temporary-bucket
```

Durch das Löschen dieser Ressource wird der S3-Bucket in gelöscht. AWS

Beibehalten

Die AWS Ressource bleibt erhalten, wenn Sie die Kubernetes-Ressource löschen:

```
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: important-bucket
  annotations:
    services.k8s.aws/deletion-policy: "retain"
spec:
  name: production-data-bucket
```

Wenn Sie diese Ressource löschen, wird sie aus Kubernetes entfernt, der S3-Bucket bleibt jedoch erhalten. AWS

Die `retain` Richtlinie ist nützlich für Produktionsdatenbanken, die die Kubernetes-Ressource überdauern sollen, für gemeinsam genutzte Ressourcen, die von mehreren Anwendungen genutzt werden, für Ressourcen mit wichtigen Daten, die nicht versehentlich gelöscht werden sollten, oder für temporäres ACK-Management, bei dem Sie eine Ressource übernehmen, konfigurieren und sie dann wieder für die manuelle Verwaltung freigeben.

Weitere Informationen zur ACK-Löschrichtlinie finden Sie unter ACK-Löschrichtlinie.

Einführung von Ressourcen

Durch die Einführung können Sie vorhandene AWS Ressourcen unter das ACK-Management stellen, ohne sie neu erstellen zu müssen.

Wann sollte Adoption verwendet werden:

- Migration der vorhandenen Infrastruktur zum ACK-Management
- Wiederherstellung verwaister AWS Ressourcen bei versehentlichem Löschen von Ressourcen in Kubernetes
- Importieren von Ressourcen, die mit anderen Tools erstellt wurden (CloudFormation, Terraform)

So funktioniert Adoption:

```
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: existing-bucket
  annotations:
    services.k8s.aws/adoption-policy: "adopt-or-create"
spec:
  name: my-existing-bucket-name
```

Wenn Sie diese Ressource erstellen:

1. ACK prüft, ob ein Bucket mit diesem Namen vorhanden ist AWS
2. Wenn es gefunden wird, übernimmt ACK es (keine API-Aufrufe zum Erstellen)
3. ACK liest die aktuelle Konfiguration von AWS
4. ACK aktualisiert den Kubernetes-Status, um den aktuellen Status widerzuspiegeln
5. Zukünftige Updates stimmen die Ressource normal ab

Nach der Übernahme werden Ressourcen wie jede andere ACK-Ressource verwaltet. Wenn Sie die Kubernetes-Ressource löschen, wird die Ressource gelöscht, sofern Sie nicht die AWS Löschrichtlinie verwenden. `retain`

Bei der Übernahme von Ressourcen muss die AWS Ressource bereits vorhanden sein und ACK benötigt Leseberechtigungen, um sie zu finden. Die `adopt-or-create` Richtlinie übernimmt die Ressource, falls sie existiert, oder erstellt sie, wenn sie nicht existiert. Dies ist nützlich, wenn Sie einen deklarativen Workflow benötigen, der unabhängig davon funktioniert, ob die Ressource existiert oder nicht.

Weitere Informationen zur Einführung von ACK-Ressourcen finden Sie unter Einführung von ACK-Ressourcen.

Konto- und regionsübergreifende Ressourcen

ACK kann Ressourcen in verschiedenen AWS Konten und Regionen von einem einzigen Cluster aus verwalten.

Regionsübergreifende Anmerkungen zu Ressourcen

Sie können die Region einer AWS Ressource mithilfe einer Anmerkung angeben:

```
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: eu-bucket
  annotations:
    services.k8s.aws/region: eu-west-1
spec:
  name: my-eu-bucket
```

Sie können auch die Region aller AWS Ressourcen angeben, die in einem bestimmten Namespace erstellt wurden:

Namespace-Anmerkungen

Legen Sie eine Standardregion für alle Ressourcen in einem Namespace fest:

```
apiVersion: v1
kind: Namespace
metadata:
  name: production
  annotations:
    services.k8s.aws/default-region: us-west-2
```

In diesem Namespace erstellte Ressourcen verwenden diese Region, sofern sie nicht durch eine Anmerkung auf Ressourcenebene überschrieben wird.

Kontoübergreifend

Verwenden Sie IAM-Rollenselektoren, um bestimmte IAM-Rollen Namespaces zuzuordnen:

```
apiVersion: services.k8s.aws/v1alpha1
kind: IAMRoleSelector
```

```
metadata:
  name: target-account-config
spec:
  arn: arn:aws:iam::444455556666:role/ACKTargetAccountRole
  namespaceSelector:
    names:
      - production
```

Ressourcen, die im zugewiesenen Namespace erstellt wurden, verwenden automatisch die angegebene Rolle.

Weitere Informationen zu IAM-Rollenselektoren finden Sie unter Accountübergreifendes ACK-Ressourcenmanagement. Einzelheiten zur kontoübergreifenden Konfiguration finden Sie unter. the section called “Konfigurieren von -Berechtigungen”

Fehlerbehandlung und Verhalten bei Wiederholungsversuchen

ACK behandelt automatisch vorübergehende Fehler und wiederholt fehlgeschlagene Operationen.

Strategie erneut versuchen:

- Vorübergehende Fehler (Ratenbegrenzung, vorübergehende Serviceprobleme, unzureichende Berechtigungen) lösen automatische Wiederholungsversuche aus
- Exponentieller Backoff verhindert Überforderung AWS APIs
- Die maximale Anzahl von Wiederholungsversuchen ist je nach Fehlertyp unterschiedlich
- Dauerhafte Fehler (ungültige Parameter, Konflikte mit Ressourcennamen) führen nicht zu einem erneuten Versuch

Überprüfen Sie den Ressourcenstatus auf Fehlerdetails mit`kubectl describe`:

```
kubectl describe bucket my-bucket
```

Suchen Sie nach Statusbedingungen mit Fehlermeldungen, Ereignissen, die die letzten Abgleichversuche anzeigen, und dem `message` Feld unter Statusbedingungen, in denen Fehler erklärt werden. Zu den häufigsten Fehlern gehören unzureichende IAM-Berechtigungen, Konflikte bei Ressourcennamen AWS, ungültige Konfigurationswerte in der `spec` und Überschreitung von AWS Servicekontingenten.

Informationen zur Behebung häufiger Fehler finden Sie unterthe section called “Fehlerbehebung”.

Ressourcenzusammensetzung mit Kro

Verwenden Sie die EKS-Funktion für kro (Kube Resource Orchestrator), um mehrere ACK-Ressourcen zusammenzustellen und miteinander zu verbinden. kro bietet eine deklarative Möglichkeit, Ressourcengruppen zu definieren und Konfigurationen zwischen Ressourcen zu übergeben, um komplexe Infrastrukturmuster einfach zu verwalten.

Ausführliche Beispiele für die Erstellung benutzerdefinierter Ressourcenzusammenstellungen mit ACK-Ressourcen finden Sie unter the section called “Kro-Konzepte”

Nächste Schritte

- the section called “Überlegungen”- EKS-spezifische Muster und Integrationsstrategien

ACK-Berechtigungen konfigurieren

ACK benötigt IAM-Berechtigungen, um AWS Ressourcen in Ihrem Namen zu erstellen und zu verwalten. In diesem Thema wird erklärt, wie IAM mit ACK zusammenarbeitet, und es finden Sie Anleitungen zur Konfiguration von Berechtigungen für verschiedene Anwendungsfälle.

Wie funktioniert IAM mit ACK

ACK verwendet IAM-Rollen, um sich bei Ihren Ressourcen zu authentifizieren AWS und Aktionen für sie auszuführen. Es gibt zwei Möglichkeiten, ACK Berechtigungen zu erteilen:

Funktionsrolle: Die IAM-Rolle, die Sie bei der Erstellung der ACK-Fähigkeit angeben. Diese Rolle wird standardmäßig für alle ACK-Operationen verwendet.

IAM-Rollenselektoren: Zusätzliche IAM-Rollen, die bestimmten Namespaces oder Ressourcen zugeordnet werden können. Diese Rollen haben Vorrang vor der Capability-Rolle für Ressourcen in ihrem Geltungsbereich.

Wenn ACK eine Ressource erstellen oder verwalten muss, bestimmt es, welche IAM-Rolle verwendet werden soll:

1. Prüfen Sie, ob ein IAMRole Selector dem Namespace der Ressource entspricht
2. Wenn eine Übereinstimmung gefunden wird, gehen Sie von dieser IAM-Rolle aus
3. Verwenden Sie andernfalls die Capability-Rolle

Dieser Ansatz ermöglicht ein flexibles Berechtigungsmanagement, von einfachen Einzelrollen-Setups bis hin zu komplexen Konfigurationen mit mehreren Konten und mehreren Teams.

Erste Schritte: Einfache Einrichtung von Berechtigungen

Für Entwicklungs-, Test- oder einfache Anwendungsfälle können Sie der Capability Role alle erforderlichen Serviceberechtigungen direkt hinzufügen.

Dieser Ansatz funktioniert gut, wenn:

- Sie fangen mit ACK an
- Alle Ressourcen befinden sich auf demselben AWS Konto
- Ein einziges Team verwaltet alle ACK-Ressourcen
- Sie vertrauen darauf, dass alle ACK-Benutzer dieselben Berechtigungen haben

Bewährte Methode für die Produktion: IAM-Rollenselektoren

Verwenden Sie in Produktionsumgebungen IAM-Rollenselektoren, um den Zugriff mit den geringsten Rechten und die Isolierung auf Namespace-Ebene zu implementieren.

Bei der Verwendung von IAM-Rollenselektoren benötigt die Capability Role nur die erforderlichen Berechtigungen, um die dienstspezifischen Rollen zu übernehmen. `sts:AssumeRole` `sts:TagSession` Sie müssen der Capability-Rolle selbst keine AWS Serviceberechtigungen (wie S3 oder RDS) hinzufügen — diese Berechtigungen werden den einzelnen IAM-Rollen erteilt, die die Capability-Rolle annimmt.

Wählen Sie zwischen Berechtigungsmodellen:

Verwenden Sie direkte Berechtigungen (Hinzufügen von Serviceberechtigungen zur Capability Role), wenn:

- Sie sind am Anfang und möchten die einfachste Einrichtung
- Alle Ressourcen befinden sich in demselben Konto wie Ihr Cluster
- Sie haben administrative, clusterweite Berechtigungsanforderungen
- Alle Teams können dieselben Berechtigungen gemeinsam nutzen

Verwenden Sie IAM-Rollenselektoren in folgenden Fällen:

- Verwaltung von Ressourcen über mehrere Konten hinweg AWS

- Verschiedene Teams oder Namespaces benötigen unterschiedliche Berechtigungen
- Sie benötigen eine differenzierte Zugriffskontrolle pro Namespace
- Sie möchten sich an die Sicherheitsverfahren mit den geringsten Rechten halten

Sie können mit direkten Berechtigungen beginnen und später zu IAM Role Selectors migrieren, wenn Ihre Anforderungen steigen.

Warum sollten Sie IAM-Rollenselektoren in der Produktion verwenden:

- Geringste Rechte: Jeder Namespace erhält nur die Berechtigungen, die er benötigt
- Teamisolierung: Team A kann nicht versehentlich die Berechtigungen von Team B verwenden
- Einfachere Prüfung: Klare Zuordnung, welcher Namespace welche Rolle verwendet
- Kontoübergreifende Unterstützung: Erforderlich für die Verwaltung von Ressourcen in mehreren Konten
- Trennung der Anliegen: Verschiedene Dienste oder Umgebungen verwenden unterschiedliche Rollen

Grundlegende Einrichtung des IAM-Rollenauswahlsystems

Schritt 1: Erstellen Sie eine dienstspezifische IAM-Rolle

Erstellen Sie eine IAM-Rolle mit Berechtigungen für bestimmte Dienste: AWS

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:*"
      ],
      "Resource": "*"
    }
  ]
}
```

Konfigurieren Sie die Vertrauensrichtlinie so, dass sie von der Capability Role übernommen wird:

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/ACKCapabilityRole"
      },
      "Action": ["sts:AssumeRole", "sts:TagSession"]
    }
  ]
}
```

Schritt 2: Erteilen Sie der Capability-Rolle die AssumeRole Berechtigung

Fügen Sie der Capability-Rolle die Erlaubnis hinzu, die dienstspezifische Rolle anzunehmen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["sts:AssumeRole", "sts:TagSession"],
      "Resource": "arn:aws:iam::111122223333:role/ACK-S3-Role"
    }
  ]
}
```

Schritt 3: Selektor erstellen IAMRole

Ordnen Sie die IAM-Rolle einem Namespace zu:

```
apiVersion: services.k8s.aws/v1alpha1
kind: IAMRoleSelector
metadata:
  name: s3-namespace-config
spec:
  arn: arn:aws:iam::111122223333:role/ACK-S3-Role
  namespaceSelector:
    names:
      - s3-resources
```

Schritt 4: Ressourcen im zugewiesenen Namespace erstellen

Ressourcen im `s3-resources` Namespace verwenden automatisch die angegebene Rolle:

```
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: my-bucket
  namespace: s3-resources
spec:
  name: my-production-bucket
```

Verwaltung mehrerer Konten

Verwenden Sie IAM-Rollenauswahlen, um Ressourcen für mehrere Konten zu verwalten. AWS

Schritt 1: Erstellen Sie eine kontoübergreifende IAM-Rolle

Erstellen Sie im Zielkonto (444455556666) eine Rolle, die der Capability-Rolle des Quellkontos vertraut:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/ACKCapabilityRole"
      },
      "Action": ["sts:AssumeRole", "sts:TagSession"]
    }
  ]
}
```

Ordnen Sie dieser Rolle dienstspezifische Berechtigungen zu.

Schritt 2: Erteilen Sie die Erlaubnis AssumeRole

Erlauben Sie der Capability Role im Quellkonto (111122223333), die Rolle des Zielkontos anzunehmen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Action": ["sts:AssumeRole", "sts:TagSession"],
      "Resource": "arn:aws:iam::444455556666:role/ACKTargetAccountRole"
    }
  ]
}
```

Schritt 3: Selektor erstellen IAMRole

Ordnen Sie die kontoübergreifende Rolle einem Namespace zu:

```
apiVersion: services.k8s.aws/v1alpha1
kind: IAMRoleSelector
metadata:
  name: production-account-config
spec:
  arn: arn:aws:iam::444455556666:role/ACKTargetAccountRole
  namespaceSelector:
    names:
      - production
```

Schritt 4: Ressourcen erstellen

Ressourcen im `production` Namespace werden im Zielkonto erstellt:

```
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: my-bucket
  namespace: production
spec:
  name: my-cross-account-bucket
```

Sitzungs-Tags

Die EKS-ACK-Fähigkeit legt automatisch Sitzungs-Tags für alle AWS API-Anfragen fest. Diese Tags ermöglichen eine detaillierte Zugriffskontrolle und Prüfung, indem sie die Quelle jeder Anfrage identifizieren.

Verfügbare Sitzungs-Tags

Die folgenden Sitzungs-Tags sind in jedem AWS API-Aufruf von ACK enthalten:

Tag-Schlüssel	Description
`eks:eks-capability-arn`	Der ARN der EKS-Funktion, die die Anfrage stellt
`eks:kubernetes-namespace`	Der Kubernetes-Namespace der verwalteten Ressource
`eks:kubernetes-api-group`	Die Kubernetes-API-Gruppe der Ressource (zum Beispiel) `s3.services.k8s.aws`

Verwenden von Sitzungs-Tags für die Zugriffskontrolle

Sie können diese Sitzungs-Tags in den IAM-Richtlinienbedingungen verwenden, um einzuschränken, welche Ressourcen ACK verwalten kann. Dies bietet eine zusätzliche Sicherheitsebene, die über die auf Namespaces basierenden IAM-Rollenselektoren hinausgeht.

Beispiel: Nach Namespace einschränken

Erlauben Sie ACK, S3-Buckets nur zu erstellen, wenn die Anfrage aus dem Namespace stammt: `production`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:CreateBucket",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/eks:kubernetes-namespace": "production"
        }
      }
    }
  ]
}
```

Beispiel: Nach Fähigkeit einschränken

Nur Aktionen von einer bestimmten ACK-Fähigkeit aus zulassen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/eks:eks-capability-arn": "arn:aws:eks:us-
west-2:111122223333:capability/my-cluster/ack/my-ack"
        }
      }
    }
  ]
}
```

> **Note**
>
> Sitzungs-Tags unterscheiden sich von selbstverwaltetem ACK, bei dem diese Tags nicht standardmäßig festgelegt werden. Dies ermöglicht eine detailliertere Zugriffskontrolle mit der verwalteten Funktion.

Erweiterte IAM-Rollenauswahlmuster

Eine erweiterte Konfiguration mit Labelselektoren, ressourcenspezifischer Rollenzuweisung und weiteren Beispielen finden Sie in der ACK IRSA-Dokumentation.

Nächste Schritte

- the section called “ACK-Konzepte”- Verstehen Sie die ACK-Konzepte und den Ressourcenlebenszyklus
- the section called “ACK-Konzepte”- Erfahren Sie mehr über Richtlinien für die Einführung und Löschung von Ressourcen
- the section called “Überlegungen zu den Funktionen von EKS”- Machen Sie sich mit bewährten Sicherheitsmethoden für Funktionen vertraut

ACK-Überlegungen für EKS

In diesem Thema werden wichtige Überlegungen zur Nutzung der EKS-Funktion für ACK behandelt, darunter die IAM-Konfiguration, Muster für mehrere Konten und die Integration mit anderen EKS-Funktionen.

IAM-Konfigurationsmuster

Die ACK-Fähigkeit verwendet eine IAM-Fähigkeitsrolle für die Authentifizierung. AWS Wählen Sie das richtige IAM-Muster, das Ihren Anforderungen entspricht.

Einfach: Eine einzige Capability-Rolle

Erteilen Sie der Capability-Rolle direkt alle erforderlichen Berechtigungen für Entwicklung, Tests oder einfache Anwendungsfälle.

Wann sollte Folgendes verwendet werden:

- Erste Schritte mit ACK
- Bereitstellungen mit einem einzigen Konto
- Alle Ressourcen werden von einem Team verwaltet
- Entwicklungs- und Testumgebungen

Beispiel: Fügen Sie Ihrer Capability Role S3- und RDS-Berechtigungen mit Bedingungen für die Kennzeichnung von Ressourcen hinzu:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:*"],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestedRegion": ["us-west-2", "us-east-1"]
        }
      }
    },
```

```
    {
      "Effect": "Allow",
      "Action": ["rds:*"],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestedRegion": ["us-west-2", "us-east-1"]
        },
      }
    }
  ]
}
```

Dieses Beispiel beschränkt die S3- und RDS-Operationen auf bestimmte Regionen und erfordert, dass RDS-Ressourcen über ein `ManagedBy: ACK` Tag verfügen.

Produktion: IAM-Rollenselektoren

Verwenden Sie in Produktionsumgebungen IAM-Rollenselektoren, um den Zugriff mit den geringsten Rechten und die Isolierung auf Namespace-Ebene zu implementieren.

Wann sollte Folgendes verwendet werden:

- Produktionsumgebungen
- Cluster mit mehreren Teams
- Ressourcenverwaltung für mehrere Konten
- Sicherheitsanforderungen mit den geringsten Rechten
- Verschiedene Dienste benötigen unterschiedliche Berechtigungen

Vorteile:

- Jeder Namespace erhält nur die Berechtigungen, die er benötigt
- Teamisolierung — Team A kann die Berechtigungen von Team B nicht verwenden
- Einfachere Prüfung und Einhaltung von Vorschriften
- Erforderlich für kontenübergreifendes Ressourcenmanagement

Eine ausführliche Konfiguration des IAM-Rollenauswahlsystems finden Sie unter. the section called “Konfigurieren von -Berechtigungen”

Integration mit anderen EKS-Funktionen

GitOps mit Argo CD

Verwenden Sie die EKS-Funktion für Argo CD, um ACK-Ressourcen aus Git-Repositorys bereitzustellen und GitOps Workflows für das Infrastrukturmanagement zu ermöglichen.

Überlegungen:

- Speichern Sie ACK-Ressourcen zusammen mit Anwendungsmanifesten für end-to-end GitOps
- Organisieren Sie auf der Grundlage Ihrer Teamstruktur nach Umgebung, Service oder Ressourcentyp
- Verwenden Sie die automatische Synchronisation von Argo CD für einen kontinuierlichen Abgleich
- Aktivieren Sie das Bereinigen, um gelöschte Ressourcen automatisch zu entfernen
- Ziehen Sie hub-and-spoke Muster für das Infrastrukturmanagement mehrerer Cluster in Betracht

GitOps bietet Prüfpfade, Rollback-Funktionen und deklaratives Infrastrukturmanagement. Weitere Informationen zu Argo CD finden Sie unter. the section called “Arbeiten mit Argo CD”

Ressourcenzusammensetzung mit Kro

Verwenden Sie die EKS-Funktion für kro (Kube Resource Orchestrator), um mehrere ACK-Ressourcen in abstrakte und benutzerdefinierte Elemente auf höherer Ebene zusammenzustellen. APIs

Wann sollte Kro mit ACK verwendet werden:

- Erstellen Sie wiederverwendbare Muster für gängige Infrastruktur-Stacks (Datenbank + Backup + Überwachung)
- Erstellen Sie Self-Service-Plattformen mit vereinfachten Funktionen APIs für Anwendungsteams
- Ressourcenabhängigkeiten verwalten und Werte zwischen Ressourcen übergeben (S3-Bucket-ARN an Lambda-Funktion)
- Standardisieren Sie die Infrastrukturkonfigurationen teamübergreifend
- Reduzieren Sie die Komplexität, indem Sie Implementierungsdetails hinter benutzerdefinierten Ressourcen verstecken

Beispielmuster:

- Anwendungsstapel: S3-Bucket + SQS-Warteschlange + Benachrichtigungskonfiguration
- Datenbank-Setup: RDS-Instanz + Parametergruppe + Sicherheitsgruppe + Geheimnisse
- Netzwerk: VPC + Subnetze + Routentabellen + Sicherheitsgruppen

kro kümmert sich um die Reihenfolge der Abhängigkeiten, die Statusweiterleitung und das Lebenszyklusmanagement für zusammengesetzte Ressourcen. Weitere Informationen zu Kro finden Sie unter. the section called “Kro-Konzepte”

Organisieren Sie Ihre Ressourcen

Organisieren Sie ACK-Ressourcen mithilfe von Kubernetes-Namespaces und AWS Ressourcen-Tags für eine bessere Verwaltung, Zugriffskontrolle und Kostenverfolgung.

Organisation des Namespaces

Verwenden Sie Kubernetes-Namespaces, um ACK-Ressourcen logisch nach Umgebung (Produktion, Staging, Entwicklung), Team (Plattform, Daten, ML) oder Anwendung zu trennen.

Vorteile:

- RBAC mit Namespace-Gültigkeitsbereich für die Zugriffskontrolle
- Legen Sie mithilfe von Anmerkungen Standardregionen pro Namespace fest
- Einfachere Ressourcenverwaltung und Bereinigung
- Logische Trennung im Einklang mit der Organisationsstruktur

Ressourcen-Markierung

Die EKS-ACK-Funktion wendet automatisch Standardtags auf alle AWS Ressourcen an, die sie erstellt. Diese Tags unterscheiden sich von selbstverwalteten ACK und bieten eine verbesserte Rückverfolgbarkeit.

Von der Funktion angewendete Standardtags:

Tag-Schlüssel	Description
`eks:controller-version`	Die Version des ACK-Controllers

Tag-Schlüssel	Description
`eks:kubernetes-namespace`	Der Kubernetes-Namespace der ACK-Ressource
`eks:kubernetes-resource-name`	Der Name der Kubernetes-Ressource
`eks:kubernetes-api-group`	Die Kubernetes-API-Gruppe (zum Beispiel) `s3.services.k8s.aws`
`eks:eks-capability-arn`	Der ARN der EKS ACK-Fähigkeit

> **Note**
>
> Selbstverwaltetes ACK verwendet verschiedene Standardtags: `services.k8s.aws/controller-version` und`services.k8s.aws/namespace`. Die Tags der Funktion verwenden das `eks:` Präfix aus Gründen der Konsistenz mit anderen EKS-Funktionen.

Zusätzliche empfohlene Tags:

Fügen Sie benutzerdefinierte Tags für die Kostenzuweisung, die Nachverfolgung der Eigentumsverhältnisse und für organisatorische Zwecke hinzu:

- Umgebung (Produktion, Bereitstellung, Entwicklung)
- Eigentümerschaft des Teams oder der Abteilung
- Kostenstelle für die Fakturierung
- Name der Anwendung oder des Dienstes

Migration von anderen Infrastructure-as-code Tools

Viele Unternehmen halten es für sinnvoll, Kubernetes über ihre Workload-Orchestrierung hinaus zu standardisieren. Durch die Migration des Infrastruktur- und AWS Ressourcenmanagements zu ACK können Sie das Infrastrukturmanagement mithilfe von Kubernetes zusammen mit Ihren Anwendungs-Workloads standardisieren. APIs

Vorteile der Standardisierung auf Kubernetes für die Infrastruktur:

- Zentrale Informationsquelle: Verwalten Sie sowohl Anwendungen als auch Infrastruktur in Kubernetes und ermöglichen Sie so eine Praxis end-to-end GitOps
- Einheitliches Tooling: Teams nutzen die Ressourcen und Tools von Kubernetes, anstatt mehrere Tools und Frameworks zu erlernen
- Konsistenter Abgleich: ACK gleicht AWS Ressourcen kontinuierlich ab, wie es Kubernetes für Workloads tut, und erkennt und korrigiert Abweichungen im Vergleich zu unverzichtbaren Tools
- Systemeigene Kompositionen: Wenn Kro und ACK zusammen verwendet werden, können AWS Ressourcen direkt in Anwendungs- und Ressourcenmanifesten referenziert werden, wobei Verbindungszeichenfolgen und zwischen Ressourcen weitergegeben werden ARNs
- Vereinfachter Betrieb: Eine Steuerungsebene für Bereitstellungen, Rollbacks und Beobachtbarkeit im gesamten System

ACK unterstützt die Übernahme vorhandener AWS Ressourcen, ohne sie neu zu erstellen, und ermöglicht so eine Migration ohne Ausfallzeiten von CloudFormation Terraform oder Ressourcen außerhalb des Clusters.

Übernehmen Sie eine bestehende Ressource:

```
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: existing-bucket
  annotations:
    services.k8s.aws/adoption-policy: "adopt-or-create"
spec:
  name: my-existing-bucket-name
```

Nach der Übernahme wird die Ressource von ACK verwaltet und kann über Kubernetes-Manifeste aktualisiert werden. Sie können schrittweise migrieren, Ressourcen nach Bedarf übernehmen und gleichzeitig die vorhandenen IaC-Tools für andere Ressourcen beibehalten.

ACK unterstützt auch schreibgeschützte Ressourcen. Bei Ressourcen, die von anderen Teams oder Tools verwaltet werden und die Sie referenzieren, aber nicht ändern möchten, kombinieren Sie die Übernahme mit der `retain` Löschrichtlinie und gewähren Sie nur IAM-Leseberechtigungen. Auf diese Weise können Anwendungen gemeinsam genutzte Infrastrukturen (VPCs, IAM-Rollen, KMS-Schlüssel) über Kubernetes APIs erkennen, ohne Änderungen riskieren zu müssen.

Weitere Informationen zur Nutzung von Ressourcen finden Sie unter. the section called “ACK-Konzepte”

Richtlinien zum Löschen

Löschrichtlinien steuern, was mit AWS Ressourcen passiert, wenn Sie die entsprechende Kubernetes-Ressource löschen. Wählen Sie die richtige Richtlinie basierend auf dem Ressourcenlebenszyklus und Ihren betrieblichen Anforderungen.

Löschen (Standard)

Die AWS Ressource wird gelöscht, wenn Sie die Kubernetes-Ressource löschen. Dadurch wird die Konsistenz zwischen Ihrem Cluster aufrechterhalten und sichergestellt AWS, dass sich keine Ressourcen ansammeln.

Wann sollte Delete verwendet werden:

- Entwicklungs- und Testumgebungen, in denen Bereinigung wichtig ist
- Kurzlebige Ressourcen, die an den Anwendungslebenszyklus gebunden sind (Testdatenbanken, temporäre Buckets)
- Ressourcen, die die Anwendung nicht überdauern sollten (SQS-Warteschlangen, Cluster) ElastiCache
- Kostenoptimierung — automatische Bereinigung ungenutzter Ressourcen
- Umgebungen, die mit verwaltet werden GitOps , in denen das Entfernen von Ressourcen aus Git die Infrastruktur löschen sollte

Die standardmäßige Löschrichtlinie entspricht dem deklarativen Modell von Kubernetes: Was sich im Cluster befindet, entspricht dem, was sich darin befindet. AWS

Beibehalten

Die AWS Ressource wird beibehalten, wenn Sie die Kubernetes-Ressource löschen. Dies schützt wichtige Daten und ermöglicht es Ressourcen, ihre Kubernetes-Repräsentation zu überleben.

Wann sollte Retain verwendet werden:

- Produktionsdatenbanken mit kritischen Daten, die Clusteränderungen überstehen müssen
- Langfristige Speicherbereiche mit Compliance- oder Prüfanforderungen

- Gemeinsam genutzte Ressourcen, die von mehreren Anwendungen oder Teams genutzt werden
- Ressourcen werden auf verschiedene Verwaltungstools migriert
- Notfallwiederherstellungsszenarien, in denen Sie die Infrastruktur erhalten möchten
- Ressourcen mit komplexen Abhängigkeiten, die eine sorgfältige Außerbetriebnahme erfordern

```
apiVersion: rds.services.k8s.aws/v1alpha1
kind: DBInstance
metadata:
  name: production-db
  annotations:
    services.k8s.aws/deletion-policy: "retain"
spec:
  dbInstanceIdentifier: prod-db
  # ... configuration
```

Important

Zurückbehaltene Ressourcen sind weiterhin mit AWS Kosten verbunden und müssen manuell gelöscht werden, AWS wenn sie nicht mehr benötigt werden. Verwenden Sie das Ressourcen-Tagging, um die zurückbehaltenen Ressourcen für die Bereinigung nachzuverfolgen.

Weitere Informationen zu Löschrichtlinien finden Sie unter. the section called “ACK-Konzepte”

Upstream-Dokumentation

Ausführliche Informationen zur Verwendung von ACK finden Sie unter

- ACK-Nutzungshandbuch — Ressourcen erstellen und verwalten
- ACK-API-Referenz — Vollständige API-Dokumentation für alle Dienste
- ACK-Dokumentation — Umfassende Benutzerdokumentation

Nächste Schritte

- the section called “Konfigurieren von -Berechtigungen”- Konfigurieren Sie IAM-Berechtigungen und Muster für mehrere Konten

- the section called “ACK-Konzepte”- Verstehen Sie die ACK-Konzepte und den Ressourcenlebenszyklus
- the section called “Fehlerbehebung”- Beheben Sie ACK-Probleme
- the section called “Arbeiten mit Argo CD”- Stellen Sie ACK-Ressourcen bereit mit GitOps
- the section called “Kro-Konzepte”- Stellen Sie ACK-Ressourcen zu Abstraktionen auf höherer Ebene zusammen

Probleme mit ACK-Funktionen beheben

Dieses Thema enthält Anleitungen zur Fehlerbehebung für die EKS-Funktion für ACK, einschließlich Funktionsprüfungen, Überprüfung des Ressourcenstatus und Probleme mit IAM-Berechtigungen.

> **Note**
>
> Die EKS-Funktionen werden vollständig verwaltet und außerhalb Ihres Clusters ausgeführt. Sie haben keinen Zugriff auf Controller-Logs oder Controller-Namespaces. Die Fehlerbehebung konzentriert sich auf den Funktionsstatus, den Ressourcenstatus und die IAM-Konfiguration.

Die Fähigkeit ist AKTIV, aber es werden keine Ressourcen erstellt

Wenn Ihre ACK-Fähigkeit `ACTIVE` den Status anzeigt, aber keine Ressourcen erstellt werden AWS, überprüfen Sie den Funktionsstatus, den Ressourcenstatus und die IAM-Berechtigungen.

Überprüfen Sie den Zustand der Fähigkeit:

Sie können Funktionszustands- und Statusprobleme in der EKS-Konsole oder über die AWS CLI anzeigen.

Konsole:

1. Öffnen Sie die Amazon EKS-Konsole unter https://console.aws.amazon.com/eks/ home#/clusters.
2. Wählen Sie Ihren Clusternamen aus.
3. Wählen Sie den Registerkarte Beobachtbarkeit.
4. Wählen Sie Cluster überwachen aus.

5. Wählen Sie die Registerkarte Funktionen, um den Zustand und den Status aller Funktionen anzuzeigen.

AWS CLI:

```
# View capability status and health
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-ack

# Look for issues in the health section
```

Häufige Ursachen:

- IAM-Berechtigungen fehlen: Der Capability-Rolle fehlen die Berechtigungen für den Dienst AWS
- Falscher Namespace: Ressourcen, die im Namespace ohne den richtigen Selektor erstellt wurden IAMRole
- Ungültige Ressourcenspezifikation: Überprüfen Sie die Bedingungen für den Ressourcenstatus auf Validierungsfehler
- API-Drosselung: AWS API-Ratenlimits wurden erreicht
- Zulassungs-Webhooks: Zulassungs-Webhooks, die den Controller daran hindern, den Ressourcenstatus zu patchen

Überprüfen Sie den Ressourcenstatus:

```
# Describe the resource to see conditions and events
kubectl describe bucket my-bucket -n default

# Look for status conditions
kubectl get bucket my-bucket -n default -o jsonpath='{.status.conditions}'

# View resource events
kubectl get events --field-selector involvedObject.name=my-bucket -n default
```

Überprüfen Sie die IAM-Berechtigungen:

```
# View the Capability Role's policies
```

```
aws iam list-attached-role-policies --role-name my-ack-capability-role
aws iam list-role-policies --role-name my-ack-capability-role

# Get specific policy details
aws iam get-role-policy --role-name my-ack-capability-role --policy-name policy-name
```

Ressourcen, die in Kubernetes erstellt wurden AWS , aber nicht in Kubernetes angezeigt werden

ACK verfolgt nur Ressourcen, die es über Kubernetes-Manifeste erstellt. Verwenden Sie die Adoption-Funktion, um vorhandene AWS Ressourcen mit ACK zu verwalten.

```
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: existing-bucket
  annotations:
    services.k8s.aws/adoption-policy: "adopt-or-create"
spec:
  name: my-existing-bucket-name
```

Weitere Informationen zur Nutzung von Ressourcen finden Sie unterthe section called “ACK-Konzepte”.

Kontoübergreifende Ressourcen werden nicht erstellt

Wenn bei der Verwendung von IAM-Rollenselektoren keine Ressourcen in einem AWS Zielkonto erstellt werden, überprüfen Sie die Vertrauensstellung und IAMRole die Auswahlkonfiguration.

Überprüfen Sie die Vertrauensbeziehung:

```
# Check the trust policy in the target account role
aws iam get-role --role-name cross-account-ack-role --query
 'Role.AssumeRolePolicyDocument'
```

Die Vertrauensrichtlinie muss es der Capability Role des Quellkontos ermöglichen, diese Rolle zu übernehmen.

Bestätigen Sie die IAMRole Selector-Konfiguration:

```
# List IAMRoleSelectors (cluster-scoped)
kubectl get iamroleselector

# Describe specific selector
kubectl describe iamroleselector my-selector
```

Überprüfen Sie die Namespace-Ausrichtung:

IAMRoleSelektoren sind Ressourcen im Clusterbereich, die jedoch auf bestimmte Namespaces abzielen. Stellen Sie sicher, dass sich Ihre ACK-Ressourcen in einem Namespace befinden, der dem Namespace-Selektor des Selectors entspricht: IAMRole

```
# Check resource namespace
kubectl get bucket my-cross-account-bucket -n production

# List all IAMRoleSelectors (cluster-scoped)
kubectl get iamroleselector

# Check which namespace the selector targets
kubectl get iamroleselector my-selector -o jsonpath='{.spec.namespaceSelector}'
```

Ausgewählte Bedingung überprüfen: IAMRole

Stellen Sie sicher, dass der IAMRole Selector erfolgreich Ihrer Ressource zugeordnet wurde, indem Sie die folgende `ACK.IAMRoleSelected` Bedingung überprüfen:

```
# Check if IAMRoleSelector was matched
kubectl get bucket my-cross-account-bucket -n production -o
 jsonpath='{.status.conditions[?(@.type=="ACK.IAMRoleSelected")]}'
```

Wenn die Bedingung zutrifft `False` oder fehlt, stimmt der IAMRole Namespace-Selektor des Selectors nicht mit dem Namespace der Ressource überein. Stellen Sie sicher, dass die des Selektors mit den `namespaceSelector` Namespace-Labels Ihrer Ressource übereinstimmen.

Überprüfen Sie die Berechtigungen für Capability Role:

Die Anforderungen `sts:AssumeRole` und `sts:TagSession` Berechtigungen der Capability Role für die Rolle des Zielkontos:

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["sts:AssumeRole", "sts:TagSession"],
      "Resource": "arn:aws:iam::[.replaceable]`444455556666`:role/[.replaceable]`cross-
account-ack-role`"
    }
  ]
}
```

Eine ausführliche kontenübergreifende Konfiguration finden Sie unterthe section called “Konfigurieren von -Berechtigungen”.

Nächste Schritte

- the section called “Überlegungen”— Überlegungen zu ACK und bewährte Methoden
- the section called “Konfigurieren von -Berechtigungen”— Konfigurieren Sie IAM-Berechtigungen und Muster für mehrere Konten
- the section called “ACK-Konzepte”- Verstehen Sie die ACK-Konzepte und den Ressourcenlebenszyklus
- the section called “Problembehebung bei Funktionen”- Allgemeine Hinweise zur Problembehebung

Vergleich der EKS-Funktionen für ACK mit selbstverwaltetem ACK

Die EKS-Funktion für ACK bietet dieselbe Funktionalität wie selbstverwaltete ACK-Controller, bietet jedoch erhebliche betriebliche Vorteile. Einen allgemeinen Vergleich zwischen EKS-Funktionen und selbstverwalteten Lösungen finden Sie unter. the section called “Überlegungen” Dieses Thema konzentriert sich auf ACK-spezifische Unterschiede.

Unterschiede zu Upstream-ACK

Die EKS-Funktion für ACK basiert auf vorgelagerten ACK-Controllern, unterscheidet sich jedoch in der IAM-Integration.

IAM-Fähigkeitsrolle: Die Funktion verwendet eine dedizierte IAM-Rolle mit einer Vertrauensrichtlinie, die den `capabilities.eks.amazonaws.com` Dienstprinzipal und nicht IRSA (IAM-Rollen für Dienstkonten) zulässt. Sie können IAM-Richtlinien direkt an die Capability Role anhängen, ohne Kubernetes-Dienstkonten erstellen oder mit Anmerkungen versehen oder OIDC-Anbieter

konfigurieren zu müssen. Eine bewährte Methode für Anwendungsfälle in der Produktion ist die Konfiguration von Serviceberechtigungen mithilfe von. `IAMRoleSelector` Weitere Details finden Sie unter the section called “Konfigurieren von -Berechtigungen”.

Sitzungs-Tags: Die verwaltete Funktion legt automatisch Sitzungs-Tags für alle AWS API-Anfragen fest und ermöglicht so eine differenzierte Zugriffskontrolle und Prüfung. Zu den Tags gehören `eks:eks-capability-arneks:kubernetes-namespace`, und. `eks:kubernetes-api-group` Dies unterscheidet sich von selbstverwaltetem ACK, bei dem diese Tags nicht standardmäßig festgelegt werden. Einzelheiten the section called “Konfigurieren von -Berechtigungen” zur Verwendung von Sitzungs-Tags in IAM-Richtlinien finden Sie unter.

Ressourcen-Tags: Diese Funktion wendet andere Standardtags auf AWS Ressourcen an als selbstverwaltetes ACK. Die Funktion verwendet Tags mit `eks:` Präfixen (z. B.`eks:kubernetes-namespace`,`eks:eks-capability-arn`) anstelle der Tags, die von selbstverwaltetem `services.k8s.aws/` ACK verwendet werden. Die the section called “Überlegungen” vollständige Liste der Standard-Ressourcen-Tags finden Sie unter.

Ressourcenkompatibilität: Benutzerdefinierte ACK-Ressourcen funktionieren genauso wie Upstream-ACK, ohne dass Ihre ACK-Ressourcen-YAML-Dateien geändert werden. Die Funktion verwendet dasselbe Kubernetes APIs und CRDs daher `kubectl` funktionieren Tools wie auf die gleiche Weise. Alle GA-Controller und Ressourcen aus dem Upstream-ACK werden unterstützt.

Die vollständige ACK-Dokumentation und dienstspezifische Anleitungen finden Sie in der ACK-Dokumentation.

Migrationspfad

Sie können ohne Ausfallzeiten vom selbstverwalteten ACK zur verwalteten Funktion migrieren:

1. Aktualisieren Sie Ihren selbstverwalteten ACK-Controller so, dass er ihn `kube-system` für Leasingverträge zur Wahl von Führungskräften verwenden kann, zum Beispiel:

   ```
   helm upgrade --install ack-s3-controller \
     oci://public.ecr.aws/aws-controllers-k8s/s3-chart \
     --namespace ack-system \
     --set leaderElection.namespace=kube-system
   ```

   Dadurch wird der Leasingvertrag des Controllers auf diesen `kube-system` übertragen, sodass sich die verwaltete Kapazität darauf abstimmen kann.

2. Erstellen Sie die ACK-Fähigkeit auf Ihrem Cluster (siehethe section called “ACK-Fähigkeit erstellen”)
3. Die verwaltete Funktion erkennt vorhandene, von ACK verwaltete AWS Ressourcen und übernimmt den Abgleich
4. Skalieren Sie selbstverwaltete Controller-Bereitstellungen schrittweise oder entfernen Sie sie:

```
helm uninstall ack-s3-controller --namespace ack-system
```

Dieser Ansatz ermöglicht eine sichere Koexistenz beider Controller während der Migration. Die verwaltete Funktion übernimmt automatisch Ressourcen, die zuvor von selbstverwalteten Controllern verwaltet wurden, und gewährleistet so einen kontinuierlichen Abgleich ohne Konflikte.

Nächste Schritte

- the section called “ACK-Fähigkeit erstellen”- Erstellen Sie eine ACK-Fähigkeitsressource
- the section called “ACK-Konzepte”- Verstehen Sie die ACK-Konzepte und den Ressourcenlebenszyklus
- the section called “Konfigurieren von -Berechtigungen”- Konfigurieren Sie IAM und Berechtigungen

Kontinuierlicher Einsatz mit Argo CD

Argo CD ist ein deklaratives Tool zur GitOps kontinuierlichen Bereitstellung für Kubernetes. Mit Argo CD können Sie die Bereitstellung und das Lebenszyklusmanagement Ihrer Anwendungen in mehreren Clustern und Umgebungen automatisieren. Argo CD unterstützt mehrere Quelltypen, darunter Git-Repositorys, Helm-Registries (HTTP und OCI) und OCI-Images, was Unternehmen mit unterschiedlichen Sicherheits- und Compliance-Anforderungen Flexibilität bietet.

Dank der EKS-Funktionen wird Argo CD vollständig verwaltet AWS, sodass Sie keine Argo-CD-Controller und deren Abhängigkeiten von Ihren Clustern installieren, warten und skalieren müssen.

Wie funktioniert Argo CD

Argo CD folgt dem GitOps Muster, bei dem Ihre Anwendungsquelle (Git-Repository, Helm-Registry oder OCI-Image) die Quelle der Wahrheit für die Definition des gewünschten Anwendungsstatus ist. Wenn Sie eine `Application` Argo-CD-Ressource erstellen, geben Sie die Quelle an, die Ihre Anwendungsmanifeste enthält, sowie den Kubernetes-Zielcluster und -Namespace. Argo CD

überwacht kontinuierlich sowohl den Quell- als auch den Live-Status im Cluster und synchronisiert automatisch alle Änderungen, um sicherzustellen, dass der Clusterstatus dem gewünschten Status entspricht.

> **Note**
>
> Mit der EKS-Funktion für Argo CD läuft die Argo-CD-Software auf der AWS Steuerungsebene, nicht auf Ihren Worker-Knoten. Das bedeutet, dass deine Worker-Knoten keinen direkten Zugriff auf Git-Repositorys oder Helm-Registries benötigen — die Funktion wickelt den Quellzugriff vom Konto aus ab. AWS

Argo CD bietet drei primäre Ressourcentypen:

- Anwendung: Definiert eine Bereitstellung von einem Git-Repository in einem Zielcluster
- ApplicationSet: Generiert mehrere Anwendungen aus Vorlagen für Bereitstellungen mit mehreren Clustern
- AppProject: Bietet logische Gruppierung und Zugriffskontrolle für Anwendungen

Beispiel: Eine Argo-CD-Anwendung erstellen

Das folgende Beispiel zeigt, wie eine `Application` Argo-CD-Ressource erstellt wird:

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: guestbook
  namespace: argocd
spec:
  project: default
  source:
    repoURL: https://github.com/argoproj/argocd-example-apps.git
    targetRevision: HEAD
    path: guestbook
  destination:
    name: in-cluster
    namespace: guestbook
  syncPolicy:
    automated:
      prune: true
```

```
    selfHeal: true
```

> **Note**
>
> Verwenden Sie es `destination.name` zusammen mit dem Clusternamen, den Sie bei der Registrierung des Clusters verwendet haben (wie `in-cluster` für den lokalen Cluster). Das `destination.server` Feld funktioniert auch mit EKS-Clustern ARNs, aus Gründen der besseren Lesbarkeit wird jedoch die Verwendung von Clusternamen empfohlen.

Vorteile von Argo CD

Argo CD implementiert einen GitOps Workflow, bei dem Sie Ihre Anwendungskonfigurationen in Git-Repositorys definieren und Argo CD Ihre Anwendungen automatisch synchronisiert, damit sie dem gewünschten Status entsprechen. Dieser GIT-zentrierte Ansatz bietet einen vollständigen Prüfpfad aller Änderungen, ermöglicht einfache Rollbacks und lässt sich auf natürliche Weise in Ihre bestehenden Codeüberprüfungs- und Genehmigungsprozesse integrieren. Argo CD erkennt automatisch Abweichungen zwischen dem gewünschten Status in Git und dem tatsächlichen Status in Ihren Clustern und gleicht sie ab, um sicherzustellen, dass Ihre Bereitstellungen mit Ihrer deklarierten Konfiguration konsistent bleiben.

Mit Argo CD können Sie Anwendungen in mehreren Clustern von einer einzigen Argo-CD-Instanz aus bereitstellen und verwalten, was den Betrieb in Umgebungen mit mehreren Clustern und mehreren Regionen vereinfacht. Die Benutzeroberfläche von Argo CD bietet Visualisierungs- und Überwachungsfunktionen, mit denen Sie den Bereitstellungsstatus, den Zustand und den Verlauf Ihrer Anwendungen einsehen können. Die Benutzeroberfläche ist in AWS Identity Center (ehemals AWS SSO) integriert und ermöglicht so eine nahtlose Authentifizierung und Autorisierung, sodass Sie den Zugriff mithilfe Ihrer vorhandenen Identitätsmanagement-Infrastruktur kontrollieren können.

Als Teil von EKS Managed Capabilities wird Argo CD vollständig von verwaltet AWS, sodass die Installation, Konfiguration und Wartung der Argo-CD-Infrastruktur entfällt. AWS kümmert sich um Skalierung, Patching und Betriebsmanagement, sodass sich Ihre Teams auf die Anwendungsbereitstellung statt auf die Wartung der Tools konzentrieren können.

Integration mit AWS Identity Center

EKS Managed Capabilities bietet eine direkte Integration zwischen Argo CD und AWS Identity Center und ermöglicht so eine nahtlose Authentifizierung und Autorisierung für Ihre Benutzer. Wenn Sie die

Argo CD-Funktion aktivieren, können Sie die AWS Identity Center-Integration so konfigurieren, dass Identity Center-Gruppen und -Benutzer den Argo CD RBAC-Rollen zugeordnet werden, sodass Sie kontrollieren können, wer auf Anwendungen in Argo CD zugreifen und diese verwalten kann.

Integration mit anderen von EKS verwalteten Funktionen

Argo CD lässt sich in andere von EKS verwaltete Funktionen integrieren.

- AWS Controller für Kubernetes (ACK): Verwenden Sie Argo CD, um die Bereitstellung von ACK-Ressourcen in mehreren Clustern zu verwalten und GitOps Workflows für Ihre Infrastruktur zu ermöglichen. AWS
- kro (Kube Resource Orchestrator): Verwenden Sie Argo CD, um Kro-Kompositionen in mehreren Clustern bereitzustellen und so eine konsistente Ressourcenzusammensetzung in Ihrem gesamten Kubernetes-Bestand zu ermöglichen.

Erste Schritte mit Argo CD

Um mit der EKS-Funktion für Argo CD zu beginnen:

1. Erstellen und konfigurieren Sie eine IAM-Capability-Rolle mit den erforderlichen Berechtigungen, damit Argo CD auf Ihre Quellen zugreifen und Anwendungen verwalten kann.
2. Erstellen Sie eine Argo-CD-Capability-Ressource auf Ihrem EKS-Cluster über die AWS Konsole, AWS CLI oder Ihr bevorzugtes Infrastructure-as-Code-Tool.
3. Konfigurieren Sie den Repository-Zugriff und registrieren Sie Cluster für die Anwendungsbereitstellung.
4. Erstellen Sie Anwendungsressourcen, um Ihre Anwendungen aus Ihren deklarativen Quellen bereitzustellen.

Erstellen einer Argo CD-Funktion

In diesem Thema wird erklärt, wie Sie eine Argo-CD-Funktion auf Ihrem Amazon EKS-Cluster erstellen.

Voraussetzungen

Bevor Sie eine Argo-CD-Funktion erstellen, stellen Sie sicher, dass Sie über Folgendes verfügen:

- Ein vorhandener Amazon EKS-Cluster, auf dem eine unterstützte Kubernetes-Version ausgeführt wird (alle Versionen mit Standard- und erweitertem Support werden unterstützt)
- AWS Identity Center konfiguriert — Für die Argo-CD-Authentifizierung erforderlich (lokale Benutzer werden nicht unterstützt)
- Eine IAM-Fähigkeitsrolle mit Berechtigungen für Argo CD
- Ausreichende IAM-Berechtigungen zum Erstellen von Funktionsressourcen auf EKS-Clustern
- `kubectl`für die Kommunikation mit Ihrem Cluster konfiguriert
- (Optional) Die Argo-CD-CLI wurde für eine einfachere Cluster- und Repositoryverwaltung installiert
- (Für CLI/eksCTL) Das entsprechende CLI-Tool wurde installiert und konfiguriert

Anweisungen zum Erstellen der IAM-Capability-Rolle finden Sie unter. the section called “Funktion: IAM-Rolle” Informationen zur Einrichtung von Identity Center finden Sie unter Erste Schritte mit AWS Identity Center.

> **Important**
>
> Die von Ihnen angegebene IAM-Fähigkeitsrolle bestimmt, auf welche AWS Ressourcen Argo CD zugreifen kann. Dies beinhaltet den Zugriff auf Git-Repositorys über CodeConnections und Secrets in Secrets Manager. Anleitungen zur Erstellung einer geeigneten Rolle mit den geringsten Rechten finden Sie unter und. the section called “Funktion: IAM-Rolle” the section called “Überlegungen zu den Funktionen von EKS”

Wählen Sie Ihr Tool

Sie können eine Argo-CD-Funktion mit der AWS-Managementkonsole AWS CLI oder eksctl erstellen:

- the section called “Konsole”- Verwenden Sie die Konsole für ein geführtes Erlebnis
- the section called “ AWS CLI”- Verwenden Sie die AWS CLI für Scripting und Automatisierung
- the section called “eksctl”- Verwenden Sie eksctl für ein Kubernetes-natives Erlebnis

Was passiert, wenn Sie eine Argo-CD-Funktion erstellen

Wenn Sie eine Argo-CD-Funktion erstellen:

1. EKS erstellt den Argo-CD-Capability Service auf der Steuerungsebene AWS

2. Benutzerdefinierte Ressourcendefinitionen (CRDs) sind in Ihrem Cluster installiert
3. Argo CD beginnt, nach seinen benutzerdefinierten Ressourcen (Anwendungen, ApplicationSets, AppProjects) zu suchen
4. Der Funktionsstatus ändert sich von zu `CREATING` `ACTIVE`
5. Auf die Benutzeroberfläche von Argo CD kann über ihre URL zugegriffen werden

Sobald sie aktiv sind, können Sie Argo-CD-Anwendungen in Ihrem Cluster erstellen, um sie aus Ihren deklarativen Quellen bereitzustellen.

Nächste Schritte

Nach der Erstellung der Argo-CD-Funktion:

- the section called “Argo-CD-Konzepte”- Erfahren Sie mehr über GitOps Prinzipien, Synchronisierungsrichtlinien und Multi-Cluster-Muster
- the section called “Arbeiten mit Argo CD”- Konfigurieren Sie den Repository-Zugriff, registrieren Sie Zielcluster und erstellen Sie Anwendungen
- the section called “Überlegungen zu Argo CD”- Erkunden Sie Architekturmuster und erweiterte Konfigurationen mit mehreren Clustern

Erstellen Sie mit der Konsole eine Argo-CD-Funktion

In diesem Thema wird beschrieben, wie Sie mit dem eine Argo-CD-Funktion erstellen. AWS-Managementkonsole

Voraussetzungen

- AWS Identity Center konfiguriert — Argo CD benötigt AWS Identity Center für die Authentifizierung. Lokale Benutzer werden nicht unterstützt. Wenn Sie AWS Identity Center nicht eingerichtet haben, finden Sie weitere Informationen unter Erste Schritte mit AWS Identity Center, um eine Identity Center-Instanz zu erstellen, und Benutzer hinzufügen und Gruppen hinzufügen, um Benutzer und Gruppen für den Zugriff auf Argo CD zu erstellen.

Erstellen Sie die Argo-CD-Funktion

1. Öffnen Sie die Amazon EKS-Konsole unter https://console.aws.amazon.com/eks/ home#/clusters.

2. Wählen Sie Ihren Clusternamen aus, um die Cluster-Detailseite zu öffnen.
3. Wählen Sie die Registerkarte Funktionen.
4. Wählen Sie in der linken Navigationsleiste Argo CD.
5. Wählen Sie die Funktion „Argo-CD erstellen“.
6. Für die IAM-Fähigkeitsrolle:
   - Wenn Sie bereits über eine IAM-Fähigkeitsrolle verfügen, wählen Sie sie aus der Dropdownliste aus
   - Wenn Sie eine Rolle erstellen müssen, wählen Sie Create Argo CD role

     Dadurch wird die IAM-Konsole auf einer neuen Registerkarte mit vorab ausgefüllten Vertrauensrichtlinien und uneingeschränktem Lesezugriff auf Secrets Manager geöffnet. Standardmäßig werden keine weiteren Berechtigungen hinzugefügt, aber Sie können sie bei Bedarf hinzufügen. Wenn Sie CodeCommit Repositorys oder andere AWS Dienste verwenden möchten, fügen Sie die entsprechenden Berechtigungen hinzu, bevor Sie die Rolle erstellen.

     Kehren Sie nach dem Erstellen der Rolle zur EKS-Konsole zurück. Die Rolle wird automatisch ausgewählt.

     > **Note**
     >
     > Wenn Sie die optionalen Integrationen mit AWS Secrets Manager oder verwenden möchten AWS CodeConnections, müssen Sie der Rolle Berechtigungen hinzufügen. Beispiele für IAM-Richtlinien und Anleitungen zur Konfiguration finden Sie unter the section called “ AWS Secrets Manager” und. the section called “ AWS CodeConnections”
7. Konfigurieren Sie die AWS Identity Center-Integration:
   a. Wählen Sie AWS Identity Center-Integration aktivieren aus.
   b. Wählen Sie Ihre Identity Center-Instanz aus der Dropdownliste aus.
   c. Konfigurieren Sie Rollenzuordnungen für RBAC, indem Sie Benutzer oder Gruppen den Argo-CD-Rollen (ADMIN, EDITOR oder VIEWER) zuweisen
8. Wählen Sie Erstellen aus.

Der Prozess zur Erstellung von Fähigkeiten beginnt.

Stellen Sie sicher, dass die Fähigkeit aktiv ist

1. Sehen Sie sich auf der Registerkarte Funktionen den Argo-CD-Funktionsstatus an.
2. Warten Sie, bis sich der Status von `CREATING` zu `ACTIVE` ändert.
3. Sobald die Funktion aktiv ist, ist sie einsatzbereit.

Informationen zum Status der Funktionen und zur Problembehandlung finden Sie unterthe section called “Mit Fähigkeiten arbeiten”.

Greifen Sie auf die Benutzeroberfläche von Argo CD zu

Nachdem die Funktion aktiv ist, können Sie auf die Benutzeroberfläche von Argo CD zugreifen:

1. Wählen Sie auf der Argo-CD-Funktionsseite die Option Open Argo CD UI.
2. Die Benutzeroberfläche von Argo CD wird in einem neuen Browser-Tab geöffnet.
3. Sie können jetzt über die Benutzeroberfläche Anwendungen erstellen und Bereitstellungen verwalten.

Nächste Schritte

- the section called “Arbeiten mit Argo CD”- Konfigurieren Sie Repositorys, registrieren Sie Cluster und erstellen Sie Anwendungen
- the section called “Überlegungen zu Argo CD”- Multi-Cluster-Architektur und erweiterte Konfiguration
- the section called “Mit Fähigkeiten arbeiten”- Verwalten Sie Ihre Argo-CD-Funktionsressource

Erstellen Sie eine Argo-CD-Funktion mit der CLI AWS

In diesem Thema wird beschrieben, wie Sie mit der AWS CLI eine Argo-CD-Funktion erstellen.

Voraussetzungen

- AWS CLI — Version `2.12.3` oder höher. Führen Sie den Befehl aus, um Ihre Version zu überprüfen`aws --version`. Weitere Informationen finden Sie im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle unter Installation.
- **`kubectl`** – Ein Befehlszeilentool für die Arbeit mit Kubernetes-Clustern. Weitere Informationen finden Sie unter the section called “kubectl und eksctl einrichten”.

- AWS Identity Center konfiguriert — Argo CD benötigt AWS Identity Center für die Authentifizierung. Lokale Benutzer werden nicht unterstützt. Wenn Sie AWS Identity Center nicht eingerichtet haben, finden Sie weitere Informationen unter Erste Schritte mit AWS Identity Center, um eine Identity Center-Instanz zu erstellen, und Benutzer hinzufügen und Gruppen hinzufügen, um Benutzer und Gruppen für den Zugriff auf Argo CD zu erstellen.

Schritt 1: Erstellen Sie eine IAM-Fähigkeitsrolle

Erstellen Sie eine Vertrauensrichtliniendatei:

```
cat > argocd-trust-policy.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "capabilities.eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
EOF
```

Erstellen Sie die IAM-Rolle:

```
aws iam create-role \
  --role-name ArgoCDCapabilityRole \
  --assume-role-policy-document file://argocd-trust-policy.json
```

> **Note**
>
> Wenn Sie die optionalen Integrationen mit AWS Secrets Manager oder verwenden möchten AWS CodeConnections, müssen Sie der Rolle Berechtigungen hinzufügen. Beispiele für

IAM-Richtlinien und Anleitungen zur Konfiguration finden Sie unter the section called “ AWS Secrets Manager” und. the section called “ AWS CodeConnections”

Schritt 2: Erstellen Sie die Argo-CD-Funktion

Erstellen Sie die Argo-CD-Capability-Ressource auf Ihrem Cluster.

Legen Sie zunächst Umgebungsvariablen für Ihre Identity Center-Konfiguration fest:

```
# Get your Identity Center instance ARN (replace region if your IDC instance is in a
 different region)
export IDC_INSTANCE_ARN=$(aws sso-admin list-instances --region [.replaceable]`region`
 --query 'Instances[0].InstanceArn' --output text)

# Get a user ID for RBAC mapping (replace with your username and region if needed)
export IDC_USER_ID=$(aws identitystore list-users \
  --region [.replaceable]`region` \
  --identity-store-id $(aws sso-admin list-instances --region [.replaceable]`region` --
query 'Instances[0].IdentityStoreId' --output text) \
  --query 'Users[?UserName==`your-username`].UserId' --output text)

echo "IDC_INSTANCE_ARN=$IDC_INSTANCE_ARN"
echo "IDC_USER_ID=$IDC_USER_ID"
```

Erstellen Sie die Funktion mit der Identity Center-Integration. *region-code*Ersetzen Sie durch die AWS Region, in der sich Ihr Cluster befindet, und *my-cluster* durch Ihren Clusternamen:

```
aws eks create-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-argocd \
  --type ARGOCD \
  --role-arn arn:aws:iam::$(aws sts get-caller-identity --query Account --output
 text):role/ArgoCDCapabilityRole \
  --delete-propagation-policy RETAIN \
  --configuration '{
    "argoCd": {
      "awsIdc": {
        "idcInstanceArn": "'$IDC_INSTANCE_ARN'",
        "idcRegion": "'[.replaceable]`region-code`'"
      },
```

```
      "rbacRoleMappings": [{
        "role": "ADMIN",
        "identities": [{
          "id": "'$IDC_USER_ID'",
          "type": "SSO_USER"
        }]
      }]
    }
  }'
```

Der Befehl wird sofort zurückgegeben, aber es dauert einige Zeit, bis die Funktion aktiv wird, da EKS die erforderliche Funktionsinfrastruktur und die erforderlichen Komponenten erstellt. EKS installiert die benutzerdefinierten Kubernetes-Ressourcendefinitionen, die sich auf diese Funktion beziehen, in Ihrem Cluster, während dieser erstellt wird.

> **Note**
>
> Wenn Sie eine Fehlermeldung erhalten, dass der Cluster nicht existiert oder Sie keine Berechtigungen haben, überprüfen Sie Folgendes:
>
> - Der Clustername ist korrekt
> - Ihre AWS CLI ist für die richtige Region konfiguriert
> - Sie verfügen über die erforderlichen IAM-Berechtigungen

Schritt 3: Stellen Sie sicher, dass die Funktion aktiv ist

Warten Sie, bis die Funktion aktiv wird. *region-code*Ersetzen Sie durch die AWS Region, in der sich Ihr Cluster befindet, und *my-cluster* durch Ihren Clusternamen.

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-argocd \
  --query 'capability.status' \
  --output text
```

Die Funktion ist bereit, wenn der Status angezeigt wird`ACTIVE`. Fahren Sie erst mit dem nächsten Schritt fort, wenn der Status lautet`ACTIVE`.

Sie können sich auch die vollständigen Funktionsdetails ansehen:

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-argocd
```

Schritt 4: Stellen Sie sicher, dass benutzerdefinierte Ressourcen verfügbar sind

Nachdem die Funktion aktiviert ist, stellen Sie sicher, dass benutzerdefinierte Argo CD-Ressourcen in Ihrem Cluster verfügbar sind:

```
kubectl api-resources | grep argoproj.io
```

Sie sollten die Liste der `ApplicationSet` Ressourcentypen sehen`Application`.

Nächste Schritte

- the section called “Arbeiten mit Argo CD”- Konfigurieren Sie Repositorys, registrieren Sie Cluster und erstellen Sie Anwendungen
- the section called “Überlegungen zu Argo CD”- Multi-Cluster-Architektur und erweiterte Konfiguration
- the section called “Mit Fähigkeiten arbeiten”- Verwalten Sie Ihre Argo-CD-Funktionsressource

Erstellen Sie eine Argo-CD-Funktion mit eksctl

In diesem Thema wird beschrieben, wie Sie mit eksctl eine Argo-CD-Funktion erstellen.

> **Note**
>
> Für die folgenden Schritte ist die Version eksctl oder höher erforderlich. `0.220.0` Führen Sie den Befehl aus, um Ihre Version zu überprüfen. `eksctl version`

Schritt 1: Erstellen Sie eine IAM-Capability-Rolle

Erstellen Sie eine Vertrauensrichtliniendatei:

```
cat > argocd-trust-policy.json << 'EOF'
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "capabilities.eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
EOF
```

Erstellen Sie die IAM-Rolle:

```
aws iam create-role \
  --role-name ArgoCDCapabilityRole \
  --assume-role-policy-document file://argocd-trust-policy.json
```

> **Note**
>
> Für diese grundlegende Einrichtung sind keine zusätzlichen IAM-Richtlinien erforderlich. Wenn Sie Secrets Manager für Repository-Anmeldeinformationen oder verwenden möchten CodeConnections, müssen Sie der Rolle Berechtigungen hinzufügen. Beispiele für IAM-Richtlinien und Anleitungen zur Konfiguration finden Sie unter the section called “ AWS Secrets Manager” undthe section called “ AWS CodeConnections”.

Schritt 2: Holen Sie sich Ihre AWS Identity Center-Konfiguration

Rufen Sie den ARN und die Benutzer-ID Ihrer Identity Center-Instanz für die RBAC-Konfiguration ab:

```
# Get your Identity Center instance ARN
aws sso-admin list-instances --query 'Instances[0].InstanceArn' --output text

# Get your Identity Center region
aws sso-admin list-instances --query 'Instances[0].IdentityStoreId' --output text | cut
 -d'/' -f1
```

```
# Get a user ID for admin access (replace 'your-username' with your Identity Center
 username)
aws identitystore list-users \
  --identity-store-id $(aws sso-admin list-instances --query
 'Instances[0].IdentityStoreId' --output text) \
  --query 'Users[?UserName==`your-username`].UserId' --output text
```

Notieren Sie sich diese Werte — Sie benötigen sie im nächsten Schritt.

Schritt 3: Erstellen Sie eine eksctl-Konfigurationsdatei

Erstellen Sie eine Datei mit dem Namen `argocd-capability.yaml` und dem folgenden Inhalt. Ersetzen Sie die Platzhalterwerte durch Ihren Clusternamen, Ihre Region, Ihren IAM-Rollen-ARN, Ihren Identity Center-Instanz-ARN, Ihre Identity Center-Region und Ihre Benutzer-ID:

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code

capabilities:
  - name: my-argocd
    type: ARGOCD
    roleArn: arn:aws:iam::[.replaceable]111122223333:role/ArgoCDCapabilityRole
    configuration:
      argocd:
        awsIdc:
          idcInstanceArn: arn:aws:sso:::instance/ssoins-123abc
          idcRegion: idc-region-code
        rbacRoleMappings:
          - role: ADMIN
            identities:
              - id: 38414300-1041-708a-01af-5422d6091e34
                type: SSO_USER
```

> **Note**
>
> Sie können den RBAC-Zuordnungen mehrere Benutzer oder Gruppen hinzufügen. Verwenden Sie für Gruppen die Gruppen-ID `type: SSO_GROUP` und geben Sie sie an. Verfügbare Rollen sind `ADMINEDITOR`, und`VIEWER`.

Schritt 4: Erstellen Sie die Argo-CD-Funktion

Wenden Sie die Konfigurationsdatei an:

```
eksctl create capability -f argocd-capability.yaml
```

Der Befehl kehrt sofort zurück, aber es dauert einige Zeit, bis die Funktion aktiv wird.

Schritt 5: Stellen Sie sicher, dass die Funktion aktiv ist

Überprüfen Sie den Status der Fähigkeit. *`region-code`*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet, und *`my-cluster`* ersetzen Sie es durch den Namen Ihres Clusters.

```
eksctl get capability \
  --region region-code \
  --cluster my-cluster \
  --name my-argocd
```

Die Funktion ist bereit, wenn der Status angezeigt wird`ACTIVE`.

Schritt 6: Stellen Sie sicher, dass benutzerdefinierte Ressourcen verfügbar sind

Nachdem die Funktion aktiviert ist, stellen Sie sicher, dass benutzerdefinierte Argo CD-Ressourcen in Ihrem Cluster verfügbar sind:

```
kubectl api-resources | grep argoproj.io
```

Sie sollten die Liste der `ApplicationSet` Ressourcentypen sehen`Application`.

Nächste Schritte

- the section called “Arbeiten mit Argo CD”- Erfahren Sie, wie Sie Argo-CD-Anwendungen erstellen und verwalten
- the section called “Überlegungen zu Argo CD”- Konfigurieren Sie SSO und Multi-Cluster-Zugriff

- the section called “Mit Fähigkeiten arbeiten”- Verwalten Sie Ihre Argo-CD-Funktionsressource

Argo CD-Konzepte

Argo CD implementiert, GitOps indem es Git als zentrale Informationsquelle für Ihre Anwendungsbereitstellungen behandelt. In diesem Thema wird ein praktisches Beispiel vorgestellt und anschließend die Kernkonzepte erläutert, die Sie bei der Arbeit mit der EKS-Funktion für Argo CD verstehen müssen.

Erste Schritte mit Argo CD

Nachdem Sie die Argo-CD-Funktion erstellt haben (siehethe section called “Erstellen der Argo CD-Funktion”), können Sie mit der Bereitstellung von Anwendungen beginnen. In diesem Beispiel werden die Registrierung eines Clusters und die Erstellung einer Anwendung beschrieben.

Schritt 1: Einrichten von

Registrieren Sie Ihren Cluster (erforderlich)

Registrieren Sie den Cluster, in dem Sie Anwendungen bereitstellen möchten. In diesem Beispiel registrieren wir denselben Cluster, auf dem Argo CD läuft (Sie können den Namen aus `in-cluster` Kompatibilitätsgründen mit den meisten Argo-CD-Beispielen verwenden):

```
# Get your cluster ARN
CLUSTER_ARN=$(aws eks describe-cluster \
  --name my-cluster \
  --query 'cluster.arn' \
  --output text)

# Register the cluster using Argo CD CLI
argocd cluster add $CLUSTER_ARN \
  --aws-cluster-name $CLUSTER_ARN \
  --name in-cluster \
  --project default
```

> **Note**
>
> Informationen zur Konfiguration der Argo-CD-CLI für die Verwendung mit der Argo-CD-Funktion in EKS finden Sie unter. the section called “Verwenden der Argo-CD-CLI mit der verwalteten Funktion”

Alternativ können Sie den Cluster mit einem Kubernetes Secret registrieren (Einzelheiten finden Sie unterthe section called “Registrieren Sie Cluster”).

Konfigurieren Sie den Repository-Zugriff (optional)

In diesem Beispiel wird ein öffentliches GitHub Repository verwendet, sodass keine Repository-Konfiguration erforderlich ist. Für private Repositorys konfigurieren Sie den Zugriff mit AWS Secrets Manager oder Kubernetes Secrets (weitere Informationen finden Sie unterthe section called “Repositorys konfigurieren”). CodeConnections

AWS Dienste (ECR für Helm-Diagramme und CodeCommit) können Sie direkt in den Anwendungsressourcen referenzieren, ohne ein Repository zu erstellen. CodeConnections Die Capability-Rolle muss über die erforderlichen IAM-Berechtigungen verfügen. Details dazu finden Sie unter the section called “Repositorys konfigurieren”.

Schritt 2: Eine Anwendung erstellen

Erstellen Sie dieses Anwendungsmanifest in`my-app.yaml`:

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: guestbook
  namespace: argocd
spec:
  project: default
  source:
    repoURL: https://github.com/argoproj/argocd-example-apps.git
    targetRevision: HEAD
    path: guestbook
  destination:
    name: in-cluster
    namespace: guestbook
  syncPolicy:
    automated:
      prune: true
      selfHeal: true
    syncOptions:
    - CreateNamespace=true
```

Wenden Sie die Anwendung an:

```
kubectl apply -f my-app.yaml
```

Nach dem Anwenden dieser Anwendung, Argo CD: 1. Synchronisiert die Anwendung von Git mit Ihrem Cluster (erste Bereitstellung) 2. Überwacht das Git-Repository auf Änderungen 3. Synchronisiert nachfolgende Änderungen automatisch mit Ihrem Cluster 4. Erkennt und korrigiert jede Abweichung vom gewünschten Zustand 5. Zeigt den Gesundheitsstatus und den Synchronisierungsverlauf auf der Benutzeroberfläche an

Den Status der Anwendung anzeigen:

```
kubectl get application guestbook -n argocd
```

Sie können die Anwendung auch über die Argo-CD-CLI oder die Argo-CD-Benutzeroberfläche anzeigen (zugänglich über die EKS-Konsole auf der Registerkarte Capabilities Ihres Clusters).

> **Note**
>
> Wenn Sie die Argo CD-CLI mit der verwalteten Funktion verwenden, geben Sie Anwendungen mit dem Namespace-Präfix an:. `argocd app get argocd/guestbook`

> **Note**
>
> Verwenden Sie den Clusternamen in `destination.name` (den Namen, den Sie bei der Registrierung des Clusters verwendet haben). Die verwaltete Funktion unterstützt nicht die lokale Standardeinstellung im Cluster (`kubernetes.default.svc`).

Schlüsselkonzepte

GitOps Prinzipien und Quelltypen

Argo CD implementiert GitOps, wobei Ihre Anwendungsquelle die zentrale Informationsquelle für Implementierungen ist:

- Deklarativ — Der gewünschte Status wird mithilfe von YAML-Manifesten, Helm-Diagrammen oder Kustomize-Overlays deklariert
- Versioniert — Jede Änderung wird anhand eines vollständigen Prüfprotokolls nachverfolgt

- Automatisiert — Argo CD überwacht kontinuierlich die Quellen und synchronisiert Änderungen automatisch
- Selbstheilung — Erkennt und korrigiert Abweichungen zwischen dem gewünschten und dem tatsächlichen Clusterstatus

Unterstützte Quelltypen:

- Git-Repositorien - GitHub, GitLab, Bitbucket, CodeCommit (HTTPS, SSH oder) CodeConnections
- Helm-Register — HTTP-Register (wie`https://aws.github.io/eks-charts`) und OCI-Register (wie) `public.ecr.aws`
- OCI-Images — Container-Images, die Manifeste oder Helm-Charts (ähnlich) enthalten `oci://registry-1.docker.io/user/my-app`

Diese Flexibilität ermöglicht es Unternehmen, Quellen auszuwählen, die ihren Sicherheits- und Compliance-Anforderungen entsprechen. Beispielsweise können Organisationen, die den Git-Zugriff von Clustern aus einschränken, ECR für Helm-Diagramme oder OCI-Images verwenden.

Weitere Informationen finden Sie in der Argo-CD-Dokumentation unter Anwendungsquellen.

Synchronisieren und Abgleichen

Argo CD überwacht kontinuierlich Ihre Quellen und Cluster, um Unterschiede zu erkennen und zu korrigieren:

1. Fragt Quellen nach Änderungen ab (Standard: alle 6 Minuten)
2. Vergleicht den gewünschten Status mit dem Clusterstatus
3. Markiert Anwendungen als `Synced` oder `OutOfSync`
4. Synchronisiert Änderungen automatisch (falls konfiguriert) oder wartet auf die manuelle Genehmigung
5. Überwacht den Zustand der Ressourcen nach der Synchronisierung

Synchronisierungswellen steuern die Reihenfolge der Ressourcenerstellung mithilfe von Anmerkungen:

```
metadata:
  annotations:
```

```
    argocd.argoproj.io/sync-wave: "0"  # Default if not specified
```

Ressourcen werden in der Reihenfolge der Wellen angewendet (niedrigere Zahlen zuerst, einschließlich negativer Zahlen wie-1). Wave `0` ist die Standardeinstellung, sofern sie nicht angegeben ist. Auf diese Weise können Sie Abhängigkeiten wie Namespaces (Wave-1) vor Bereitstellungen (Wave) vor Diensten (Wave0) erstellen. 1

Durch die Selbstheilung werden manuelle Änderungen automatisch rückgängig gemacht:

```
spec:
  syncPolicy:
    automated:
      selfHeal: true
```

> **Note**
>
> Die verwaltete Funktion verwendet eine auf Anmerkungen basierende Ressourcenverfolgung (nicht labelbasiert), um die Kompatibilität mit Kubernetes-Konventionen und anderen Tools zu verbessern.

Ausführliche Informationen zu Synchronisierungsphasen, Hooks und erweiterten Mustern finden Sie in der Argo CD-Synchronisierungsdokumentation.

Integrität der Anwendung

Argo CD überwacht den Zustand aller Ressourcen in Ihrer Anwendung:

Integritätsstatus: * Fehlerfrei — Alle Ressourcen laufen wie erwartet * Fortschritt — Ressourcen werden erstellt oder aktualisiert * Heruntergestuft — Einige Ressourcen sind nicht fehlerfrei (Pods stürzen ab, Jobs schlagen fehl) * Suspendiert — Anwendung wurde absichtlich angehalten * Fehlt — In Git definierte Ressourcen sind nicht im Cluster vorhanden

Argo CD verfügt über integrierte Integritätsprüfungen für gängige Kubernetes-Ressourcen (Deployments StatefulSets, Jobs usw.) und unterstützt benutzerdefinierte Integritätsprüfungen für. CRDs

Der Zustand einer Anwendung wird durch all ihre Ressourcen bestimmt — wenn es eine Ressource gibt`Degraded`, ist es auch die Anwendung. `Degraded`

Weitere Informationen finden Sie unter Resource Health in der Argo-CD-Dokumentation.

Muster mit mehreren Clustern

Argo CD unterstützt zwei Hauptbereitstellungsmuster:

H ub-and-spoke — Führen Sie Argo CD auf einem dedizierten Management-Cluster aus, der auf mehreren Workload-Clustern bereitgestellt wird: * Zentrale Steuerung und Transparenz * Konsistente Richtlinien für alle Cluster * Eine Argo-CD-Instanz zur Verwaltung * Klare Trennung zwischen Steuerungsebene und Workloads

Pro Cluster — Führen Sie Argo CD auf jedem Cluster aus und verwalten Sie nur die Anwendungen dieses Clusters: * Clustertrennung (ein Fehler hat keine Auswirkungen auf andere) * Einfachere Vernetzung (keine clusterübergreifende Kommunikation) * Einfachere Ersteinrichtung (keine Clusterregistrierung)

hub-and-spokeEntscheiden Sie sich für Plattformteams, die viele Cluster verwalten, oder pro Cluster für unabhängige Teams oder wenn Cluster vollständig isoliert werden müssen.

Eine detaillierte Konfiguration mehrerer Cluster finden Sie unter. the section called “Überlegungen zu Argo CD”

Projekte

Projekte bieten logische Gruppierung und Zugriffskontrolle für Anwendungen:

- Quellenbeschränkungen — Beschränken Sie, welche Git-Repositorys verwendet werden können
- Zielbeschränkungen — Beschränken Sie, welche Cluster und Namespaces als Ziel ausgewählt werden können
- Ressourceneinschränkungen — Beschränken Sie, welche Kubernetes-Ressourcentypen bereitgestellt werden können
- RBAC-Integration — Ordnen Sie Projekte AWS Identity Center-Benutzern und Gruppen zu IDs

Anwendungen gehören zu einem einzigen Projekt. Wenn nicht angegeben, verwenden sie das `default` Projekt, für das standardmäßig keine Einschränkungen gelten. Bearbeiten Sie das `default` Projekt für den produktiven Einsatz, um den Zugriff einzuschränken, und erstellen Sie neue Projekte mit entsprechenden Einschränkungen.

Informationen zur Projektkonfiguration und zu RBAC-Mustern finden Sie unter. the section called “Konfigurieren von -Berechtigungen”

Synchronisierungsoptionen

Optimieren Sie das Synchronisierungsverhalten mit gängigen Optionen:

- `CreateNamespace=true`— Automatisch einen Ziel-Namespace erstellen
- `ServerSideApply=true`- Verwenden Sie serverseitiges Anwenden für eine bessere Konfliktlösung
- `SkipDryRunOnMissingResource=true`- Überspringe den Testlauf, wenn er noch CRDs nicht existiert (nützlich für Kro-Instanzen)

```
spec:
  syncPolicy:
    syncOptions:
    - CreateNamespace=true
    - ServerSideApply=true
    - SkipDryRunOnMissingResource=true
```

Eine vollständige Liste der Synchronisierungsoptionen finden Sie in der Dokumentation zu den Argo-CD-Synchronisierungsoptionen.

Nächste Schritte

- the section called “Repositorys konfigurieren”- Git-Repository-Zugriff konfigurieren
- the section called “Registrieren Sie Cluster”- Registrieren Sie Zielcluster für die Bereitstellung
- the section called “Anwendungen erstellen”- Erstellen Sie Ihre erste Anwendung
- the section called “Überlegungen zu Argo CD”- EKS-spezifische Muster, Identity Center-Integration und Multi-Cluster-Konfiguration
- Argo-CD-Dokumentation — Umfassende Argo-CD-Dokumentation mit Sync-Hooks, Integritätsprüfungen und erweiterten Mustern

Argo-CD-Berechtigungen konfigurieren

Die verwaltete Funktion von Argo CD ist zur Authentifizierung in AWS Identity Center integriert und verwendet integrierte RBAC-Rollen für die Autorisierung. In diesem Thema wird erklärt, wie Berechtigungen für Benutzer und Teams konfiguriert werden.

Wie funktionieren Berechtigungen mit Argo CD

Die Argo-CD-Funktion verwendet AWS Identity Center für die Authentifizierung und bietet drei integrierte RBAC-Rollen für die Autorisierung.

Wenn ein Benutzer auf Argo CD zugreift:

1. Sie authentifizieren sich mithilfe von AWS Identity Center (das eine Verbindung zu Ihrem Corporate Identity Provider herstellen kann)
2. AWS Identity Center stellt Benutzer- und Gruppeninformationen für Argo CD bereit
3. Argo CD ordnet Benutzer und Gruppen basierend auf Ihrer Konfiguration RBAC-Rollen zu
4. Benutzer sehen nur die Anwendungen und Ressourcen, für deren Zugriff sie berechtigt sind

Integrierte RBAC-Rollen

Die Argo-CD-Funktion bietet drei integrierte Rollen, die Sie AWS Identity Center-Benutzern und -Gruppen zuordnen. Dabei handelt es sich um Rollen mit globalem Geltungsbereich, die den Zugriff auf Argo CD-Ressourcen wie Projekte, Cluster und Repositorys steuern.

> **Important**
>
> Globale Rollen kontrollieren den Zugriff auf Argo CD selbst, nicht auf projektbezogene Ressourcen wie Anwendungen. EDITOR- und VIEWER-Benutzer können Anwendungen standardmäßig nicht sehen oder verwalten — sie benötigen Projektrollen, um auf projektbezogene Ressourcen zugreifen zu können. Einzelheiten the section called “Projektrollen und projektbezogener Zugriff” zur Gewährung des Zugriffs auf Anwendungen und andere projektbezogene Ressourcen finden Sie unter.

ADMINISTRATOR

Voller Zugriff auf alle Ressourcen und Einstellungen von Argo CD:

- Anwendungen in beliebigen Projekten erstellen, aktualisieren und ApplicationSets löschen
- Verwalten Sie die Argo-CD-Konfiguration
- Registrieren und verwalten Sie Bereitstellungszielcluster
- Konfigurieren Sie den Zugriff auf das Repository
- Projekte erstellen und verwalten

- Den gesamten Bewerbungsstatus und die Historie anzeigen
- Alle Cluster und Repositorys auflisten und darauf zugreifen

HERAUSGEBER

Kann Projekte aktualisieren und Projektrollen konfigurieren, aber die globalen Argo-CD-Einstellungen nicht ändern:

- Bestehende Projekte aktualisieren (Projekte können nicht erstellt oder gelöscht werden)
- Projektrollen und Berechtigungen konfigurieren
- Alle Cluster und Repositorys auflisten und darauf zugreifen
- Die globale Argo-CD-Konfiguration kann nicht geändert werden
- Cluster oder Repositorys können nicht direkt verwaltet werden
- Anwendungen ohne Projektrollen können nicht angezeigt oder verwaltet werden

BETRACHTER

Nur-Lese-Zugriff auf Argo-CD-Ressourcen:

- Projektkonfigurationen anzeigen
- Listet alle Projekte auf (einschließlich Projekten, denen der Benutzer nicht zugewiesen ist)
- Cluster oder Repositorys können nicht aufgelistet werden
- Es können keine Änderungen vorgenommen werden
- Anwendungen ohne Projektrollen können nicht angezeigt oder verwaltet werden

> **Note**
>
> Um EDITOR- oder VIEWER-Benutzern Zugriff auf Anwendungen zu gewähren, muss ein ADMIN oder EDITOR Projektrollen erstellen, die Identity Center-Gruppen bestimmten Berechtigungen innerhalb eines Projekts zuordnen.

Projektrollen und projektbezogener Zugriff

Globale Rollen (ADMIN, EDITOR, VIEWER) steuern den Zugriff auf Argo CD selbst. Projektrollen steuern den Zugriff auf Ressourcen und Funktionen innerhalb eines bestimmten Projekts, darunter:

- Ressourcen: Anwendungen ApplicationSets, Repository-Anmeldeinformationen, Cluster-Anmeldeinformationen
- Funktionen: Protokollzugriff, Zugriff für Führungskräfte auf Anwendungs-Pods

Grundlegendes zum zweistufigen Berechtigungsmodell:

- Globaler Geltungsbereich: Integrierte Rollen bestimmen, was Benutzer mit Projekten, Clustern, Repositorys und Argo-CD-Einstellungen tun können
- Projektumfang: Projektrollen bestimmen, was Benutzer mit Ressourcen und Fähigkeiten innerhalb eines bestimmten Projekts tun können

Das bedeutet Folgendes:

- ADMIN-Benutzer können ohne zusätzliche Konfiguration auf alle Projektressourcen und Funktionen zugreifen
- EDITOR- und VIEWER-Benutzern müssen Projektrollen zugewiesen werden, um auf Projektressourcen und -funktionen zugreifen zu können
- EDITOR-Benutzer können Projektrollen erstellen, um sich selbst und anderen Zugriff auf Projekte zu gewähren, die sie aktualisieren können

Beispiel für einen Arbeitsablauf:

1. Ein ADMIN ordnet der Rolle EDITOR weltweit eine Identity Center-Gruppe zu
2. Ein ADMIN erstellt ein Projekt für ein Team
3. Der EDITOR konfiguriert die Projektrollen innerhalb dieses Projekts, um Teammitgliedern Zugriff auf projektbezogene Ressourcen zu gewähren
4. Teammitglieder (die möglicherweise die globale VIEWER-Rolle haben) können nun Anwendungen in diesem Projekt auf der Grundlage ihrer Projektrollenberechtigungen sehen und verwalten

Einzelheiten zur Konfiguration von Projektrollen finden Sie unterthe section called “Projektbasierte Zugriffskontrolle”.

Rollenzuordnungen konfigurieren

Ordnen Sie AWS Identity Center-Benutzer und -Gruppen bei der Erstellung oder Aktualisierung der Funktion den Argo-CD-Rollen zu.

Beispiel für eine Rollenzuweisung:

```
{
  "rbacRoleMapping": {
    "ADMIN": ["AdminGroup", "alice@example.com"],
    "EDITOR": ["DeveloperGroup", "DevOpsTeam"],
    "VIEWER": ["ReadOnlyGroup", "bob@example.com"]
  }
}
```

> **Note**
>
> Bei Rollennamen wird zwischen Groß- und Kleinschreibung unterschieden und sie müssen in Großbuchstaben geschrieben werden (ADMIN, EDITOR, VIEWER).

> **Important**
>
> Die Integration von EKS Capabilities in AWS Identity Center unterstützt bis zu 1.000 Identitäten pro Argo-CD-Funktion. Eine Identität kann ein Benutzer oder eine Gruppe sein.

Rollenzuordnungen aktualisieren:

```
aws eks update-capability \
  --region us-east-1 \
  --cluster-name cluster \
  --capability-name capname \
  --endpoint "https://eks.ap-northeast-2.amazonaws.com" \
  --role-arn "arn:aws:iam::[.replaceable]111122223333:role/
[.replaceable]`EKSCapabilityRole`" \
  --configuration '{
    "argoCd": {
      "rbacRoleMappings": {
        "addOrUpdateRoleMappings": [
          {
            "role": "ADMIN",
            "identities": [
              { "id": "686103e0-f051-7068-b225-e6392b959d9e", "type": "SSO_USER" }
            ]
          }
```

```
        ]
      }
    }
}'
```

Nutzung des Administratorkontos

Das Administratorkonto ist für die Ersteinrichtung und administrative Aufgaben wie die Registrierung von Clustern und die Konfiguration von Repositorys konzipiert.

Wenn das Administratorkonto geeignet ist:

- Erste Einrichtung und Konfiguration der Funktionen
- Einzelentwicklung oder schnelle Vorführungen
- Administrative Aufgaben (Clusterregistrierung, Repository-Konfiguration, Projekterstellung)

Bewährte Methoden für Administratorkonten:

- Übergeben Sie Konto-Tokens nicht der Versionskontrolle
- Wechseln Sie Tokens sofort, wenn sie offengelegt werden
- Beschränken Sie die Verwendung von Konto-Tokens auf Einrichtungs- und Verwaltungsaufgaben
- Legen Sie kurze Ablaufzeiten fest (maximal 12 Stunden)
- Es können jeweils nur 5 Konto-Tokens erstellt werden

Wann sollte stattdessen der projektbasierte Zugriff verwendet werden:

- Gemeinsame Entwicklungsumgebungen mit mehreren Benutzern
- Jede Umgebung, die der Produktion ähnelt
- Wenn Sie Prüfprotokolle darüber benötigen, wer Aktionen ausgeführt hat
- Wenn Sie Ressourcen- oder Zugriffsbeschränkungen durchsetzen müssen

Verwenden Sie für Produktionsumgebungen und Szenarien mit mehreren Benutzern eine projektbasierte Zugriffskontrolle mit speziellen RBAC-Rollen, die Identity Center-Gruppen zugeordnet sind. AWS

Projektbasierte Zugriffskontrolle

Verwenden Sie Argo CD Projects (AppProject), um Teams eine detaillierte Zugriffskontrolle und Ressourcenisolierung zu bieten.

> **Important**
>
> Bevor Sie Benutzern oder Gruppen projektspezifischen Rollen zuweisen, müssen Sie sie zunächst einer globalen Argo-CD-Rolle (ADMIN, EDITOR oder VIEWER) in der Funktionskonfiguration zuordnen. Benutzer können ohne eine globale Rollenzuweisung nicht auf Argo CD zugreifen, selbst wenn sie Projektrollen zugewiesen sind.
> Erwägen Sie, Benutzer global der VIEWER-Rolle zuzuordnen und dann zusätzliche Berechtigungen über projektspezifische Rollen zu gewähren. Dies bietet Basiszugriff und ermöglicht gleichzeitig eine detaillierte Steuerung auf Projektebene.

Projekte bieten:

- Quellenbeschränkungen: Beschränken Sie, welche Git-Repositorys verwendet werden können
- Zieleinschränkungen: Beschränken Sie, welche Cluster und Namespaces als Ziel ausgewählt werden können
- Ressourceneinschränkungen: Beschränken Sie, welche Kubernetes-Ressourcentypen bereitgestellt werden können
- RBAC-Integration: Ordnen Sie Projekte AWS Identity Center-Gruppen oder Argo-CD-Rollen zu

Beispielprojekt für die Isolierung von Teams:

```
apiVersion: argoproj.io/v1alpha1
kind: AppProject
metadata:
  name: team-a
  namespace: argocd
spec:
  description: Team A applications

  # Required: Specify which namespaces this project watches for Applications
  sourceNamespaces:
  - argocd
```

```
  # Source restrictions
  sourceRepos:
  - https://github.com/myorg/team-a-apps

  # Destination restrictions
  destinations:
  - namespace: team-a-*
    server: arn:aws:eks:us-west-2:111122223333:cluster/production

  # Resource restrictions
  clusterResourceWhitelist:
  - group: ''
    kind: Namespace
  namespaceResourceWhitelist:
  - group: 'apps'
    kind: Deployment
  - group: ''
    kind: Service
  - group: ''
    kind: ConfigMap
```

Quell-Namespaces

Bei Verwendung der EKS Argo CD-Funktion ist das `spec.sourceNamespaces` Feld in Definitionen erforderlich. AppProject Dieses Feld gibt an, welcher Namespace Anwendungen enthalten kann oder ApplicationSets die auf dieses Projekt verweisen.

> **Important**
>
> Die EKS Argo-CD-Funktion unterstützt nur einen einzigen Namespace für Anwendungen und ApplicationSets den Namespace, den Sie bei der Erstellung der Funktion angegeben haben (normalerweise). `argocd` Dies unterscheidet sich von der Open-Source-Version von Argo CD, die mehrere Namespaces unterstützt.

AppProject Konfiguration

Alle AppProjects müssen den konfigurierten Namespace der Fähigkeit enthalten in`sourceNamespaces`:

```
apiVersion: argoproj.io/v1alpha1
```

```
kind: AppProject
metadata:
  name: team-a-project
  namespace: argocd
spec:
  description: Applications for Team A

  # Required: Specify the capability's configured namespace
 (configuration.argoCd.namespace)
  sourceNamespaces:
    - argocd  # Must match your capability's namespace configuration

  # Source repositories this project can deploy from
  sourceRepos:
    - 'https://github.com/my-org/team-a-*'

  # Destination restrictions
  destinations:
    - namespace: 'team-a-*'
      server: arn:aws:eks:us-west-2:111122223333:cluster/my-cluster
```

> **Note**
>
> Wenn Sie den Namespace der Fähigkeit weglassen`sourceNamespaces`, können Anwendungen oder ApplicationSets in diesem Namespace nicht auf dieses Projekt verweisen, was zu Bereitstellungsfehlern führt.

Weisen Sie Benutzer Projekten zu:

Projektrollen gewähren EDITOR- und VIEWER-Benutzern Zugriff auf Projektressourcen (Anwendungen ApplicationSets, Repository- und Cluster-Anmeldeinformationen) und Funktionen (Logs, Exec). Ohne Projektrollen können diese Benutzer nicht auf diese Ressourcen zugreifen, selbst wenn sie globalen Rollenzugriff haben.

ADMIN-Benutzer haben Zugriff auf alle Anwendungen, ohne Projektrollen zu benötigen.

Beispiel: Teammitgliedern Anwendungszugriff gewähren

```
apiVersion: argoproj.io/v1alpha1
kind: AppProject
metadata:
```

```
  name: team-a
  namespace: argocd
spec:
  # ... project configuration ...

  sourceNamespaces:
  - argocd

  # Project roles grant Application-level access
  roles:
  - name: developer
    description: Team A developers - can manage Applications
    policies:
    - p, proj:team-a:developer, applications, *, team-a/*, allow
    groups:
    - 686103e0-f051-7068-b225-e6392b959d9e  # Identity Center group ID

  - name: viewer
    description: Team A viewers - read-only Application access
    policies:
    - p, proj:team-a:viewer, applications, get, team-a/*, allow
    groups:
    - 786203e0-f051-7068-b225-e6392b959d9f  # Identity Center group ID
```

Grundlegendes zu den Rollenrichtlinien von Projekten:

Das Richtlinienformat lautet: `p, proj:<project>:<role>, <resource>, <action>, <object>, <allow/deny>`

Ressourcenrichtlinien:

- `applications, , team-a/, allow`- Voller Zugriff auf alle Anwendungen im Team-A-Projekt
- `applications, get, team-a/*, allow`- Nur-Lese-Zugriff auf Anwendungen
- `applications, sync, team-a/*, allow`- Kann Anwendungen synchronisieren, aber nicht erstellen/löschen
- `applications, delete, team-a/*, allow`- Kann Anwendungen löschen (mit Vorsicht verwenden)
- `applicationsets, , team-a/, allow`- Voller Zugriff auf ApplicationSets
- `repositories, *, *, allow`- Zugriff auf Repository-Anmeldeinformationen
- `clusters, *, *, allow`- Zugriff auf Cluster-Anmeldeinformationen

Fähigkeitsrichtlinien:

- `logs, , team-a/, allow`- Zugriff auf Anwendungsprotokolle
- `exec, , team-a/, allow`- Zugriff für Führungskräfte auf Anwendungs-Pods

> **Note**
>
> EDITOR-Benutzer können Projektrollen erstellen, um sich selbst und anderen Berechtigungen innerhalb von Projekten zu gewähren, die sie aktualisieren können. Auf diese Weise können Teamleiter den Zugriff auf projektbezogene Ressourcen für ihr Team kontrollieren, ohne dass ein Eingreifen durch den Administrator erforderlich ist.

> **Note**
>
> Verwenden Sie die Identity Center-Gruppe IDs (keine Gruppennamen) im Feld. `groups` Sie können den Identity Center-Benutzer auch IDs für den individuellen Benutzerzugriff verwenden. Sie finden diese IDs in der AWS Identity Center-Konsole oder mithilfe der AWS CLI.

Allgemeine Berechtigungsmuster

Muster 1: Admin-Team mit vollem Zugriff

```
{
  "rbacRoleMapping": {
    "ADMIN": ["PlatformTeam", "SRETeam"]
  }
}
```

ADMIN-Benutzer können alle projektbezogenen Ressourcen ohne zusätzliche Konfiguration einsehen und verwalten.

Muster 2: Teamleiter verwalten Projekte, Entwickler greifen über Projektrollen darauf zu

```
{
  "rbacRoleMapping": {
```

```
    "ADMIN": ["PlatformTeam"],
    "EDITOR": ["TeamLeads"],
    "VIEWER": ["AllDevelopers"]
  }
}
```

1. ADMIN erstellt Projekte für jedes Team
2. Teamleiter (EDITOR) konfigurieren Projektrollen, um ihren Entwicklern Zugriff auf Projektressourcen (Anwendungen ApplicationSets, Anmeldeinformationen) und Funktionen (Logs, Exec) zu gewähren
3. Entwickler (VIEWER) können nur auf Ressourcen und Funktionen zugreifen, die ihren Projektrollen entsprechen

Muster 3: Teambasierter Zugriff mit Projektrollen

1. ADMIN erstellt Projekte und ordnet Teamleiter der Rolle EDITOR weltweit zu
2. Teamleiter (EDITOR) weisen Teammitgliedern Projektrollen innerhalb ihrer Projekte zu
3. Teammitglieder benötigen nur die globale VIEWER-Rolle — Projektrollen bieten Zugriff auf Projektressourcen und -funktionen

```
{
  "rbacRoleMapping": {
    "ADMIN": ["PlatformTeam"],
    "EDITOR": ["TeamLeads"],
    "VIEWER": ["AllDevelopers"]
  }
}
```

Best Practices

Verwenden Sie Gruppen statt einzelner Benutzer: Ordnen Sie AWS Identity Center-Gruppen Argo-CD-Rollen statt einzelnen Benutzern zu, um die Verwaltung zu vereinfachen.

Beginnen Sie mit den geringsten Rechten: Beginnen Sie mit VIEWER-Zugriff und gewähren Sie je nach Bedarf EDITOR oder ADMIN.

Verwenden Sie Projekte zur Teamisolierung: Erstellen Sie separate Projekte AppProjects für verschiedene Teams oder Umgebungen, um Grenzen durchzusetzen.

Nutzen Sie den Identity Center-Verbund: Konfigurieren Sie AWS Identity Center so, dass es für eine zentrale Benutzerverwaltung eine Verbindung mit Ihrem Corporate Identity Provider herstellt.

Regelmäßige Zugriffsprüfungen: Überprüfen Sie regelmäßig die Rollenzuordnungen und Projektzuweisungen, um sicherzustellen, dass die Zugriffsebenen angemessen sind.

Clusterzugriff einschränken: Denken Sie daran, dass Argo CD RBAC den Zugriff auf Argo CD-Ressourcen und -Operationen kontrolliert, aber nicht Kubernetes RBAC entspricht. Benutzer mit Argo-CD-Zugriff können Anwendungen auf Clustern bereitstellen, auf die Argo CD Zugriff hat. Beschränken Sie, auf welche Cluster Argo CD zugreifen kann, und kontrollieren Sie anhand von Beschränkungen für Projektziele, wo Anwendungen bereitgestellt werden können.

AWS Serviceberechtigungen

Um AWS Dienste direkt in Anwendungsressourcen zu verwenden (ohne Repository-Ressourcen zu erstellen), fügen Sie der Capability Role die erforderlichen IAM-Berechtigungen hinzu.

ECR für Helm-Diagramme:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecr:GetAuthorizationToken",
        "ecr:BatchCheckLayerAvailability",
        "ecr:GetDownloadUrlForLayer",
        "ecr:BatchGetImage"
      ],
      "Resource": "*"
    }
  ]
}
```

CodeCommit Repositorien:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": [
        "codecommit:GitPull"
      ],
      "Resource": "arn:aws:codecommit:region:account-id:repository-name"
    }
  ]
}
```

CodeConnections (GitHub, GitLab, Bitbucket):

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codeconnections:UseConnection"
      ],
      "Resource": "arn:aws:codeconnections:region:account-id:connection/connection-id"
    }
  ]
}
```

Einzelheiten the section called “Repositorys konfigurieren” zur Verwendung dieser Integrationen finden Sie unter.

Nächste Schritte

- the section called “Arbeiten mit Argo CD”- Erfahren Sie, wie Sie Anwendungen erstellen und Bereitstellungen verwalten
- the section called “Argo-CD-Konzepte”- Verstehen Sie die Konzepte von Argo CD, einschließlich Projekte
- the section called “Überlegungen zu den Funktionen von EKS”- Informieren Sie sich über bewährte Sicherheitsverfahren im Hinblick auf Funktionen

Arbeiten mit Argo CD

Mit Argo CD definieren Sie Anwendungen in Git-Repositorys und Argo CD synchronisiert sie automatisch mit Ihren Kubernetes-Clustern. Dies ermöglicht eine deklarative, versionskontrollierte Anwendungsbereitstellung mit automatisierter Drift-Erkennung.

Voraussetzungen

Bevor Sie mit Argo CD arbeiten können, benötigen Sie:

- Es wurde ein EKS-Cluster mit der Argo-CD-Funktion erstellt (siehe) the section called “Erstellen der Argo CD-Funktion”
- Ein Git-Repository mit Kubernetes-Manifesten
- `kubectl`konfiguriert für die Kommunikation mit Ihrem Cluster

Allgemeine Aufgaben

Die folgenden Themen führen Sie durch die häufigsten Aufgaben von Argo CD:

the section called “Repositorys konfigurieren”- Konfigurieren Sie Argo CD für den Zugriff auf Ihre Git-Repositorys mit AWS Secrets Manager oder AWS CodeConnections Kubernetes Secrets.

the section called “Registrieren Sie Cluster”- Registrieren Sie Zielcluster, auf denen Argo CD Anwendungen bereitstellen wird.

the section called “Projekte”- Organisieren Sie Anwendungen und setzen Sie Sicherheitsgrenzen durch, indem Sie Projekte für Umgebungen mit mehreren Mandanten einsetzen.

the section called “Anwendungen erstellen”- Erstellen Sie Anwendungen, die über Git-Repositorys mit automatisierten oder manuellen Synchronisierungsrichtlinien bereitgestellt werden.

the section called “ApplicationSets”- Wird verwendet ApplicationSets , um Anwendungen mithilfe von Vorlagen und Generatoren in mehreren Umgebungen oder Clustern bereitzustellen.

Greifen Sie auf die Benutzeroberfläche von Argo CD zu

Greifen Sie über die EKS-Konsole auf die Benutzeroberfläche von Argo CD zu:

1. Öffnen Sie die Amazon EKS-Konsole
2. Wählen Sie Ihren Cluster aus
3. Wählen Sie die Registerkarte Funktionen
4. Wählen Sie Argo CD
5. Wählen Sie Open Argo CD UI

Die Benutzeroberfläche bietet visuelle Anwendungstopologie, Synchronisierungsstatus und -verlauf, Ressourcenstatus und Ereignisse, manuelle Synchronisierungssteuerungen und Anwendungsverwaltung.

Upstream-Dokumentation

Ausführliche Informationen zu den Funktionen von Argo CD finden Sie unter:

- Argo CD-Dokumentation — Vollständiges Benutzerhandbuch
- Anwendungsspezifikation — Vollständige Anwendungs-API-Referenz
- ApplicationSet Leitfaden — ApplicationSet Muster und Beispiele
- Argo CD GitHub - Quellcode und Beispiele

Repository-Zugriff konfigurieren

Bevor Sie Anwendungen bereitstellen, konfigurieren Sie Argo CD für den Zugriff auf Ihre Git-Repositorys und Helm-Chart-Registries. Argo CD unterstützt mehrere Authentifizierungsmethoden für GitHub, GitLab, Bitbucket und ECR. AWS CodeCommit AWS

> **Note**
>
> Bei direkten AWS Serviceintegrationen (ECR-Helm-Diagramme, CodeCommit Repositorys und CodeConnections) können Sie direkt in den Anwendungsressourcen darauf verweisen, ohne Repository-Konfigurationen erstellen zu müssen. Die Capability-Rolle muss über die erforderlichen IAM-Berechtigungen verfügen. Details dazu finden Sie unter the section called “Konfigurieren von -Berechtigungen”.

Voraussetzungen

- Es wurde ein EKS-Cluster mit der Argo-CD-Funktion erstellt
- Git-Repositorys, die Kubernetes-Manifeste enthalten
- `kubectl`konfiguriert für die Kommunikation mit Ihrem Cluster

> **Note**
>
> AWS CodeConnections kann eine Verbindung zu Git-Servern herstellen, die sich in der AWS Cloud oder vor Ort befinden. Weitere Informationen finden Sie unter AWS CodeConnections.

Authentifizierungsmethoden

Methode	Anwendungsfall	Erforderliche IAM-Berechtigungen
Direkte Integration mit Diensten AWS		
CodeCommit	Direkte Integration mit AWS CodeCommit Git-Repositorys. Keine Repository-Konfiguration erforderlich.	`codecommit:GitPull`
CodeConnections	Stellen Sie mit verwalteter Authentifizierung eine Connect zu GitHub GitLab, oder Bitbucket her. Erfordert eine Verbindungseinrichtung.	`codeconnections:UseConnection`
ECR OCI-Artefakte	Direkte Integration mit AWS ECR für OCI Helm-Diagramme und Manifestbilder. Keine Repository-Konfiguration erforderlich.	`arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryPullOnly`
Repository-Konfiguration mit Anmeldeinformationen		
AWS Secrets Manager (Benutzername/Token)	Speichern Sie persönliche Zugriffstoken oder Passwörter. Ermöglicht die Rotation von Anmeldeinformationen ohne Kubernetes-Zugriff.	`arn:aws:iam::aws:policy/AWSSecretsManagerClientReadOnlyAccess`
AWS Secrets Manager (SSH-Schlüssel)	Verwenden Sie die SSH-Schlüsselauthentifizierung. Ermöglicht die Rotation von Anmeldeinformationen ohne Kubernetes-Zugriff.	`arn:aws:iam::aws:policy/AWSSecretsManagerClientReadOnlyAccess`

Methode	Anwendungsfall	Erforderliche IAM-Berechtigungen
AWS Secrets Manager (GitHub App)	GitHub App-Authentifizierung mit privatem Schlüssel. Ermöglicht die Rotation von Anmeldeinformationen ohne Kubernetes-Zugriff.	`arn:aws:iam::aws:policy/AWSSecretsManagerClientReadOnlyAccess`
Kubernetes-Geheimnis	Standard-Argo-CD-Methode mit Cluster-internen Geheimnissen	Keine (Berechtigungen werden von EKS Access Entry mit Kubernetes RBAC verwaltet)

Direkter Zugriff auf Dienste AWS

AWS Dienste können Sie direkt in den Anwendungsressourcen referenzieren, ohne Repository-Konfigurationen erstellen zu müssen. Die Capability-Rolle muss über die erforderlichen IAM-Berechtigungen verfügen.

CodeCommit Repositorien

CodeCommit Referenz-Repositorien direkt in Anwendungen:

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: my-app
  namespace: argocd
spec:
  source:
    repoURL: https://git-codecommit.region.amazonaws.com/v1/repos/repository-name
    targetRevision: main
    path: kubernetes/manifests
```

Erforderliche Berechtigungen für Capability Role:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codecommit:GitPull",
```

```
      "Resource": "arn:aws:codecommit:region:account-id:repository-name"
    }
  ]
}
```

CodeConnections

Referenz GitHub - GitLab oder Bitbucket-Repositorys durch. CodeConnections Das Repository-URL-Format wird vom CodeConnections Verbindungs-ARN abgeleitet.

Das Repository-URL-Format ist:

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: my-app
  namespace: argocd
spec:
  source:
    repoURL: https://codeconnections.region.amazonaws.com/git-http/account-
id/region/connection-id/owner/repository.git
    targetRevision: main
    path: kubernetes/manifests
```

Erforderliche Berechtigungen für Capability Role:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codeconnections:UseConnection",
      "Resource": "arn:aws:codeconnections:region:account-id:connection/connection-id"
    }
  ]
}
```

ECR-Helm-Diagramme

ECR speichert Helm-Diagramme als OCI-Artefakte. Argo CD unterstützt zwei Möglichkeiten, auf sie zu verweisen:

Helm-Format (empfohlen für Helm-Charts):

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: my-app-helm
  namespace: argocd
spec:
  source:
    repoURL: account-id.dkr.ecr.region.amazonaws.com/repository-name
    targetRevision: chart-version
    chart: chart-name
    helm:
      valueFiles:
        - values.yaml
```

Hinweis: Geben Sie das `oci://` Präfix nicht an, wenn Sie das Helm-Format verwenden. Verwenden Sie das `chart` Feld, um den Namen des Diagramms anzugeben.

OCI-Format (für OCI-Artefakte mit Kubernetes-Manifesten):

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: my-app-oci
  namespace: argocd
spec:
  source:
    repoURL: oci://account-id.dkr.ecr.region.amazonaws.com/repository-name
    targetRevision: artifact-version
    path: path-to-manifests
```

Hinweis: Geben Sie das `oci://` Präfix an, wenn Sie das OCI-Format verwenden. Verwenden Sie das `path` Feld anstelle von. `chart`

Erforderliche Berechtigungen für Capability Role — fügen Sie die verwaltete Richtlinie an:

```
arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryPullOnly
```

Diese Richtlinie umfasst die erforderlichen ECR-Berechtigungen: `ecr:GetAuthorizationTokenecr:BatchGetImage`, und`ecr:GetDownloadUrlForLayer`.

AWS Secrets Manager verwenden

Speichern Sie die Repository-Anmeldeinformationen in Secrets Manager und referenzieren Sie sie in den Argo CD Repository-Konfigurationen. Die Verwendung von Secrets Manager ermöglicht die automatische Rotation von Anmeldeinformationen, ohne dass Kubernetes RBAC-Zugriff erforderlich ist. Anmeldeinformationen können mithilfe von IAM-Berechtigungen an Secrets Manager rotiert werden, und Argo CD liest die aktualisierten Werte automatisch.

> **Note**
>
> Für die Wiederverwendung von Anmeldeinformationen in mehreren Repositorys (z. B. für alle Repositorys innerhalb einer Organisation) verwenden Sie Vorlagen für Repository-Anmeldeinformationen mit. GitHub `argocd.argoproj.io/secret-type: repo-creds` Dies bietet eine bessere Benutzererfahrung als die Erstellung einzelner Repository-Geheimnisse. Weitere Informationen finden Sie unter Repository-Anmeldeinformationen in der Argo-CD-Dokumentation.

Authentifizierung mit Benutzername und Token

Für HTTPS-Repositorys mit persönlichen Zugriffstoken oder Passwörtern:

Erstellen Sie das Geheimnis in Secrets Manager:

```
aws secretsmanager create-secret \
  --name argocd/my-repo \
  --description "GitHub credentials for Argo CD" \
  --secret-string '{"username":"your-username","token":"your-personal-access-token"}'
```

Optionale Felder für TLS-Client-Zertifikate (für private Git-Server):

```
aws secretsmanager create-secret \
  --name argocd/my-private-repo \
  --secret-string '{
    "username":"your-username",
    "token":"your-token",

 "tlsClientCertData":"LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCi4uLgotLS0tLUVORCBDRVJUSUZJQ0FURS0tL

 "tlsClientCertKey":"LS0tLS1CRUdJTiBQUklWQVRFIEtFWS0tLS0tCi4uLgotLS0tLUVORCBQUklWQVRFIEtFWS0tLS
```

```
  }'
```

> **Note**
>
> Die `tlsClientCertKey` Werte `tlsClientCertData` und müssen base64-codiert sein.

Erstellen Sie ein Repository-Geheimnis, das auf Secrets Manager verweist:

```
apiVersion: v1
kind: Secret
metadata:
  name: my-repo
  namespace: argocd
  labels:
    argocd.argoproj.io/secret-type: repository
stringData:
  type: git
  url: https://github.com/your-org/your-repo
  secretArn: arn:aws:secretsmanager:us-west-2:111122223333:secret:argocd/my-repo-AbCdEf
  project: default
```

SSH-Schlüsselauthentifizierung

Für SSH-basierten Git-Zugriff speichern Sie den privaten Schlüssel als Klartext (nicht JSON):

Erstelle das Geheimnis mit dem privaten SSH-Schlüssel:

```
aws secretsmanager create-secret \
  --name argocd/my-repo-ssh \
  --description "SSH key for Argo CD" \
  --secret-string "-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
...
-----END OPENSSH PRIVATE KEY-----"
```

Erstellen Sie ein Repository-Geheimnis für SSH:

```
apiVersion: v1
kind: Secret
metadata:
  name: my-repo-ssh
```

```
  namespace: argocd
  labels:
    argocd.argoproj.io/secret-type: repository
stringData:
  type: git
  url: git@github.com:your-org/your-repo.git
  secretArn: arn:aws:secretsmanager:us-west-2:111122223333:secret:argocd/my-repo-ssh-
AbCdEf
  project: default
```

GitHub App-Authentifizierung

Für die GitHub App-Authentifizierung mit einem privaten Schlüssel:

Erstellen Sie das Geheimnis mit den Anmeldeinformationen für die GitHub App:

```
aws secretsmanager create-secret \
  --name argocd/github-app \
  --description "GitHub App credentials for Argo CD" \
  --secret-string '{

 "githubAppPrivateKeySecret":"LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQouLi4KLS0tLS1FTkQgUlNBI
    "githubAppID":"123456",
    "githubAppInstallationID":"12345678"
  }'
```

> **Note**
>
> Der `githubAppPrivateKeySecret` Wert muss Base64-codiert sein.

Optionales Feld für Enterprise GitHub :

```
aws secretsmanager create-secret \
  --name argocd/github-enterprise-app \
  --secret-string '{

 "githubAppPrivateKeySecret":"LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQouLi4KLS0tLS1FTkQgUlNBI
    "githubAppID":"123456",
    "githubAppInstallationID":"12345678",
    "githubAppEnterpriseBaseUrl":"https://github.example.com/api/v3"
  }'
```

Erstellen Sie einen geheimen Repository-Schlüssel für die GitHub App:

```
apiVersion: v1
kind: Secret
metadata:
  name: my-repo-github-app
  namespace: argocd
  labels:
    argocd.argoproj.io/secret-type: repository
stringData:
  type: git
  url: https://github.com/your-org/your-repo
  secretArn: arn:aws:secretsmanager:us-west-2:111122223333:secret:argocd/github-app-
AbCdEf
  project: default
```

Vorlagen für Repository-Anmeldeinformationen

Für die Wiederverwendung von Anmeldeinformationen in mehreren Repositorys (z. B. für alle Repositorys einer GitHub Organisation oder eines Benutzers) verwenden Sie Vorlagen für Repository-Anmeldeinformationen mit. `argocd.argoproj.io/secret-type: repo-creds` Dies bietet eine bessere Benutzererfahrung als die Erstellung von individuellen Repository-Geheimnissen für jedes Repository.

Erstellen Sie eine Vorlage für Repository-Anmeldeinformationen:

```
apiVersion: v1
kind: Secret
metadata:
  name: github-org-creds
  namespace: argocd
  labels:
    argocd.argoproj.io/secret-type: repo-creds
stringData:
  type: git
  url: https://github.com/your-org
  secretArn: arn:aws:secretsmanager:us-west-2:111122223333:secret:argocd/github-org-
AbCdEf
```

Diese Vorlage für Anmeldeinformationen gilt für alle Repositorys, die dem URL-Präfix entsprechen. `https://github.com/your-org` Sie können dann in Anwendungen auf jedes Repository dieser Organisation verweisen, ohne zusätzliche Geheimnisse zu erstellen.

Weitere Informationen finden Sie unter Repository-Anmeldeinformationen in der Argo-CD-Dokumentation.

> **Important**
>
> Stellen Sie sicher, dass Ihrer IAM-Capability-Rolle die verwaltete Richtlinie oder gleichwertige Berechtigungen, einschließlich `secretsmanager:GetSecretValue` KMS-Entschlüsselungsberechtigungen, `arn:aws:iam::aws:policy/AWSSecretsManagerClientReadOnlyAccess` zugewiesen sind. Informationen the section called “Überlegungen zu Argo CD” zur Konfiguration der IAM-Richtlinie finden Sie unter.

Verwenden AWS CodeConnections

Informationen zur CodeConnections Integration finden Sie unterthe section called “ AWS CodeConnections”.

CodeConnections bietet verwaltete Authentifizierung für GitHub GitLab, und Bitbucket, ohne Anmeldeinformationen zu speichern.

Verwendung von Kubernetes Secrets

Speichern Sie Anmeldeinformationen mithilfe der Standard-Argo-CD-Methode direkt in Kubernetes.

Für HTTPS mit persönlichem Zugriffstoken:

```
apiVersion: v1
kind: Secret
metadata:
  name: my-repo
  namespace: argocd
  labels:
    argocd.argoproj.io/secret-type: repository
stringData:
  type: git
  url: https://github.com/your-org/your-repo
  username: your-username
  password: your-personal-access-token
```

Für SSH:

```
apiVersion: v1
kind: Secret
metadata:
  name: my-repo-ssh
  namespace: argocd
  labels:
    argocd.argoproj.io/secret-type: repository
stringData:
  type: git
  url: git@github.com:your-org/your-repo.git
  sshPrivateKey: |
    -----BEGIN OPENSSH PRIVATE KEY-----
    ... your private key ...
    -----END OPENSSH PRIVATE KEY-----
```

CodeCommit Repositorien

Erteilen Sie für AWS CodeCommit Ihre CodeCommit IAM-Capability-Rollenberechtigungen ()`codecommit:GitPull`.

Konfigurieren Sie das Repository:

```
apiVersion: v1
kind: Secret
metadata:
  name: codecommit-repo
  namespace: argocd
  labels:
    argocd.argoproj.io/secret-type: repository
stringData:
  type: git
  url: https://git-codecommit.us-west-2.amazonaws.com/v1/repos/my-repo
  project: default
```

Eine detaillierte Konfiguration der IAM-Richtlinien finden Sie unterthe section called “Überlegungen zu Argo CD”.

Überprüfen Sie die Verbindung zum Repository

Überprüfen Sie den Verbindungsstatus über die Benutzeroberfläche von Argo CD unter Einstellungen → Repositories. Die Benutzeroberfläche zeigt den Verbindungsstatus und alle Authentifizierungsfehler an.

Repository Secrets enthalten keine Statusinformationen.

Weitere Ressourcen

- the section called “Registrieren Sie Cluster”— Registrieren Sie Zielcluster für Bereitstellungen
- the section called “Anwendungen erstellen”- Erstellen Sie Ihre erste Anwendung
- the section called “Überlegungen zu Argo CD”- IAM-Berechtigungen und Sicherheitskonfiguration
- Private Repositorys — Referenz zur Konfiguration des Upstream-Repositorys

Zielcluster registrieren

Registrieren Sie Cluster, damit Argo CD Anwendungen für sie bereitstellen kann. Sie können denselben Cluster registrieren, auf dem Argo CD läuft (lokaler Cluster), oder Remote-Cluster in verschiedenen Konten oder Regionen.

Voraussetzungen

- Es wurde ein EKS-Cluster mit der Argo-CD-Funktion erstellt
- `kubectl`konfiguriert für die Kommunikation mit Ihrem Cluster
- Für Remote-Cluster: entsprechende IAM-Berechtigungen und Zugriffseinträge

Registrieren Sie den lokalen Cluster

Um Anwendungen auf demselben Cluster bereitzustellen, auf dem Argo CD läuft, registrieren Sie ihn als Bereitstellungsziel.

Important

Die Argo-CD-Funktion registriert den lokalen Cluster nicht automatisch. Sie müssen es explizit registrieren, um Anwendungen auf demselben Cluster bereitzustellen. Sie können den Clusternamen aus `in-cluster` Gründen der Kompatibilität mit den meisten Argo-CD-Beispielen im Internet verwenden.

Note

Ein EKS-Zugriffseintrag wird automatisch für den lokalen Cluster mit der Argo-CD-Capability-Rolle erstellt, aber standardmäßig werden keine Kubernetes-RBAC-Berechtigungen

gewährt. Dies folgt dem Prinzip der geringsten Rechte — Sie müssen die Berechtigungen, die Argo CD benötigt, explizit auf der Grundlage Ihres Anwendungsfalls konfigurieren. Wenn Sie diesen Cluster beispielsweise nur als Argo-CD-Hub für die Verwaltung von Remote-Clustern verwenden, benötigt er keine lokalen Bereitstellungsberechtigungen. Die Konfigurationsoptionen finden Sie im Abschnitt mit den RBAC-Anforderungen für Access Entry weiter unten.

Verwenden der Argo-CD-CLI:

```
argocd cluster add <cluster-context-name> \
  --aws-cluster-name arn:aws:eks:us-west-2:111122223333:cluster/my-cluster \
  --name local-cluster
```

Verwendung eines Kubernetes-Geheimnisses:

```
apiVersion: v1
kind: Secret
metadata:
  name: local-cluster
  namespace: argocd
  labels:
    argocd.argoproj.io/secret-type: cluster
stringData:
  name: local-cluster
  server: arn:aws:eks:us-west-2:111122223333:cluster/my-cluster
  project: default
```

Wenden Sie die Konfiguration an:

```
kubectl apply -f local-cluster.yaml
```

Note

Verwenden Sie den EKS-Cluster-ARN im `server` Feld, nicht die Kubernetes-API-Server-URL. Die verwaltete Funktion erfordert die Identifizierung ARNs von Clustern. Die Standardeinstellung `kubernetes.default.svc` wird nicht unterstützt.

Registrieren Sie Remote-Cluster

So führen Sie die Bereitstellung auf Remote-Clustern durch:

Schritt 1: Erstellen Sie den Zugriffseintrag auf dem Remote-Cluster

*region-code*Ersetzen Sie es durch die AWS Region, in der sich Ihr Remote-Cluster befindet, *remote-cluster* ersetzen Sie es durch den Namen Ihres Remote-Clusters und ersetzen Sie den ARN durch Ihren ARGO-CD-Funktionsrollen-ARN.

```
aws eks create-access-entry \
  --region region-code \
  --cluster-name remote-cluster \
  --principal-arn arn:aws:iam::[.replaceable]111122223333:role/ArgoCDCapabilityRole \
  --type STANDARD
```

Schritt 2: Verknüpfen Sie eine Zugriffsrichtlinie mit Kubernetes-RBAC-Berechtigungen

Für den Access Entry sind Kubernetes-RBAC-Berechtigungen erforderlich, damit Argo CD Anwendungen bereitstellen kann. Für einen schnellen Einstieg können Sie Folgendes verwenden: `AmazonEKSClusterAdminPolicy`

```
aws eks associate-access-policy \
  --region region-code \
  --cluster-name remote-cluster \
  --principal-arn arn:aws:iam::[.replaceable]111122223333:role/ArgoCDCapabilityRole \
  --policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy \
  --access-scope type=cluster
```

Important

Das `AmazonEKSClusterAdminPolicy` bietet vollen Cluster-Admin-Zugriff (entsprichtsystem:masters). Dies ist praktisch für den Einstieg, sollte aber nicht in der Produktion verwendet werden. Verwenden Sie für Produktionsumgebungen restriktivere Berechtigungen, indem Sie den Access Entry benutzerdefinierten Kubernetes-Gruppen zuordnen und entsprechende Rollen oder Bindungen erstellen. ClusterRole Informationen zur Konfiguration mit den geringsten Rechten finden Sie im Abschnitt zur Produktionseinrichtung weiter unten.

Schritt 3: Registrieren Sie den Cluster auf Argo CD

Verwenden der Argo-CD-CLI:

```
argocd cluster add <cluster-context-name> \
  --aws-cluster-name arn:aws:eks:us-west-2:111122223333:cluster/remote-cluster \
  --name remote-cluster
```

Verwendung eines Kubernetes-Geheimnisses:

```
apiVersion: v1
kind: Secret
metadata:
  name: remote-cluster
  namespace: argocd
  labels:
    argocd.argoproj.io/secret-type: cluster
stringData:
  name: remote-cluster
  server: arn:aws:eks:us-west-2:111122223333:cluster/remote-cluster
  project: default
```

Wenden Sie die Konfiguration an:

```
kubectl apply -f remote-cluster.yaml
```

Kontenübergreifende Cluster

So führen Sie die Bereitstellung auf Clustern mit unterschiedlichen AWS Konten durch:

1. Erstellen Sie im Zielkonto einen Access-Eintrag auf dem EKS-Ziel-Cluster, indem Sie den Argo CD IAM Capability Role ARN aus dem Quellkonto als Principal verwenden
2. Ordnen Sie eine Zugriffsrichtlinie den entsprechenden Kubernetes-RBAC-Berechtigungen zu
3. Registrieren Sie den Cluster in Argo CD mit seinem EKS-Cluster-ARN

Es ist keine zusätzliche IAM-Rollenerstellung oder Konfiguration von Vertrauensrichtlinien erforderlich — EKS Access Entries kümmern sich um den kontoübergreifenden Zugriff.

Das Cluster-ARN-Format umfasst die Region, sodass regionsübergreifende Bereitstellungen denselben Prozess verwenden wie Bereitstellungen derselben Region.

Überprüfen Sie die Clusterregistrierung

Registrierte Cluster anzeigen:

```
kubectl get secrets -n argocd -l argocd.argoproj.io/secret-type=cluster
```

Oder überprüfen Sie den Clusterstatus in der Benutzeroberfläche von Argo CD unter Einstellungen → Cluster.

Private Cluster

Die Argo-CD-Funktion bietet transparenten Zugriff auf vollständig private EKS-Cluster, ohne dass VPC-Peering oder eine spezielle Netzwerkkonfiguration erforderlich sind.

AWS verwaltet automatisch die Konnektivität zwischen der Argo-CD-Funktion und privaten Remote-Clustern.

Registrieren Sie den privaten Cluster einfach mit seinem ARN — es ist keine zusätzliche Netzwerkeinrichtung erforderlich.

Access Entry RBAC-Anforderungen

Wenn Sie eine Argo-CD-Funktion erstellen, wird automatisch ein EKS-Zugriffseintrag für die Capability Role erstellt, aber standardmäßig werden keine Kubernetes-RBAC-Berechtigungen gewährt. Dieses absichtliche Design folgt dem Prinzip der geringsten Rechte — unterschiedliche Anwendungsfälle erfordern unterschiedliche Berechtigungen.

Zum Beispiel: * Wenn Sie den Cluster nur als Argo-CD-Hub zur Verwaltung von Remote-Clustern verwenden, benötigt er keine lokalen Bereitstellungsberechtigungen. * Wenn Sie Anwendungen lokal bereitstellen, benötigt er Lesezugriff für den gesamten Cluster und Schreibzugriff auf bestimmte Namespaces. * Wenn Sie Anwendungen erstellen müssen, sind zusätzliche Cluster-Admin-Rechte erforderlich CRDs

Sie müssen die Berechtigungen, die Argo CD benötigt, explizit auf der Grundlage Ihrer Anforderungen konfigurieren.

Mindestberechtigungen für Argo CD

Argo CD benötigt zwei Arten von Berechtigungen, um fehlerfrei zu funktionieren:

Leseberechtigungen (clusterweit): Argo CD muss in der Lage sein, alle Ressourcentypen und benutzerdefinierten Ressourcendefinitionen (CRDs) im gesamten Cluster zu lesen für:

- Erkennung von Ressourcen und Zustandsprüfungen
- Erkennung von Abweichungen zwischen dem gewünschten und dem tatsächlichen Zustand
- Validierung der Ressourcen vor der Bereitstellung

Schreibberechtigungen (namespace-spezifisch): Argo CD benötigt Erstellungs-, Aktualisierungs- und Löschberechtigungen für Ressourcen, die in Anwendungen definiert sind:

- Stellen Sie Anwendungs-Workloads bereit (Bereitstellungen, Dienste usw.) ConfigMaps
- Wenden Sie benutzerdefinierte Ressourcen an (CRDs spezifisch für Ihre Anwendungen)
- Verwalten Sie den Anwendungslebenszyklus

Quick Setup

Verwenden Sie für einen schnellen Einstieg in Test- oder EntwicklungsumgebungenAmazonEKSClusterAdminPolicy:

```
aws eks associate-access-policy \
  --region region-code \
  --cluster-name my-cluster \
  --principal-arn arn:aws:iam::[.replaceable]111122223333:role/ArgoCDCapabilityRole \
  --policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy \
  --access-scope type=cluster
```

Important

Das AmazonEKSClusterAdminPolicy bietet vollen Cluster-Admin-Zugriff (entsprichtsystem:masters), einschließlich der Möglichkeit, clusterweite Ressourcen zu erstellen CRDs, zu ändern und in jedem Namespace bereitzustellen. Dies ist praktisch für die Entwicklung, sollte POCs aber nicht in der Produktion verwendet werden. Verwenden Sie für die Produktion das unten stehende Setup mit den geringsten Rechten.

Produktions-Setup mit den geringsten Rechten

Erstellen Sie für Produktionsumgebungen eine benutzerdefinierte Kubernetes-RBAC, die Folgendes gewährt:

- Clusterweiter Lesezugriff auf alle Ressourcen (für Erkennungs- und Integritätsprüfungen)

- Namespace-spezifischer Schreibzugriff (für Bereitstellungen)

Schritt 1: Verknüpfen Sie Access Entry mit einer benutzerdefinierten Kubernetes-Gruppe

```
aws eks associate-access-policy \
  --region region-code \
  --cluster-name my-cluster \
  --principal-arn arn:aws:iam::[.replaceable]111122223333:role/ArgoCDCapabilityRole \
  --policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSEditPolicy \
  --access-scope type=namespace,namespaces=app-namespace
```

Schritt 2: ClusterRole Für Lesezugriff erstellen

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: argocd-read-all
rules:
# Read access to all resources for discovery and health checks
- apiGroups: ["*"]
  resources: ["*"]
  verbs: ["get", "list", "watch"]
```

Schritt 3: Rolle für den Schreibzugriff auf Anwendungs-Namespaces erstellen

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: argocd-deploy
  namespace: app-namespace
rules:
# Full access to deploy application resources
- apiGroups: ["*"]
  resources: ["*"]
  verbs: ["*"]
```

Schritt 4: Binden Sie Rollen an die Kubernetes-Gruppe

```
apiVersion: rbac.authorization.k8s.io/v1
```

```
kind: ClusterRoleBinding
metadata:
  name: argocd-read-all
subjects:
- kind: Group
  name: eks-access-entry:arn:aws:iam::111122223333:role/ArgoCDCapabilityRole
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: argocd-read-all
  apiGroup: rbac.authorization.k8s.io
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: argocd-deploy
  namespace: app-namespace
subjects:
- kind: Group
  name: eks-access-entry:arn:aws:iam::111122223333:role/ArgoCDCapabilityRole
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: argocd-deploy
  apiGroup: rbac.authorization.k8s.io
```

Note

Auf das Gruppennamenformat für Access Entries `eks-access-entry:` folgt der Prinzipal-ARN. Wiederholen Sie den Vorgang RoleBinding für jeden Namespace, in dem Argo CD Anwendungen bereitstellen soll.

Important

Argo CD muss in der Lage sein, alle Ressourcentypen im gesamten Cluster für Integritätsprüfungen und Erkennungen zu lesen, auch wenn es nur in bestimmten Namespaces bereitgestellt wird. Ohne clusterweiten Lesezugriff zeigt Argo CD bei der Überprüfung des Anwendungszustands Fehler an.

Beschränken Sie den Clusterzugriff mit Projekten

> **Important**
>
> In der verwalteten Funktion kann jeder Zielcluster nur einem Argo-CD-Projekt zugeordnet werden. Dies unterscheidet sich von der Open-Source-Argo-CD, bei der mehrere Projekte auf demselben Cluster bereitgestellt werden können. Geben Sie das Projekt bei der Registrierung des Clusters mithilfe des `project` Felds im Clustergeheimnis an.

Verwenden Sie Projekte, um zu steuern, in welchen Clustern und Namespaces Anwendungen bereitgestellt werden können, indem Sie die zulässigen Zielcluster und Namespaces konfigurieren in: `spec.destinations`

```
apiVersion: argoproj.io/v1alpha1
kind: AppProject
metadata:
  name: production
  namespace: argocd
spec:
  destinations:
  - server: arn:aws:eks:us-west-2:111122223333:cluster/prod-cluster
    namespace: '*'
  - server: arn:aws:eks:eu-west-1:111122223333:cluster/prod-eu-cluster
    namespace: '*'
  sourceRepos:
  - 'https://github.com/example/production-apps'
```

Details hierzu finden Sie unter the section called “Projekte”.

Weitere Ressourcen

- the section called “Projekte”- Organisieren Sie Anwendungen und setzen Sie Sicherheitsgrenzen durch
- the section called “Anwendungen erstellen”- Stellen Sie Ihre erste Anwendung bereit
- the section called “ApplicationSets”- Stellen Sie die Lösung auf mehreren Clustern bereit mit ApplicationSets
- the section called “Überlegungen zu Argo CD”- Multi-Cluster-Muster und kontenübergreifende Einrichtung

- Deklaratives Cluster-Setup — Referenz zur Upstream-Cluster-Konfiguration

Zusammenarbeit mit Argo CD Projects

Argo CD Projects (AppProject) bieten logische Gruppierung und Zugriffskontrolle für Anwendungen. Projekte definieren, welche Git-Repositorys, Zielcluster und Namespaces Anwendungen verwenden können, wodurch Mehrmandantenfähigkeit und Sicherheitsgrenzen in gemeinsam genutzten Argo-CD-Instanzen aktiviert werden.

Wann sollten Projekte verwendet werden

Verwenden Sie Projekte, um:

- Trennen Sie Anwendungen nach Team, Umgebung oder Geschäftseinheit
- Schränken Sie ein, aus welchen Repositorys Teams Anwendungen bereitstellen können
- Beschränken Sie, in welchen Clustern und Namespaces Teams Deployment durchführen können
- Setzen Sie Ressourcenkontingente und zulässige Ressourcentypen durch
- Sorgen Sie für die Self-Service-Anwendungsbereitstellung mit Leitplanken

Standardprojekt

Jede Argo-CD-Funktion beinhaltet ein `default` Projekt, das den Zugriff auf alle Repositorys, Cluster und Namespaces ermöglicht. Das ist zwar für erste Tests nützlich, aber Sie können spezielle Projekte mit ausdrücklichen Einschränkungen für den produktiven Einsatz erstellen.

Einzelheiten zur Standard-Projektkonfiguration und wie man sie einschränkt, finden Sie unter Das Standardprojekt in der Argo-CD-Dokumentation.

Erstellen eines Projekts

Erstellen Sie ein Projekt, indem Sie eine `AppProject` Ressource auf Ihren Cluster anwenden.

Beispiel: Teamspezifisches Projekt

```
apiVersion: argoproj.io/v1alpha1
kind: AppProject
metadata:
  name: team-a
  namespace: argocd
```

```
spec:
  description: Applications for Team A

  # Source repositories this project can deploy from
  sourceRepos:
    - 'https://github.com/my-org/team-a-*'
    - 'https://github.com/my-org/shared-libs'

  # Destination clusters and namespaces
  destinations:
    - name: dev-cluster
      namespace: team-a-dev
    - name: prod-cluster
      namespace: team-a-prod

  # Allowed resource types
  clusterResourceWhitelist:
    - group: ''
      kind: Namespace

  namespaceResourceWhitelist:
    - group: 'apps'
      kind: Deployment
    - group: ''
      kind: Service
    - group: ''
      kind: ConfigMap
```

Wenden Sie das Projekt an:

```
kubectl apply -f team-a-project.yaml
```

Konfiguration des Projekts

Quell-Repositorien

Steuern Sie, welche Git-Repositorys Anwendungen in diesem Projekt verwenden können:

```
spec:
  sourceRepos:
    - 'https://github.com/my-org/app-*'  # Wildcard pattern
    - 'https://github.com/my-org/infra'  # Specific repo
```

Du kannst Platzhalter und Negationsmuster (!Präfix) verwenden, um bestimmte Repositorys zuzulassen oder abzulehnen. Einzelheiten finden Sie unter Projekte verwalten in der Argo-CD-Dokumentation.

Einschränkungen im Hinblick auf Ziele

Schränken Sie ein, wo Anwendungen bereitgestellt werden können:

```
spec:
  destinations:
    - name: prod-cluster  # Specific cluster by name
      namespace: production
    - name: '*'  # Any cluster
      namespace: team-a-*  # Namespace pattern
```

> **Important**
>
> Verwenden Sie für Produktionsprojekte spezifische Clusternamen und Namespace-Muster anstelle von Platzhaltern. Dies verhindert versehentliche Bereitstellungen in nicht autorisierten Clustern oder Namespaces.

Sie können Platzhalter und Negationsmuster verwenden, um Ziele zu steuern. Einzelheiten finden Sie unter Projekte verwalten in der Argo-CD-Dokumentation.

Einschränkungen bei den Ressourcen

Steuern Sie, welche Kubernetes-Ressourcentypen bereitgestellt werden können:

Ressourcen im Clusterbereich:

```
spec:
  clusterResourceWhitelist:
    - group: ''
      kind: Namespace
    - group: 'rbac.authorization.k8s.io'
      kind: Role
```

Ressourcen im Namespace-Bereich:

```
spec:
```

```
  namespaceResourceWhitelist:
    - group: 'apps'
      kind: Deployment
    - group: ''
      kind: Service
    - group: ''
      kind: ConfigMap
    - group: 's3.services.k8s.aws'
      kind: Bucket
```

Verwenden Sie schwarze Listen, um bestimmte Ressourcen abzulehnen:

```
spec:
  namespaceResourceBlacklist:
    - group: ''
      kind: Secret  # Prevent direct Secret creation
```

Ordnen Sie Anwendungen Projekten zu

Wenn Sie eine Anwendung erstellen, geben Sie das Projekt im `spec.project` Feld an:

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: my-app
  namespace: argocd
spec:
  project: team-a  # Assign to team-a project
  source:
    repoURL: https://github.com/my-org/my-app
    path: manifests
  destination:
    name: prod-cluster
    namespace: team-a-prod
```

Anwendungen ohne ein bestimmtes Projekt verwenden das `default` Projekt.

Projektrollen und RBAC

Projekte können benutzerdefinierte Rollen für eine differenzierte Zugriffskontrolle definieren. Ordnen Sie in Ihrer Funktionskonfiguration Projektrollen AWS Identity Center-Benutzern und -Gruppen zu, um zu kontrollieren, wer Anwendungen synchronisieren, aktualisieren oder löschen kann.

Beispiel: Projekt mit Entwickler- und Administratorrollen

```
apiVersion: argoproj.io/v1alpha1
kind: AppProject
metadata:
  name: team-a
  namespace: argocd
spec:
  sourceRepos:
    - '*'
  destinations:
    - name: '*'
      namespace: 'team-a-*'

  roles:
    - name: developer
      description: Developers can sync applications
      policies:
        - p, proj:team-a:developer, applications, sync, team-a/*, allow
        - p, proj:team-a:developer, applications, get, team-a/*, allow
      groups:
        - team-a-developers

    - name: admin
      description: Admins have full access
      policies:
        - p, proj:team-a:admin, applications, *, team-a/*, allow
      groups:
        - team-a-admins
```

Einzelheiten zu Projektrollen, JWT-Token für CI/CD Pipelines und der RBAC-Konfiguration finden Sie unter Projektrollen in der Argo-CD-Dokumentation.

Allgemeine Muster

Umweltbezogene Projekte

Erstellen Sie separate Projekte für jede Umgebung:

```
apiVersion: argoproj.io/v1alpha1
kind: AppProject
metadata:
  name: production
```

```
  namespace: argocd
spec:
  sourceRepos:
    - 'https://github.com/my-org/*'
  destinations:
    - name: prod-cluster
      namespace: '*'
  # Strict resource controls for production
  clusterResourceWhitelist: []
  namespaceResourceWhitelist:
    - group: 'apps'
      kind: Deployment
    - group: ''
      kind: Service
```

Teambasierte Projekte

Isolieren Sie Teams mit speziellen Projekten:

```
apiVersion: argoproj.io/v1alpha1
kind: AppProject
metadata:
  name: platform-team
  namespace: argocd
spec:
  sourceRepos:
    - 'https://github.com/my-org/platform-*'
  destinations:
    - name: '*'
      namespace: 'platform-*'
  # Platform team can manage cluster resources
  clusterResourceWhitelist:
    - group: '*'
      kind: '*'
```

Projekte mit mehreren Clustern

Stellen Sie die Lösung auf mehreren Clustern mit konsistenten Richtlinien bereit:

```
apiVersion: argoproj.io/v1alpha1
kind: AppProject
metadata:
```

```
  name: global-app
  namespace: argocd
spec:
  sourceRepos:
    - 'https://github.com/my-org/global-app'
  destinations:
    - name: us-west-cluster
      namespace: app
    - name: eu-west-cluster
      namespace: app
    - name: ap-south-cluster
      namespace: app
```

Best Practices

Beginnen Sie mit restriktiven Projekten: Beginnen Sie mit engen Berechtigungen und erweitern Sie diese nach Bedarf, anstatt mit einem breiten Zugriff zu beginnen.

Verwenden Sie Namespace-Muster: Nutzen Sie Platzhalter in Namespace-Einschränkungen (wie`team-a-*`), um Flexibilität zu gewährleisten und gleichzeitig die Grenzen beizubehalten.

Separate Produktionsprojekte: Verwenden Sie spezielle Projekte für die Produktion mit strengeren Kontrollen und manuellen Synchronisierungsrichtlinien.

Projektzwecke dokumentieren: Verwenden Sie das `description` Feld, um zu erklären, wofür jedes Projekt gedacht ist und wer es verwenden sollte.

Überprüfen Sie die Projektberechtigungen regelmäßig: Prüfen Sie Projekte regelmäßig, um sicherzustellen, dass die Einschränkungen weiterhin den Teamanforderungen und Sicherheitsanforderungen entsprechen.

Weitere Ressourcen

- the section called “Konfigurieren von -Berechtigungen”- Konfigurieren Sie die Integration von RBAC und Identity Center
- the section called “Anwendungen erstellen”- Erstellen Sie Anwendungen innerhalb von Projekten
- the section called “ApplicationSets”- Verwendung ApplicationSets mit Projekten für Bereitstellungen mit mehreren Clustern
- Dokumentation zu Argo CD Projects — Vollständige Upstream-Referenz

Anwendungen erstellen

Anwendungen stellen Bereitstellungen in Zielclustern dar. Jede Anwendung definiert eine Quelle (Git-Repository) und ein Ziel (Cluster und Namespace). Wenn es angewendet wird, erstellt Argo CD die durch Manifeste im Git-Repository angegebenen Ressourcen für den Namespace im Cluster. Anwendungen spezifizieren häufig Workload-Bereitstellungen, können aber alle im Zielcluster verfügbaren Kubernetes-Ressourcen verwalten.

Voraussetzungen

- Es wurde ein EKS-Cluster mit der Argo-CD-Funktion erstellt
- Der Repository-Zugriff ist konfiguriert (siehethe section called “Repositorys konfigurieren”)
- Der Zielcluster ist registriert (siehethe section called “Registrieren Sie Cluster”)
- `kubectl`für die Kommunikation mit Ihrem Cluster konfiguriert

Erstellen Sie eine Basisanwendung

Definieren Sie eine Anwendung, die aus einem Git-Repository bereitgestellt wird:

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: guestbook
  namespace: argocd
spec:
  project: default
  source:
    repoURL: https://github.com/argoproj/argocd-example-apps
    targetRevision: HEAD
    path: guestbook
  destination:
    name: in-cluster
    namespace: default
```

> **Note**
>
> Verwenden Sie es `destination.name` mit dem Clusternamen, den Sie bei der Registrierung des Clusters verwendet haben (wie `in-cluster` für den lokalen Cluster). Das

`destination.server` Feld funktioniert auch mit EKS-Clustern ARNs, aus Gründen der besseren Lesbarkeit wird jedoch die Verwendung von Clusternamen empfohlen.

Wenden Sie die Anwendung an:

```
kubectl apply -f application.yaml
```

Den Status der Bewerbung anzeigen:

```
kubectl get application guestbook -n argocd
```

Konfiguration der Quelle

Git-Repository:

```
spec:
  source:
    repoURL: https://github.com/example/my-app
    targetRevision: main
    path: kubernetes/manifests
```

Spezifisches Git-Tag oder Commit:

```
spec:
  source:
    targetRevision: v1.2.0  # or commit SHA
```

Helmdiagramm:

```
spec:
  source:
    repoURL: https://github.com/example/helm-charts
    targetRevision: main
    path: charts/my-app
    helm:
      valueFiles:
      - values.yaml
      parameters:
```

```
      - name: image.tag
        value: v1.2.0
```

Helmdiagramm mit Werten aus dem externen Git-Repository (Multisource-Muster):

```
spec:
  sources:
  - repoURL: https://github.com/example/helm-charts
    targetRevision: main
    path: charts/my-app
    helm:
      valueFiles:
      - $values/environments/production/values.yaml
  - repoURL: https://github.com/example/config-repo
    targetRevision: main
    ref: values
```

Weitere Informationen finden Sie unter Helm Value Files from External Git Repository in der Argo-CD-Dokumentation.

Helm-Diagramm aus ECR:

```
spec:
  source:
    repoURL: oci://account-id.dkr.ecr.region.amazonaws.com/repository-name
    targetRevision: chart-version
    chart: chart-name
```

Wenn die Capability-Rolle über die erforderlichen ECR-Berechtigungen verfügt, wird das Repository direkt verwendet und es ist keine Repository-Konfiguration erforderlich. Details dazu finden Sie unter the section called “Repositorys konfigurieren”.

Git-Repository von CodeCommit:

```
spec:
  source:
    repoURL: https://git-codecommit.region.amazonaws.com/v1/repos/repository-name
    targetRevision: main
    path: kubernetes/manifests
```

Wenn die Capability-Rolle über die erforderlichen CodeCommit Berechtigungen verfügt, wird das Repository direkt verwendet und es ist keine Repository-Konfiguration erforderlich. Details dazu finden Sie unter the section called “Repositorys konfigurieren”.

Git-Repository von CodeConnections:

```
spec:
  source:
    repoURL: https://codeconnections.region.amazonaws.com/git-http/account-
id/region/connection-id/owner/repository.git
    targetRevision: main
    path: kubernetes/manifests
```

Das Repository-URL-Format wird vom CodeConnections Verbindungs-ARN abgeleitet. Wenn die Capability-Rolle über die erforderlichen CodeConnections Berechtigungen verfügt und eine Verbindung konfiguriert ist, wird das Repository direkt verwendet und es ist keine Repository-Konfiguration erforderlich. Details dazu finden Sie unter the section called “Repositorys konfigurieren”.

Anpassen:

```
spec:
  source:
    repoURL: https://github.com/example/kustomize-app
    targetRevision: main
    path: overlays/production
    kustomize:
      namePrefix: prod-
```

Richtlinien synchronisieren

Steuern Sie, wie Argo CD Anwendungen synchronisiert.

Manuelle Synchronisation (Standard):

Für die Synchronisierung von Anwendungen ist eine manuelle Genehmigung erforderlich:

```
spec:
  syncPolicy: {}  # No automated sync
```

Synchronisation manuell auslösen:

```
kubectl patch application guestbook -n argocd \
  --type merge \
  --patch '{"operation": {"initiatedBy": {"username": "admin"}, "sync": {}}}'
```

Automatische Synchronisation:

Anwendungen werden automatisch synchronisiert, wenn Git-Änderungen erkannt werden:

```
spec:
  syncPolicy:
    automated: {}
```

Selbstheilung:

Manuelle Änderungen am Cluster automatisch rückgängig machen:

```
spec:
  syncPolicy:
    automated:
      selfHeal: true
```

Wenn diese Option aktiviert ist, macht Argo CD alle manuellen Änderungen rückgängig, die direkt am Cluster vorgenommen wurden, und stellt so sicher, dass Git die Quelle der Wahrheit bleibt.

Beschneiden:

Automatisches Löschen von Ressourcen, die aus Git entfernt wurden:

```
spec:
  syncPolicy:
    automated:
      prune: true
```

> **Warning**
>
> Durch das Bereinigen werden Ressourcen aus Ihrem Cluster gelöscht. In Produktionsumgebungen mit Vorsicht verwenden.

Kombinierte automatische Synchronisation:

```
spec:
  syncPolicy:
    automated:
      prune: true
      selfHeal: true
    syncOptions:
    - CreateNamespace=true
```

Konfiguration erneut versuchen:

Konfigurieren Sie das Wiederholungsverhalten für fehlgeschlagene Synchronisierungen:

```
spec:
  syncPolicy:
    retry:
      limit: 5  # Number of failed sync attempts; unlimited if less than 0
      backoff:
        duration: 5s  # Amount to back off (default unit: seconds, also supports "2m",
 "1h")
        factor: 2  # Factor to multiply the base duration after each failed retry
        maxDuration: 3m  # Maximum amount of time allowed for the backoff strategy
```

Dies ist besonders nützlich für Ressourcen, die darauf angewiesen sind, dass CRDs sie zuerst erstellt wurden, oder wenn Sie mit Kro-Instanzen arbeiten, bei denen die CRD möglicherweise nicht sofort verfügbar ist.

Synchronisierungsoptionen

Zusätzliche Synchronisierungskonfiguration:

Erstellen Sie einen Namespace, falls er nicht existiert:

```
spec:
  syncPolicy:
    syncOptions:
    - CreateNamespace=true
```

Probelauf wegen fehlender Ressourcen überspringen:

Nützlich beim Anwenden von Ressourcen, CRDs die davon abhängen, die noch nicht existieren (wie Kro-Instanzen):

```
spec:
  syncPolicy:
    syncOptions:
    - SkipDryRunOnMissingResource=true
```

Dies kann auch auf bestimmte Ressourcen angewendet werden, indem ein Label auf der Ressource selbst verwendet wird.

Überprüfen Sie die Ressourcen, bevor Sie sie anwenden:

```
spec:
  syncPolicy:
    syncOptions:
    - Validate=true
```

Nur unsynchron anwenden:

```
spec:
  syncPolicy:
    syncOptions:
    - ApplyOutOfSyncOnly=true
```

Erweiterte Synchronisierungsfunktionen

Argo CD unterstützt erweiterte Synchronisierungsfunktionen für komplexe Bereitstellungen:

- Sync Waves — Steuern Sie die Reihenfolge der Ressourcenerstellung mit Anmerkungen `argocd.argoproj.io/sync-wave`
- Hooks synchronisieren — Führt Jobs vor oder nach der Synchronisierung mit `argocd.argoproj.io/hook` Anmerkungen aus (PreSync,, PostSync) SyncFail
- Bewertung des Ressourcenzustands — Benutzerdefinierte Zustandsprüfungen für anwendungsspezifische Ressourcen

Einzelheiten finden Sie unter Sync Waves und Resource Hooks in der Argo-CD-Dokumentation.

Ignoriere Unterschiede

Verhindern Sie, dass Argo CD bestimmte Felder synchronisiert, die von anderen Controllern verwaltet werden (wie HPA, das Replikate verwaltet):

```
spec:
  ignoreDifferences:
  - group: apps
    kind: Deployment
    jsonPointers:
    - /spec/replicas
```

Einzelheiten zu Ignorierungsmustern und Feldausschlüssen finden Sie unter Diffing Customization in der Argo-CD-Dokumentation.

Einsatz in mehreren Umgebungen

Stellen Sie dieselbe Anwendung in mehreren Umgebungen bereit:

Entwicklung:

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: my-app-dev
  namespace: argocd
spec:
  project: default
  source:
    repoURL: https://github.com/example/my-app
    targetRevision: develop
    path: overlays/development
  destination:
    name: dev-cluster
    namespace: my-app
```

Produktion:

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: my-app-prod
  namespace: argocd
spec:
  project: default
  source:
    repoURL: https://github.com/example/my-app
```

```
    targetRevision: main
    path: overlays/production
  destination:
    name: prod-cluster
    namespace: my-app
  syncPolicy:
    automated:
      prune: true
      selfHeal: true
```

Anwendungen überwachen und verwalten

Status der Bewerbung anzeigen:

```
kubectl get application my-app -n argocd
```

Greifen Sie auf die Benutzeroberfläche von Argo CD zu:

Öffnen Sie die Benutzeroberfläche von Argo CD über die EKS-Konsole, um die Anwendungstopologie, den Synchronisierungsstatus, den Zustand der Ressourcen und den Bereitstellungsverlauf einzusehen. Anweisungen the section called “Arbeiten mit Argo CD” zum Zugriff auf die Benutzeroberfläche finden Sie unter.

Rollback-Anwendungen:

Rollback zu einer früheren Version mithilfe der Argo-CD-Benutzeroberfläche, der Argo-CD-CLI oder durch Aktualisierung der Anwendungsspezifikation `targetRevision` auf einen früheren Git-Commit oder -Tag.

Verwenden der Argo-CD-CLI:

```
argocd app rollback argocd/my-app <revision-id>
```

> **Note**
>
> Wenn Sie die Argo CD-CLI mit der verwalteten Funktion verwenden, geben Sie Anwendungen mit dem Namespace-Präfix an:. `namespace/appname`

Weitere Informationen finden Sie unter argocd app rollback in der Argo-CD-Dokumentation.

Weitere Ressourcen

- the section called “Projekte”- Organisieren Sie Anwendungen mit Projekten für Umgebungen mit mehreren Mandanten
- the section called “ApplicationSets”- Stellen Sie die Lösung mithilfe von Vorlagen auf mehreren Clustern bereit
- Anwendungsspezifikation — Vollständige Anwendungs-API-Referenz
- Synchronisierungsoptionen — Erweiterte Synchronisierungskonfiguration

Benutzen ApplicationSets

ApplicationSets Generieren Sie mehrere Anwendungen anhand von Vorlagen, sodass Sie dieselbe Anwendung mit einer einzigen Ressourcendefinition in mehreren Clustern, Umgebungen oder Namespaces bereitstellen können.

Voraussetzungen

- Es wurde ein EKS-Cluster mit der Argo-CD-Funktion erstellt
- Der Repository-Zugriff ist konfiguriert (siehethe section called “Repositorys konfigurieren”)
- `kubectl`für die Kommunikation mit Ihrem Cluster konfiguriert

> **Note**
>
> Für sind nicht mehrere Zielcluster erforderlich ApplicationSets. Sie können andere Generatoren als den Cluster-Generator (wie Listen-, Git- oder Matrix-Generatoren) verwenden, um Anwendungen ohne Remote-Cluster bereitzustellen.

Wie ApplicationSets funktionieren

ApplicationSets Verwenden Sie Generatoren, um Parameter zu erzeugen, und wenden Sie diese Parameter dann auf eine Anwendungsvorlage an. Jeder Satz generierter Parameter erstellt eine Anwendung.

Allgemeine Generatoren für EKS-Bereitstellungen:

- Listengenerator — Definieren Sie explizit Cluster und Parameter für jede Umgebung

- Cluster-Generator — Automatische Bereitstellung auf allen registrierten Clustern
- Git Generator - Generieren Sie Anwendungen aus der Repository-Struktur
- Matrixgenerator — Kombinieren Sie Generatoren für mehrdimensionale Bereitstellungen
- Generator zusammenführen — Fügt Parameter aus mehreren Generatoren zusammen

Eine vollständige Generator-Referenz finden Sie in der ApplicationSet Dokumentation.

Listengenerator

Bereitstellung auf mehreren Clustern mit expliziter Konfiguration:

```
apiVersion: argoproj.io/v1alpha1
kind: ApplicationSet
metadata:
  name: guestbook-all-clusters
  namespace: argocd
spec:
  generators:
  - list:
      elements:
      - environment: dev
        replicas: "2"
      - environment: staging
        replicas: "3"
      - environment: prod
        replicas: "5"
  template:
    metadata:
      name: 'guestbook-{{environment}}'
    spec:
      project: default
      source:
        repoURL: https://github.com/example/guestbook
        targetRevision: HEAD
        path: 'overlays/{{environment}}'
      destination:
        name: '{{environment}}-cluster'
        namespace: guestbook
      syncPolicy:
        automated:
          prune: true
```

```
        selfHeal: true
```

> **Note**
>
> Zur besseren Lesbarkeit `destination.name` zusammen mit Clusternamen verwenden. Das `destination.server` Feld funktioniert bei ARNs Bedarf auch mit EKS-Clustern.

Dadurch werden drei Anwendungen erstellt: `guestbook-devguestbook-staging`, und`guestbook-prod`.

Clustergenerator

Automatisch auf allen registrierten Clustern bereitstellen:

```
apiVersion: argoproj.io/v1alpha1
kind: ApplicationSet
metadata:
  name: cluster-addons
  namespace: argocd
spec:
  generators:
  - clusters: {}
  template:
    metadata:
      name: '{{name}}-addons'
    spec:
      project: default
      source:
        repoURL: https://github.com/example/cluster-addons
        targetRevision: HEAD
        path: addons
      destination:
        server: '{{server}}'
        namespace: kube-system
      syncPolicy:
        automated:
          prune: true
          selfHeal: true
```

Dadurch wird automatisch eine Anwendung für jeden registrierten Cluster erstellt.

Cluster filtern:

Wird verwendet`matchLabels`, um bestimmte Cluster einzubeziehen oder `matchExpressions` um Cluster auszuschließen:

```
spec:
  generators:
  - clusters:
      selector:
        matchLabels:
          environment: production
        matchExpressions:
        - key: skip-appset
          operator: DoesNotExist
```

Git-Generatoren

Git-Generatoren erstellen Anwendungen, die auf der Repository-Struktur basieren:

- Verzeichnisgenerator — Stellen Sie jedes Verzeichnis als separate Anwendung bereit (nützlich für Microservices)
- Dateigenerator — Generieren Sie Anwendungen aus Parameterdateien (nützlich für Bereitstellungen mit mehreren Mandanten)

Beispiel: Bereitstellung von Microservices

```
apiVersion: argoproj.io/v1alpha1
kind: ApplicationSet
metadata:
  name: microservices
  namespace: argocd
spec:
  generators:
  - git:
      repoURL: https://github.com/example/microservices
      revision: HEAD
      directories:
      - path: services/*
  template:
    metadata:
      name: '{{path.basename}}'
```

```
    spec:
      project: default
      source:
        repoURL: https://github.com/example/microservices
        targetRevision: HEAD
        path: '{{path}}'
      destination:
        name: my-cluster
        namespace: '{{path.basename}}'
      syncPolicy:
        automated:
          prune: true
          selfHeal: true
        syncOptions:
        - CreateNamespace=true
```

Einzelheiten zu Git-Generatoren und dateibasierter Konfiguration finden Sie unter Git Generator in der Argo-CD-Dokumentation.

Matrixgenerator

Kombinieren Sie mehrere Generatoren, um sie in mehreren Dimensionen (Umgebungen × Cluster) einzusetzen:

```
apiVersion: argoproj.io/v1alpha1
kind: ApplicationSet
metadata:
  name: multi-env-multi-cluster
  namespace: argocd
spec:
  generators:
  - matrix:
      generators:
      - list:
          elements:
          - environment: dev
          - environment: staging
          - environment: prod
      - clusters:
          selector:
            matchLabels:
              region: us-west-2
  template:
```

```
    metadata:
      name: 'app-{{environment}}-{{name}}'
    spec:
      project: default
      source:
        repoURL: https://github.com/example/app
        targetRevision: HEAD
        path: 'overlays/{{environment}}'
      destination:
        name: '{{name}}'
        namespace: 'app-{{environment}}'
```

Einzelheiten zur Kombination von Generatoren finden Sie unter Matrix Generator in der Argo-CD-Dokumentation.

Einsatz in mehreren Regionen

Bereitstellung auf Clustern in mehreren Regionen:

```
apiVersion: argoproj.io/v1alpha1
kind: ApplicationSet
metadata:
  name: global-app
  namespace: argocd
spec:
  generators:
  - list:
      elements:
      - clusterName: prod-us-west
        region: us-west-2
      - clusterName: prod-us-east
        region: us-east-1
      - clusterName: prod-eu-west
        region: eu-west-1
  template:
    metadata:
      name: 'app-{{region}}'
    spec:
      project: default
      source:
        repoURL: https://github.com/example/app
        targetRevision: HEAD
        path: kubernetes
```

```
        helm:
          parameters:
          - name: region
            value: '{{region}}'
      destination:
        name: '{{clusterName}}'
        namespace: app
      syncPolicy:
        automated:
          prune: true
          selfHeal: true
```

Verwalten ApplicationSets

Anwendungen anzeigen ApplicationSets und generieren:

```
kubectl get applicationsets -n argocd
kubectl get applications -n argocd -l argocd.argoproj.io/application-set-
name=<applicationset-name>
```

Aktualisieren und ApplicationSet:

Ändern Sie die ApplicationSet Spezifikation und wenden Sie sie erneut an. Argo CD aktualisiert automatisch alle generierten Anwendungen:

```
kubectl apply -f applicationset.yaml
```

Löschen Sie und ApplicationSet:

```
kubectl delete applicationset <name> -n argocd
```

> **Warning**
>
> Beim Löschen und ApplicationSet Löschen aller generierten Anwendungen werden alle generierten Anwendungen gelöscht. Wenn dies bei diesen Anwendungen der Fall ist`prune: true`, werden ihre Ressourcen ebenfalls aus den Zielclustern gelöscht.
> Um bereitgestellte Ressourcen beim Löschen eines `.syncPolicy.preserveResourcesOnDeletion` zu erhalten ApplicationSet, ist `true` in der ApplicationSet Spezifikation auf festgelegt. Weitere Informationen finden Sie unter Anwendungsbereinigung und Löschen von Ressourcen in der Argo-CD-Dokumentation.

> **Important**
>
> Die ApplicationSets Funktion von Argo CD beinhaltet Sicherheitsaspekte, die Sie vor der Verwendung beachten sollten. ApplicationSets Weitere Informationen finden Sie in der Argo-CD-Dokumentation unter ApplicationSet Sicherheit.

Weitere Ressourcen

- the section called “Projekte”- Organisieren Sie ApplicationSets mit Projekten
- the section called “Anwendungen erstellen”- Verstehen Sie die Anwendungskonfiguration
- ApplicationSet Dokumentation — Vollständige Generatorreferenz und Muster
- Generator-Referenz — Detaillierte Generatorspezifikationen

Überlegungen zu Argo CD

In diesem Thema werden wichtige Überlegungen zur Nutzung der EKS-Funktion für Argo CD behandelt, darunter Planung, Berechtigungen, Authentifizierung und Bereitstellungsmuster für mehrere Cluster.

Planung

Beachten Sie vor der Bereitstellung von Argo CD Folgendes:

Repository-Strategie: Legen Sie fest, wo Ihre Anwendungsmanifeste gespeichert werden (CodeCommit,, GitHub GitLab, Bitbucket). Plane deine Repository-Struktur und Branching-Strategie für verschiedene Umgebungen.

RBAC-Strategie: Planen Sie, welche Teams oder Benutzer Administrator-, Redakteur- oder Viewer-Zugriff haben sollen. Ordnen Sie diese AWS Identity Center-Gruppen oder Argo-CD-Rollen zu.

Multi-Cluster-Architektur: Stellen Sie fest, ob Sie mehrere Cluster von einer einzigen Argo-CD-Instance aus verwalten möchten. Erwägen Sie die Verwendung eines dedizierten Management-Clusters für Argo CD.

Organisation der Anwendungen: Planen Sie, wie Sie Anwendungen strukturieren und ApplicationSets. Erwägen Sie die Verwendung von Projekten, um Anwendungen nach Team oder Umgebung zu organisieren.

Synchronisierungsrichtlinien: Entscheiden Sie, ob Anwendungen automatisch synchronisiert werden sollen oder ob eine manuelle Genehmigung erforderlich ist. Automatisierte Synchronisation ist in der Entwicklung üblich, manuelle in der Produktion.

Berechtigungen

Ausführliche Informationen zu IAM-Fähigkeitsrollen, Vertrauensrichtlinien und bewährten Sicherheitsmethoden finden Sie unter the section called “Funktion: IAM-Rolle” undthe section called “Überlegungen zu den Funktionen von EKS”.

Überblick über die IAM-Fähigkeitsrollen

Wenn Sie eine Argo-CD-Funktionsressource erstellen, geben Sie eine IAM-Fähigkeitsrolle an. Im Gegensatz zu ACK verwaltet Argo CD hauptsächlich Kubernetes-Ressourcen, nicht Ressourcen direkt. AWS Die IAM-Capability Role ist jedoch erforderlich für:

- Zugriff auf private Git-Repositorys in CodeCommit
- Integration mit AWS Identity Center zur Authentifizierung
- Zugreifen auf AWS Geheimnisse im Secrets Manager (falls konfiguriert)
- Clusterübergreifende Bereitstellungen auf anderen EKS-Clustern

CodeCommit Integration

Wenn Sie CodeCommit Repositorys verwenden, fügen Sie eine Richtlinie mit Leseberechtigungen hinzu:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codecommit:GitPull"
      ],
      "Resource": "*"
    }
  ]
}
```

> **Important**
>
> Beschränken Sie das `Resource` Feld für Produktionszwecke auf ein bestimmtes Repository, ARNs anstatt es zu verwenden`"*"`.
> Beispiel:
>
> ```
> "Resource": "arn:aws:codecommit:us-west-2:111122223333:my-app-repo"
> ```
>
> Dadurch wird der Zugriff der Argo-CD-Funktion nur auf die Repositorys beschränkt, die sie verwalten muss.

Integration von Secrets Manager

Wenn Sie Repository-Anmeldeinformationen in Secrets Manager speichern, fügen Sie die verwaltete Richtlinie für den Lesezugriff hinzu:

```
arn:aws:iam::aws:policy/AWSSecretsManagerClientReadOnlyAccess
```

Diese Richtlinie umfasst die erforderlichen Berechtigungen: `secretsmanager:GetSecretValuesecretsmanager:DescribeSecret`, und KMS-Entschlüsselungsberechtigungen.

Grundlegende Einrichtung

Für die grundlegende Funktionalität von Argo CD mit öffentlichen Git-Repositorys sind außer der Vertrauensrichtlinie keine zusätzlichen IAM-Richtlinien erforderlich.

Authentifizierung

AWS Identity Center-Integration

Die verwaltete Funktion von Argo CD lässt sich direkt in AWS Identity Center (früher AWS SSO) integrieren, sodass Sie Ihren vorhandenen Identitätsanbieter für die Authentifizierung verwenden können.

Wenn Sie die AWS Identity Center-Integration konfigurieren:

1. Benutzer greifen über die EKS-Konsole auf die Benutzeroberfläche von Argo CD zu

2. Sie authentifizieren sich mithilfe von AWS Identity Center (das eine Verbindung zu Ihrem Corporate Identity Provider herstellen kann)
3. AWS Identity Center stellt Benutzer- und Gruppeninformationen für Argo CD bereit
4. Argo CD ordnet Benutzer und Gruppen basierend auf Ihrer Konfiguration RBAC-Rollen zu
5. Benutzer sehen nur die Anwendungen und Ressourcen, für deren Zugriff sie berechtigt sind

Vereinfachung des Zugriffs mit Identity Center-Berechtigungssätzen

AWS Identity Center bietet bei der Arbeit mit Argo CD zwei unterschiedliche Authentifizierungspfade:

Argo CD API-Authentifizierung: Identity Center bietet SSO-Authentifizierung für die Benutzeroberfläche und API von Argo CD. Dies wird über die RBAC-Rollenzuordnungen der Argo-CD-Funktion konfiguriert.

EKS-Clusterzugriff: Die Argo-CD-Funktion verwendet die vom Kunden bereitgestellte IAM-Rolle, um sich über Zugriffseinträge bei EKS-Clustern zu authentifizieren. Diese Zugriffseinträge können manuell konfiguriert werden, um Berechtigungen hinzuzufügen oder zu entfernen.

Sie können Identity Center-Berechtigungssätze verwenden, um das Identitätsmanagement zu vereinfachen, indem Sie einer einzigen Identität den Zugriff auf Argo CD- und EKS-Cluster ermöglichen. Dies reduziert den Aufwand, da Sie nur eine Identität auf beiden Systemen verwalten müssen, anstatt separate Anmeldeinformationen für den Argo-CD-Zugriff und den Clusterzugriff zu verwalten.

RBAC-Rollenzuordnungen

Argo CD verfügt über integrierte Rollen, die Sie AWS Identity Center-Benutzern und Gruppen zuordnen können:

ADMIN: Vollzugriff auf alle Anwendungen und Einstellungen. Kann Anwendungen erstellen, aktualisieren und löschen. Kann die Argo-CD-Konfiguration verwalten.

EDITOR: Kann Anwendungen erstellen und ändern. Die Argo-CD-Einstellungen können nicht geändert oder Anwendungen gelöscht werden.

VIEWER: Schreibgeschützter Zugriff auf Anwendungen. Kann den Status und den Verlauf der Bewerbung einsehen. Es können keine Änderungen vorgenommen werden.

> **Note**
>
> Bei Rollennamen wird zwischen Groß- und Kleinschreibung unterschieden und sie müssen in Großbuchstaben geschrieben werden (ADMIN, EDITOR, VIEWER).

> **Important**
>
> Die Integration von EKS Capabilities in AWS Identity Center unterstützt bis zu 1.000 Identitäten pro Argo-CD-Funktion. Eine Identität kann ein Benutzer oder eine Gruppe sein.

Bereitstellungen mit mehreren Clustern

Die verwaltete Argo-CD-Funktion unterstützt Bereitstellungen mit mehreren Clustern, sodass Sie Anwendungen für Entwicklungs-, Staging- und Produktionscluster von einer einzigen Argo-CD-Instance aus verwalten können.

So funktioniert Multi-Cluster

Wenn Sie zusätzliche Cluster bei Argo CD registrieren:

1. Sie erstellen Clustergeheimnisse, die per ARN auf EKS-Zielcluster verweisen.
2. Sie erstellen Anwendungen oder ApplicationSets die auf verschiedene Cluster abzielen
3. Argo CD stellt eine Verbindung zu jedem Cluster her, um Ressourcen bereitzustellen und zu überwachen
4. Sie können alle Cluster von einer einzigen Argo-CD-Benutzeroberfläche aus anzeigen und verwalten

Voraussetzungen für Multi-Cluster

Bevor Sie weitere Cluster registrieren:

- Erstellen Sie einen Access-Eintrag auf dem Zielcluster für die Argo-CD-Capability-Rolle
- Stellen Sie die Netzwerkkonnektivität zwischen der Argo-CD-Funktion und den Zielclustern sicher
- Überprüfen Sie die IAM-Berechtigungen für den Zugriff auf die Zielcluster

Registrieren Sie einen Cluster

Registrieren Sie Cluster mithilfe von Kubernetes Secrets im `argocd` Namespace.

Rufen Sie den Zielcluster-ARN ARN. *`region-code`*Ersetzen Sie es durch die AWS Region, in der sich Ihr Zielcluster befindet, und *`target-cluster`* ersetzen Sie es durch den Namen Ihres Zielclusters.

```
aws eks describe-cluster \
  --region region-code \
  --name target-cluster \
  --query 'cluster.arn' \
  --output text
```

Erstellen Sie mit dem Cluster-ARN ein Cluster-Geheimnis:

```
apiVersion: v1
kind: Secret
metadata:
  name: target-cluster
  namespace: argocd
  labels:
    argocd.argoproj.io/secret-type: cluster
type: Opaque
stringData:
  name: target-cluster
  server: arn:aws:eks:us-west-2:111122223333:cluster/target-cluster
  project: default
```

> **⚠ Important**
>
> Verwenden Sie den EKS-Cluster-ARN im `server` Feld, nicht die Kubernetes-API-Server-URL. Die verwaltete Funktion erfordert die Identifizierung ARNs von Zielclustern.

Wenden Sie das Geheimnis an:

```
kubectl apply -f cluster-secret.yaml
```

Access Entry auf dem Zielcluster konfigurieren

Der Zielcluster muss über einen Access-Eintrag verfügen, der der Argo-CD-Capability-Rolle die Berechtigung zur Bereitstellung von Anwendungen erteilt. *region-code*Ersetzen Sie es durch die AWS Region, in der sich Ihr Zielcluster befindet, *target-cluster* ersetzen Sie es durch den Namen Ihres Zielclusters und ersetzen Sie den ARN durch den ARN Ihrer Argo-CD-Funktionsrolle.

```
aws eks create-access-entry \
  --region region-code \
  --cluster-name target-cluster \
  --principal-arn arn:aws:iam::[.replaceable]111122223333:role/ArgoCDCapabilityRole \
  --type STANDARD \
  --kubernetes-groups system:masters
```

> **Note**
>
> Für den Einsatz in der Produktion sollten Sie erwägen, restriktivere Kubernetes-Gruppen anstelle von zu verwenden. `system:masters`

Privater Cluster-Zugriff

Die verwaltete Argo-CD-Funktion kann auf vollständig privaten EKS-Clustern bereitgestellt werden, ohne dass VPC-Peering oder eine spezielle Netzwerkkonfiguration erforderlich sind. AWS verwaltet automatisch die Konnektivität zwischen der Argo-CD-Funktion und privaten Remote-Clustern. Stellen Sie sicher, dass Ihre Repository-Zugriffskontrollen und die Argo CD RBAC-Richtlinien ordnungsgemäß konfiguriert sind.

Kontoübergreifende Bereitstellungen

Für kontoübergreifende Bereitstellungen fügen Sie die Argo CD IAM Capability Role aus dem Quellkonto zum EKS-Zugriffseintrag des Zielclusters hinzu:

1. Erstellen Sie im Zielkonto einen Access-Eintrag auf dem EKS-Zielcluster
2. Verwenden Sie den Argo CD IAM Capability Role ARN aus dem Quellkonto als Principal
3. Konfigurieren Sie die entsprechenden Kubernetes-RBAC-Berechtigungen für den Access-Eintrag
4. Registrieren Sie den Zielcluster in Argo CD mit seinem EKS-Cluster-ARN

Es ist keine zusätzliche IAM-Rollenerstellung oder Konfiguration von Vertrauensrichtlinien erforderlich — EKS Access Entries kümmern sich um den kontoübergreifenden Zugriff.

Best Practices

Verwenden Sie deklarative Quellen als Informationsquelle: Speichern Sie alle Ihre Anwendungsmanifeste in deklarativen Quellen (Git-Repositorys, Helm-Registries oder OCI-Images), um Versionskontrolle, Audit-Trails und Zusammenarbeit zu ermöglichen.

Richtiges RBAC implementieren: Verwenden Sie die AWS Identity Center-Integration, um zu kontrollieren, wer auf Anwendungen in Argo CD zugreifen und diese verwalten kann. Argo CD unterstützt eine differenzierte Zugriffskontrolle auf Ressourcen innerhalb von Anwendungen (Deployments, Pods, Secrets). ConfigMaps

Verwendung ApplicationSets für Bereitstellungen in mehreren Umgebungen: Wird verwendet, ApplicationSets um Anwendungen in mehreren Clustern oder Namespaces mit unterschiedlichen Konfigurationen bereitzustellen.

Lebenszyklusmanagement

Richtlinien zur Anwendungssynchronisierung

Steuern Sie, wie Argo CD Anwendungen synchronisiert:

Manuelle Synchronisierung: Anwendungen müssen manuell genehmigt werden, um Änderungen zu synchronisieren. Für Produktionsumgebungen empfohlen.

Automatische Synchronisierung: Anwendungen werden automatisch synchronisiert, wenn Git-Änderungen erkannt werden. Häufig in Entwicklungs- und Staging-Umgebungen.

Selbstheilung: Automatische Wiederherstellung manueller Änderungen am Cluster. Stellt sicher, dass der Clusterstatus mit Git übereinstimmt.

Bereinigen: Automatisches Löschen von Ressourcen, die aus Git entfernt wurden. Verwenden Sie dies mit Vorsicht, da dadurch Ressourcen gelöscht werden können.

Integrität der Anwendung

Argo CD überwacht kontinuierlich den Zustand der Anwendung:

- Fehlerfrei: Alle Ressourcen laufen wie erwartet

- Fortschritt: Ressourcen werden erstellt oder aktualisiert
- Eingeschränkt: Einige Ressourcen sind nicht funktionsfähig
- Ausgesetzt: Die Anwendung wurde angehalten
- Fehlt: Ressourcen fehlen im Cluster

Synchronisieren Sie Fenster

Konfigurieren Sie Synchronisierungsfenster, um zu steuern, wann Anwendungen synchronisiert werden können:

- Synchronisierungen nur während Wartungsfenstern zulassen
- Blockieren Sie Synchronisierungen während der Geschäftszeiten
- Planen Sie automatische Synchronisierungen für bestimmte Zeiten
- Verwenden Sie Synchronisierungsfenster in Situationen, in denen Sie Änderungen vornehmen und Synchronisierungen beenden müssen (Break-Glass-Szenarien)

Webhook-Konfiguration für eine schnellere Synchronisierung

Standardmäßig fragt Argo CD alle 6 Minuten Git-Repositorys ab, um Änderungen zu erkennen. Für reaktionsschnellere Bereitstellungen konfigurieren Sie Git-Webhooks so, dass sie sofortige Synchronisierungen auslösen, wenn Änderungen übertragen werden.

Webhooks bieten mehrere Vorteile:

- Sofortige Synchronisierungsreaktion, wenn Code übertragen wird (Sekunden statt Minuten)
- Reduzierter Mehraufwand für Abfragen und verbesserte Systemleistung
- Effizientere Nutzung von API-Ratenbegrenzungen
- Bessere Benutzererfahrung mit schnellerem Feedback

Webhook-Endpunkt

Die Webhook-URL folgt dem Muster`${serverUrl}/api/webhook`, wo sich Ihre Argo-CD-Server-URL `serverUrl` befindet.

Wenn Ihre Argo-CD-Server-URL beispielsweise lautet`https://abc123.eks-capabilities.us-west-2.amazonaws.com`, lautet die Webhook-URL:

```
https://abc123.eks-capabilities.us-west-2.amazonaws.com/api/webhook
```

Webhooks nach Git-Anbieter konfigurieren

GitHub: Fügen Sie in Ihren Repository-Einstellungen einen Webhook mit der Argo CD-Webhook-URL hinzu. Stellen Sie den Inhaltstyp auf ein `application/json` und wählen Sie „Nur das Push-Ereignis“.

GitLab: Fügen Sie in Ihren Projekteinstellungen einen Webhook mit der Argo CD-Webhook-URL hinzu. Aktivieren Sie „Push-Ereignisse“ und optional „Tag-Push-Ereignisse“.

Bitbucket: Fügen Sie in Ihren Repository-Einstellungen einen Webhook mit der Argo CD-Webhook-URL hinzu. Wählen Sie „Repository-Push“ als Auslöser aus.

CodeCommit: Erstellen Sie eine EventBridge Amazon-Regel, die bei Änderungen des CodeCommit Repository-Status ausgelöst wird und Benachrichtigungen an den Argo CD-Webhook-Endpunkt sendet.

Eine ausführliche Anleitung zur Webhook-Konfiguration finden Sie unter Argo CD Webhook-Konfiguration.

> **Note**
>
> Webhooks ergänzen das Polling — sie ersetzen es nicht. Argo CD fragt weiterhin Repositorys als Ausweichmechanismus für den Fall ab, dass Webhook-Benachrichtigungen übersehen werden.

Nächste Schritte

- the section called “Arbeiten mit Argo CD”- Erfahren Sie, wie Sie Argo CD-Anwendungen erstellen und verwalten
- the section called “Fehlerbehebung”- Beheben Sie Probleme mit Argo CD
- the section called “Mit Fähigkeiten arbeiten”- Verwalten Sie Ihre Argo-CD-Funktionsressourcen

Probleme mit den Funktionen von Argo CD beheben

Dieses Thema enthält Anleitungen zur Fehlerbehebung für die EKS-Capability for Argo-CD, einschließlich Funktionsprüfungen, Problemen mit der Anwendungssynchronisierung, Repository-Authentifizierung und Bereitstellungen in mehreren Clustern.

> **Note**
>
> Die EKS-Funktionen werden vollständig verwaltet und außerhalb Ihres Clusters ausgeführt. Sie haben keinen Zugriff auf die Argo CD-Serverprotokolle oder den `argocd` Namespace. Die Fehlerbehebung konzentriert sich auf den Zustand der Funktionen, den Anwendungsstatus und die Konfiguration.

Die Funktion ist AKTIV, aber die Anwendungen werden nicht synchronisiert

Wenn Ihre Argo-CD-Funktion `ACTIVE` den Status anzeigt, Anwendungen aber nicht synchronisiert werden, überprüfen Sie den Funktionsstatus und den Anwendungsstatus.

Überprüfen Sie den Zustand der Funktionen:

Sie können Funktionszustands- und Statusprobleme in der EKS-Konsole oder über die AWS CLI anzeigen.

Konsole:

1. Öffnen Sie die Amazon EKS-Konsole unter https://console.aws.amazon.com/eks/ home#/clusters.
2. Wählen Sie Ihren Clusternamen aus.
3. Wählen Sie den Registerkarte Beobachtbarkeit.
4. Wählen Sie Cluster überwachen aus.
5. Wählen Sie die Registerkarte Funktionen, um den Zustand und den Status aller Funktionen anzuzeigen.

AWS CLI:

```
# View capability status and health
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
```

```
  --capability-name my-argocd

# Look for issues in the health section
```

Häufige Ursachen:

- Repository nicht konfiguriert: Git-Repository wurde nicht zur Argo-CD hinzugefügt
- Authentifizierung fehlgeschlagen: SSH-Schlüssel, Token oder CodeCommit Anmeldeinformationen sind ungültig
- Anwendung nicht erstellt: Im Cluster sind keine Anwendungsressourcen vorhanden
- Synchronisierungsrichtlinie: Manuelle Synchronisierung erforderlich (automatische Synchronisierung nicht aktiviert)
- IAM-Berechtigungen: Fehlende Berechtigungen für CodeCommit oder Secrets Manager

Überprüfen Sie den Status der Bewerbung:

```
# List applications
kubectl get application -n argocd

# View sync status
kubectl get application my-app -n argocd -o jsonpath='{.status.sync.status}'

# View application health
kubectl get application my-app -n argocd -o jsonpath='{.status.health}'
```

Überprüfen Sie die Bewerbungsbedingungen:

```
# Describe application to see detailed status
kubectl describe application my-app -n argocd

# View application health
kubectl get application my-app -n argocd -o jsonpath='{.status.health}'
```

Anwendungen bleiben im Status „In Bearbeitung“ hängen

Wenn eine Anwendung angezeigt wird, `Progressing` aber nie erreicht wird`Healthy`, überprüfen Sie den Ressourcenstatus und die Ereignisse der Anwendung.

Überprüfen Sie den Zustand der Ressourcen:

```
# View application resources
kubectl get application my-app -n argocd -o jsonpath='{.status.resources}'

# Check for unhealthy resources
kubectl describe application my-app -n argocd | grep -A 10 "Health Status"
```

Häufige Ursachen:

- Bereitstellung nicht bereit: Pods können nicht gestartet werden oder Bereitschaftstests schlagen fehl
- Ressourcenabhängigkeiten: Ressourcen, die darauf warten, dass andere Ressourcen bereit sind
- Fehler beim Abrufen von Bildern: Auf Container-Images kann nicht zugegriffen werden
- Unzureichende Ressourcen: Dem Cluster fehlt es an CPU oder Arbeitsspeicher für Pods

Überprüfen Sie die Zielclusterkonfiguration (für Multi-Cluster-Setups):

```
# List registered clusters
kubectl get secret -n argocd -l argocd.argoproj.io/secret-type=cluster

# View cluster secret details
kubectl get secret cluster-secret-name -n argocd -o yaml
```

Fehler beim Repository-Authentifizierung

Wenn Argo CD nicht auf Ihre Git-Repositorys zugreifen kann, überprüfen Sie die Authentifizierungskonfiguration.

Für CodeCommit Repositorys:

Stellen Sie sicher, dass die IAM-Capability-Rolle über Berechtigungen verfügt CodeCommit :

```
# View IAM policies
aws iam list-attached-role-policies --role-name my-argocd-capability-role
aws iam list-role-policies --role-name my-argocd-capability-role

# Get specific policy details
aws iam get-role-policy --role-name my-argocd-capability-role --policy-name policy-name
```

Die Rolle benötigt eine `codecommit:GitPull` Genehmigung für die Repositorys.

Für private Git-Repositorys:

Stellen Sie sicher, dass die Repository-Anmeldeinformationen korrekt konfiguriert sind:

```
# Check repository secret exists
kubectl get secret -n argocd repo-secret-name -o yaml
```

Stellen Sie sicher, dass das Geheimnis die richtigen Authentifizierungsdaten (SSH-Schlüssel, Token oder Benutzername/Passwort) enthält.

Für Repositorien, die Secrets Manager verwenden:

```
# Verify IAM Capability Role has Secrets Manager permissions
aws iam list-attached-role-policies --role-name my-argocd-capability-role

# Test secret retrieval
aws secretsmanager get-secret-value --secret-id arn:aws:secretsmanager:region-
code:111122223333:secret:my-secret
```

Probleme bei der Bereitstellung mehrerer Cluster

Wenn Anwendungen nicht auf Remote-Clustern bereitgestellt werden, überprüfen Sie die Clusterregistrierung und die Zugriffskonfiguration.

Überprüfen Sie die Clusterregistrierung:

```
# List registered clusters
kubectl get secret -n argocd -l argocd.argoproj.io/secret-type=cluster

# Verify cluster secret format
kubectl get secret CLUSTER_SECRET_NAME -n argocd -o yaml
```

Stellen Sie sicher, dass das `server` Feld den EKS-Cluster-ARN und nicht die Kubernetes-API-URL enthält.

Überprüfen Sie den Zugriffseintrag für den Zielcluster:

Überprüfen Sie auf dem Zielcluster, ob die Argo-CD-Capability-Rolle über einen Zugriffseintrag verfügt:

```
# List access entries (run on target cluster or use AWS CLI)
aws eks list-access-entries --cluster-name target-cluster

# Describe specific access entry
aws eks describe-access-entry \
  --cluster-name target-cluster \
  --principal-arn arn:aws:iam::[.replaceable]111122223333:role/my-argocd-capability-
role
```

Überprüfen Sie die IAM-Berechtigungen für kontoübergreifend:

Stellen Sie bei kontoübergreifenden Bereitstellungen sicher, dass die Argo-CD-Capability-Rolle über einen Zugriffseintrag auf dem Zielcluster verfügt. Die verwaltete Funktion verwendet EKS-Zugriffseinträge für den kontoübergreifenden Zugriff und nicht für die Übernahme von IAM-Rollen.

Weitere Informationen zur Konfiguration mehrerer Cluster finden Sie unter. the section called “Registrieren Sie Cluster”

Nächste Schritte

- the section called “Überlegungen zu Argo CD”- Überlegungen und bewährte Verfahren für Argo CD
- the section called “Arbeiten mit Argo CD”- Erstellen und verwalten Sie Argo CD-Anwendungen
- the section called “Registrieren Sie Cluster”- Konfigurieren Sie Bereitstellungen mit mehreren Clustern
- the section called “Problembehebung bei Funktionen”- Allgemeine Hinweise zur Problembehebung

Vergleich der EKS-Funktionen für Argo CD mit selbstverwalteter Argo-CD

Die EKS-Funktion für Argo CD bietet ein vollständig verwaltetes Argo-CD-Erlebnis, das in EKS ausgeführt wird. Einen allgemeinen Vergleich zwischen EKS-Funktionen und selbstverwalteten Lösungen finden Sie unter. the section called “Überlegungen” Dieses Thema konzentriert sich auf die spezifischen Unterschiede zwischen Argo CD, einschließlich Authentifizierung, Multi-Cluster-Management und Unterstützung von Upstream-Funktionen.

Unterschiede zur Upstream-Version von Argo CD

Die EKS-Funktion für Argo CD basiert auf der Upstream-Argo-CD, unterscheidet sich jedoch darin, wie auf sie zugegriffen wird, wie sie konfiguriert und in Dienste integriert wird. AWS

RBAC und Authentifizierung: Die Funktion umfasst drei RBAC-Rollen (Admin, Editor, Viewer) und verwendet AWS Identity Center für die Authentifizierung anstelle der integrierten Authentifizierung von Argo CD. Konfigurieren Sie Rollenzuordnungen über den `rbacRoleMapping` Parameter der Funktion, um Identity Center-Gruppen Argo-CD-Rollen zuzuordnen, nicht über Argo-CDs. `argocd-rbac-cm` ConfigMap Die Benutzeroberfläche von Argo CD wird mit einer eigenen direkten URL gehostet (diese finden Sie in der EKS-Konsole unter dem Tab Capabilities Ihres Clusters), und der API-Zugriff erfolgt über AWS Authentifizierung und Autorisierung über IAM.

Cluster-Konfiguration: Die Funktion konfiguriert nicht automatisch lokale Cluster oder hub-and-spoke Topologien. Sie konfigurieren Ihre Bereitstellungszielcluster und EKS-Zugriffseinträge. Die Funktion unterstützt nur Amazon EKS-Cluster als Bereitstellungsziele, die den EKS-Cluster ARNs (nicht den Kubernetes-API-Server URLs) verwenden. Die Funktion fügt den lokalen Cluster (`kubernetes.default.svc`) nicht automatisch als Bereitstellungsziel hinzu. Um die Bereitstellung auf demselben Cluster durchzuführen, auf dem die Funktion erstellt wurde, registrieren Sie diesen Cluster explizit mit seinem ARN.

Vereinfachter Remote-Cluster-Zugriff: Die Funktion vereinfacht Bereitstellungen auf mehreren Clustern, indem mithilfe von EKS-Zugriffseinträgen Argo-CD-Zugriff auf Remote-Cluster gewährt wird. Dadurch entfällt die Notwendigkeit, IAM-Rollen für Dienstkonten (IRSA) zu konfigurieren oder kontenübergreifende IAM-Rollenannahmen einzurichten. Die Funktion bietet auch transparenten Zugriff auf vollständig private EKS-Cluster, ohne dass VPC-Peering oder eine spezielle Netzwerkkonfiguration erforderlich sind. Die Konnektivität zwischen der Argo-CD-Funktion und privaten Remote-Clustern wird automatisch AWS verwaltet.

Direkte AWS Serviceintegration: Die Funktion ermöglicht die direkte Integration mit AWS Diensten über die IAM-Berechtigungen der Capability Role. Sie können CodeCommit Repositorys, ECR-Helm-Diagramme und CodeConnections direkt in Anwendungsressourcen referenzieren, ohne Repository-Konfigurationen erstellen zu müssen. Dies vereinfacht die Authentifizierung und macht die Verwaltung separater Anmeldeinformationen für AWS Dienste überflüssig. Details dazu finden Sie unter the section called “Repositorys konfigurieren”.

Namespace-Unterstützung: Für diese Funktion müssen Sie einen einzelnen Namespace angeben, ApplicationSet in dem Argo CD Application und AppProject benutzerdefinierte Ressourcen erstellt werden müssen.

> **Note**
>
> Diese Namespace-Beschränkung gilt nur für die eigenen benutzerdefinierten Ressourcen von Argo CD (Application,,). ApplicationSet AppProject Ihre Anwendungs-Workloads können

in jedem Namespace in jedem Zielcluster bereitgestellt werden. Wenn Sie die Funktion beispielsweise mit Namespace erstellen`argocd`, CRs müssen alle Anwendungen im Namespace erstellt werden, aber diese Anwendungen können Workloads in,, oder jedem anderen `argocd` Namespace bereitstellen. `default` `production` `staging`

> **Note**
>
> Die verwaltete Funktion hat spezifische Anforderungen an die Verwendung und AppProject Konfiguration der CLI:
>
> - Wenn Sie die Argo-CD-CLI verwenden, geben Sie Anwendungen mit dem Namespace-Präfix an: `argocd app sync namespace/appname`
> - AppProject Ressourcen müssen angeben`.spec.sourceNamespaces`, um zu definieren, welche Namespaces das Projekt nach Anwendungen durchsuchen kann (normalerweise auf den Namespace gesetzt, den Sie bei der Erstellung der Funktion angegeben haben)
> - Anmerkungen zur Ressourcenverfolgung verwenden das folgende Format: `namespace_appname:group/kind:namespace/name`

Nicht unterstützte Funktionen: Die folgenden Funktionen sind in der verwalteten Funktion nicht verfügbar:

- Config Management Plugins (CMPs) für die Generierung benutzerdefinierter Manifeste
- Benutzerdefinierte Lua-Skripte für die Bewertung des Ressourcenzustands (integrierte Zustandsprüfungen für Standardressourcen werden unterstützt)
- Der Benachrichtigungs-Controller
- Benutzerdefinierte SSO-Anbieter (nur AWS Identity Center wird unterstützt, einschließlich föderierter Identitäten von Drittanbietern über AWS Identity Center)
- Benutzeroberflächenerweiterungen und benutzerdefinierte Banner
- Direkter Zugriff auf `argocd-cm`argocd-params, und andere Konfigurationen ConfigMaps
- Änderung des Synchronisierungs-Timeouts (fest auf 120 Sekunden festgelegt)

Kompatibilität: Anwendungen und E ApplicationSets funktionieren genauso wie die Upstream-Argo-CD, ohne dass Ihre Manifeste geändert werden. Die Funktion verwendet dasselbe Kubernetes

APIs und CRDs daher `kubectl` funktionieren Tools wie auf die gleiche Weise. Die Funktion unterstützt vollständig Anwendungen und GitOps Workflows mit automatischer Synchronisierung ApplicationSets, Multi-Cluster-Bereitstellungen, Synchronisierungsrichtlinien (automatisiert, bereinigt, selbstheilend), Synchronisierungswellen und Hooks, Integritätsbeurteilungen für Standard-Kubernetes-Ressourcen, Rollback-Funktionen, Git-Repository-Quellen (HTTPS und SSH), Helm-, Kustomize- und einfache YAML-Manifeste, GitHub App-Anmeldeinformationen, Projekte für Mehrmandantenfähigkeit sowie Ressourcenausschlüsse und -einschlüsse.

Verwenden der Argo-CD-CLI mit der verwalteten Funktion

Die Argo-CD-CLI funktioniert für die meisten Operationen genauso wie die Upstream-Argo-CD, aber Authentifizierung und Clusterregistrierung unterscheiden sich.

Voraussetzungen

Installieren Sie die Argo-CD-CLI gemäß den Upstream-Installationsanweisungen.

Konfiguration

Konfigurieren Sie die CLI mithilfe von Umgebungsvariablen:

1. Rufen Sie die URL des Argo-CD-Servers von der EKS-Konsole (auf der Registerkarte Capabilities Ihres Clusters) oder mithilfe der AWS CLI ab. Das `https://` Präfix muss entfernt werden:

```
export ARGOCD_SERVER=$(aws eks describe-capability \
  --cluster-name my-cluster \
  --capability-name my-argocd \
  --query 'capability.configuration.argoCd.serverUrl' \
  --output text \
  --region region-code | sed 's|^https://||')
```

2. Generieren Sie ein Konto-Token über die Benutzeroberfläche von Argo CD (Einstellungen → Konten → admin → Neues Token generieren) und legen Sie es dann als Umgebungsvariable fest:

```
export ARGOCD_AUTH_TOKEN="your-token-here"
```

> **Important**
>
> Diese Konfiguration verwendet das Admin-Kontotoken für die Ersteinrichtung und die Entwicklungsabläufe. Verwenden Sie für Anwendungsfälle in der Produktion projektbezogene

Rollen und Token, um dem Prinzip der geringsten Rechte zu folgen. Weitere Informationen zur Konfiguration von Projektrollen und RBAC finden Sie unter. the section called "Konfigurieren von -Berechtigungen"

1. Stellen Sie die erforderliche gRPC-Option ein:

```
export ARGOCD_OPTS="--grpc-web"
```

Wenn diese Umgebungsvariablen gesetzt sind, können Sie die Argo CD CLI ohne den `argocd login` Befehl verwenden.

Die wichtigsten Unterschiede:

Für die verwaltete Funktion gelten die folgenden CLI-Einschränkungen:

- `argocd admin`Befehle werden nicht unterstützt (sie erfordern direkten Pod-Zugriff)
- `argocd login`wird nicht unterstützt (verwenden Sie stattdessen Konto- oder Projekt-Token)
- `argocd cluster add`benötigt das `--aws-cluster-name` Flag mit dem EKS-Cluster-ARN

Beispiel: Registrieren Sie einen Cluster

Registrieren Sie einen EKS-Cluster für die Anwendungsbereitstellung:

```
# Get the cluster ARN
CLUSTER_ARN=$(aws eks describe-cluster \
  --name my-cluster \
  --query 'cluster.arn' \
  --output text)

# Register the cluster
argocd cluster add $CLUSTER_ARN \
  --aws-cluster-name $CLUSTER_ARN \
  --name in-cluster \
  --project default
```

Die vollständige Argo CD CLI-Dokumentation finden Sie in der Argo CD CLI-Referenz.

Migrationspfad

Sie können von der selbstverwalteten Argo-CD zur verwalteten Funktion migrieren:

1. Überprüfen Sie Ihre aktuelle Argo-CD-Konfiguration auf nicht unterstützte Funktionen (Benachrichtigungs-Controller, benutzerdefinierte Integritätsprüfungen CMPs, Benutzeroberflächenerweiterungen)
2. Skalieren Sie Ihre selbstverwalteten Argo-CD-Controller auf null Replikate, um Konflikte zu vermeiden
3. Erstellen Sie eine Argo-CD-Funktionsressource auf Ihrem Cluster
4. Exportieren Sie Ihre vorhandenen Anwendungen, ApplicationSets, und AppProjects
5. Migrieren Sie Repository-Anmeldeinformationen, Cluster-Geheimnisse und Vorlagen für Repository-Anmeldeinformationen (Repocreds)
6. Wenn Sie GPG-Schlüssel, TLS-Zertifikate oder bekannte SSH-Hosts verwenden, migrieren Sie auch diese Konfigurationen
7. Aktualisieren Sie die `destination.server` Felder, um Clusternamen oder EKS-Cluster zu verwenden ARNs
8. Wenden Sie sie auf die verwaltete Argo-CD-Instanz an
9. Stellen Sie sicher, dass die Anwendungen korrekt synchronisiert werden
10. Nehmen Sie Ihre selbstverwaltete Argo-CD-Installation außer Betrieb

Die verwaltete Funktion verwendet dieselben Argo-CD APIs - und Ressourcendefinitionen, sodass Ihre vorhandenen Manifeste mit minimalen Änderungen funktionieren.

Nächste Schritte

- the section called “Erstellen der Argo CD-Funktion”- Erstellen Sie eine Argo-CD-Funktionsressource
- the section called “Arbeiten mit Argo CD”- Stellen Sie Ihre erste Anwendung bereit
- the section called “Überlegungen zu Argo CD”- Konfigurieren Sie die AWS Identity Center-Integration

Zusammenstellung von Ressourcen mit kro (Kube Resource Orchestrator)

kro (Kube Resource Orchestrator) ist ein Kubernetes-natives Open-Source-Projekt, mit dem Sie benutzerdefinierte Kubernetes mithilfe einer einfachen und unkomplizierten Konfiguration definieren können. APIs Mit kro können Sie ganz einfach neue benutzerdefinierte Objekte konfigurieren, die eine Gruppe von Kubernetes-Objekten und APIs die logischen Operationen zwischen ihnen erstellen.

Mit EKS Capabilities wird kro vollständig von verwaltet AWS, sodass Sie Kro-Controller nicht mehr auf Ihren Clustern installieren, warten und skalieren müssen.

Wie funktioniert Kro

kro führt eine benutzerdefinierte Ressourcendefinition (CRD) namens `ResourceGraphDefinition` (RGD) ein, die die einfache und optimierte Erstellung von benutzerdefinierten Kubernetes ermöglicht. APIs Wenn Sie eine erstellen, verwendet Kro native Kubernetes-Erweiterungen`ResourceGraphDefinition`, um neue in Ihrem Cluster zu erstellen und zu verwalten. APIs Auf der Grundlage dieser einzelnen Ressourcenspezifikation erstellt und registriert kro für Sie eine neue CRD auf der Grundlage Ihrer Spezifikation und passt sich an die Verwaltung Ihrer neu definierten benutzerdefinierten Ressourcen an.

RGDs kann mehrere Ressourcen beinhalten, und Kro bestimmt die gegenseitigen Abhängigkeiten und die Reihenfolge der Ressourcen, sodass Sie sich nicht darum kümmern müssen. Sie können eine einfache Syntax verwenden, um Konfigurationen von einer Ressource in eine andere zu übertragen, wodurch Kompositionen erheblich vereinfacht werden und die Notwendigkeit von „Glue“ -Operatoren in Ihrem Cluster entfällt. Mit kro können Ihre benutzerdefinierten Ressourcen sowohl native Kubernetes-Ressourcen als auch alle im Cluster installierten benutzerdefinierten Ressourcendefinitionen (CRDs) enthalten.

kro unterstützt einen einzigen primären Ressourcentyp:

- ResourceGraphDefinition (RGD): Definiert eine benutzerdefinierte Kubernetes-Ressource, die eine oder mehrere zugrunde liegende native oder benutzerdefinierte Kubernetes-Ressourcen kapselt

Zusätzlich zu dieser Ressource erstellt und verwaltet kro den Lebenszyklus Ihrer damit erstellten benutzerdefinierten Ressourcen sowie aller zugehörigen Ressourcen.

kro lässt sich nahtlos in AWS Controllers for Kubernetes (ACK) integrieren, sodass Sie Workload-Ressourcen mit AWS Ressourcen zusammenstellen können, um Abstraktionen auf höherer Ebene zu erstellen. Auf diese Weise können Sie Ihre eigenen Cloud-Bausteine erstellen, das Ressourcenmanagement vereinfachen und wiederverwendbare Muster mit standardmäßigen und unveränderlichen Konfigurationseinstellungen ermöglichen, die auf Ihren Unternehmensstandards basieren.

Vorteile von Kro

kro ermöglicht es Plattformteams, benutzerdefinierte Kubernetes zu erstellen, APIs die mehrere Ressourcen zu Abstraktionen auf höherer Ebene zusammensetzen. Dies vereinfacht das Ressourcenmanagement, da Entwickler komplexe Anwendungen mithilfe einfacher, standardisierter und versionierter benutzerdefinierter Ressourcen bereitstellen können. Sie definieren wiederverwendbare Muster für gängige Ressourcenkombinationen und ermöglichen so eine konsistente Ressourcenerstellung in Ihrer gesamten Organisation.

kro verwendet Common Expression Language (CEL) in Kubernetes, um Werte zwischen Ressourcen zu übergeben und bedingte Logik zu integrieren, was Flexibilität bei der Ressourcenzusammensetzung bietet. Sie können sowohl Kubernetes-Ressourcen als auch von ACK verwaltete Ressourcen zu einheitlichen benutzerdefinierten AWS Ressourcen zusammenfassen APIs, sodass vollständige Anwendungs- und Infrastrukturdefinitionen möglich sind.

kro unterstützt die deklarative Konfiguration über Kubernetes-Manifeste und ermöglicht so GitOps Workflows und Infrastruktur als Code-Praktiken, die sich nahtlos in Ihre bestehenden Entwicklungsprozesse integrieren lassen. Als Teil von EKS Managed Capabilities wird kro vollständig von verwaltet AWS, sodass Sie keine Kro-Controller auf Ihren Clustern installieren, konfigurieren und warten müssen.

Beispiel: Erstellen eines ResourceGraphDefinition

Das folgende Beispiel zeigt ein einfaches Beispiel`ResourceGraphDefinition`, das eine Webanwendung mit einem Deployment und einem Service erstellt:

```
apiVersion: kro.run/v1alpha1
kind: ResourceGraphDefinition
metadata:
  name: web-application
spec:
  schema:
    apiVersion: v1alpha1
```

```
    kind: WebApplication
    spec:
      name: string
      replicas: integer | default=3
  resources:
    - id: deployment
      template:
        apiVersion: apps/v1
        kind: Deployment
        metadata:
          name: ${schema.spec.name}
        spec:
          replicas: ${schema.spec.replicas}
    - id: service
      template:
        apiVersion: v1
        kind: Service
        metadata:
          name: ${schema.spec.name}
```

Wenn Benutzer Instanzen der `WebApplication` benutzerdefinierten Ressource erstellen, erstellt Kro automatisch die entsprechenden Bereitstellungs- und Dienstressourcen und verwaltet deren Lebenszyklus zusammen mit der benutzerdefinierten Ressource.

Integration mit anderen von EKS verwalteten Funktionen

kro lässt sich in andere von EKS verwaltete Funktionen integrieren.

- AWS Controller für Kubernetes (ACK): Verwenden Sie Kro, um ACK-Ressourcen in Abstraktionen auf höherer Ebene zusammenzustellen und so das Ressourcenmanagement zu vereinfachen. AWS
- Argo CD: Verwenden Sie Argo CD, um die Bereitstellung von benutzerdefinierten Kro-Ressourcen in mehreren Clustern zu verwalten und so Workflows für Ihre Plattformbausteine und Anwendungsstapel zu ermöglichen GitOps .

Erste Schritte mit Kro

Um mit der EKS-Funktion für kro zu beginnen:

1. Erstellen Sie eine Kro-Capability-Ressource auf Ihrem EKS-Cluster über die AWS Konsole, AWS CLI oder Ihr bevorzugtes Infrastructure-as-Code-Tool.

2. Create ResourceGraphDefinitions (RGDs), die Ihre benutzerdefinierte Zusammensetzung APIs und Ihre Ressourcenzusammensetzung definieren.
3. Wenden Sie Instanzen Ihrer benutzerdefinierten Ressourcen an, um die zugrunde liegenden Kubernetes und Ressourcen bereitzustellen und AWS zu verwalten.

Eine Kro-Fähigkeit erstellen

In diesem Thema wird erklärt, wie Sie eine Kro-Funktion auf Ihrem Amazon EKS-Cluster erstellen.

Voraussetzungen

Bevor Sie eine Kro-Fähigkeit erstellen, stellen Sie sicher, dass Sie über Folgendes verfügen:

- Ein vorhandener Amazon EKS-Cluster, auf dem eine unterstützte Kubernetes-Version ausgeführt wird (alle Versionen mit Standard- und erweitertem Support werden unterstützt)
- Ausreichende IAM-Berechtigungen zum Erstellen von Funktionsressourcen auf EKS-Clustern
- (Für CLI/eksCTL) Das entsprechende CLI-Tool wurde installiert und konfiguriert

> **Note**
>
> Im Gegensatz zu ACK und Argo CD benötigt kro keine zusätzlichen IAM-Berechtigungen, die über die Vertrauensrichtlinie hinausgehen. Kro arbeitet ausschließlich innerhalb Ihres Clusters und führt keine API-Aufrufe durch. AWS Sie müssen jedoch weiterhin eine IAM-Fähigkeitsrolle mit der entsprechenden Vertrauensrichtlinie bereitstellen. Informationen zur Konfiguration von Kubernetes-RBAC-Berechtigungen für Kro finden Sie unter. the section called “Konfigurieren von -Berechtigungen”

Wählen Sie Ihr Tool

Sie können eine Kro-Fähigkeit mit der AWS-Managementkonsole AWS CLI oder eksctl erstellen:

- the section called “Konsole”- Verwenden Sie die Konsole für ein geführtes Erlebnis
- the section called “ AWS CLI”- Verwenden Sie die AWS CLI für Scripting und Automatisierung
- the section called “eksctl”- Verwenden Sie eksctl für ein Kubernetes-natives Erlebnis

Was passiert, wenn Sie eine Kro-Fähigkeit erstellen

Wenn Sie eine Kro-Fähigkeit erstellen:

1. EKS erstellt den Kro-Capability Service und konfiguriert ihn für die Überwachung und Verwaltung der Ressourcen in Ihrem Cluster
2. Benutzerdefinierte Ressourcendefinitionen (CRDs) sind in Ihrem Cluster installiert
3. Die Funktion setzt die von Ihnen bereitgestellte IAM-Fähigkeitsrolle voraus (wird nur für die Vertrauensbeziehung verwendet)
4. kro beginnt, nach `ResourceGraphDefinition` Ressourcen und ihren Instanzen Ausschau zu halten
5. Der Status der Fähigkeit ändert sich von zu `CREATING` `ACTIVE`

Sobald diese Option aktiviert ist, können Sie benutzerdefinierte Objekte erstellen, definieren APIs und Instanzen davon erstellen APIs. ResourceGraphDefinitions

Nächste Schritte

Nach der Erstellung der Kro-Fähigkeit:

- the section called “Kro-Konzepte”- Verstehen Sie die KRO-Konzepte und die Zusammensetzung der Ressourcen
- the section called “Kro-Konzepte”- Erfahren Sie mehr über SimpleSchema CEL-Ausdrücke und Muster der Ressourcenzusammensetzung

Erstellen Sie mit der Konsole eine Kro-Fähigkeit

In diesem Thema wird beschrieben, wie Sie eine Kro-Fähigkeit (Kube Resource Orchestrator) mithilfe von erstellen. AWS-Managementkonsole

Erstellen Sie die Kro-Fähigkeit

1. Öffnen Sie die Amazon EKS-Konsole unter https://console.aws.amazon.com/eks/ home#/clusters.
2. Wählen Sie Ihren Clusternamen aus, um die Cluster-Detailseite zu öffnen.
3. Wählen Sie die Registerkarte Funktionen.
4. Wählen Sie in der linken Navigationsleiste kro (Kube Resource Orchestrator) aus.
5. Wählen Sie Kro-Fähigkeit erstellen aus.

6. Für die Rolle „IAM-Fähigkeit“:
   - Wenn Sie bereits über eine IAM-Fähigkeitsrolle verfügen, wählen Sie sie aus der Dropdownliste aus
   - Wenn Sie eine Rolle erstellen müssen, wählen Sie Create kro role

     Dadurch wird die IAM-Konsole auf einer neuen Registerkarte mit einer vorab ausgefüllten Vertrauensrichtlinie geöffnet. Für die Rolle sind keine zusätzlichen IAM-Berechtigungen erforderlich, da Kro vollständig in Ihrem Cluster ausgeführt wird.

     Kehren Sie nach dem Erstellen der Rolle zur EKS-Konsole zurück. Die Rolle wird automatisch ausgewählt.

     > **Note**
     >
     > Im Gegensatz zu ACK und Argo CD benötigt kro keine zusätzlichen IAM-Berechtigungen, die über die Vertrauensrichtlinie hinausgehen. Kro arbeitet ausschließlich innerhalb Ihres Clusters und führt keine API-Aufrufe durch. AWS
7. Wählen Sie Erstellen aus.

Der Prozess zur Schaffung von Fähigkeiten beginnt.

Stellen Sie sicher, dass die Fähigkeit aktiv ist

1. Sehen Sie sich auf der Registerkarte Funktionen den Status der Kro-Fähigkeit an.
2. Warten Sie, bis sich der Status von `CREATING` zu `ACTIVE` ändert.
3. Sobald die Funktion aktiv ist, ist sie einsatzbereit.

Informationen zum Status der Funktionen und zur Problembehandlung finden Sie unterthe section called “Mit Fähigkeiten arbeiten”.

Erteilen Sie Berechtigungen zur Verwaltung von Kubernetes-Ressourcen

Wenn Sie eine Kro-Funktion erstellen, wird automatisch ein EKS-Zugriffseintrag mit dem erstelltAmazonEKSKROPolicy, sodass Kro und ihre Instanzen verwalten ResourceGraphDefinitions können. Es werden jedoch standardmäßig keine Berechtigungen zum Erstellen der zugrunde liegenden Kubernetes-Ressourcen (wie Bereitstellungen, Dienste usw.) gewährt, die in ConfigMaps Ihrem definiert sind. ResourceGraphDefinitions

Dieses vorsätzliche Design folgt dem Prinzip der geringsten Rechte ResourceGraphDefinitions — unterschiedliche Rechte erfordern unterschiedliche Berechtigungen. Sie müssen die Berechtigungen, die kro benötigt, explizit auf der Grundlage der Ressourcen konfigurieren, die Sie verwalten werden. ResourceGraphDefinitions

Verwenden `AmazonEKSClusterAdminPolicy` Sie für einen schnellen Einstieg in Test- oder Entwicklungsumgebungen:

1. Navigieren Sie in der EKS-Konsole zur Registerkarte Access Ihres Clusters.
2. Suchen Sie unter Access-Einträge den Eintrag für Ihre Kro-Capability-Rolle (sie wird den Rollen-ARN haben, den Sie zuvor erstellt haben).
3. Wählen Sie den Access-Eintrag aus, um die zugehörigen Details zu öffnen.
4. Wählen Sie im Abschnitt Zugriffsrichtlinien die Option Zugriffsrichtlinie zuordnen aus.
5. Wählen Sie `AmazonEKSClusterAdminPolicy` aus der Richtlinienliste aus.
6. Wählen Sie als Zugriffsbereich die Option Cluster aus.
7. Wählen Sie Associate aus.

> **Important**
>
> Das `AmazonEKSClusterAdminPolicy` gewährt umfassende Berechtigungen zum Erstellen und Verwalten aller Kubernetes-Ressourcen, einschließlich der Möglichkeit, beliebige Ressourcentypen in allen Namespaces zu erstellen. Dies ist praktisch für die Entwicklung, sollte POCs aber nicht in der Produktion verwendet werden. Erstellen Sie für die Produktion benutzerdefinierte RBAC-Richtlinien, die nur die Berechtigungen gewähren, die für die spezifischen Ressourcen erforderlich sind, die Sie verwalten ResourceGraphDefinitions werden. Anleitungen zur Konfiguration von Berechtigungen mit den geringsten Rechten finden Sie unter und. the section called “Konfigurieren von -Berechtigungen” the section called “Überlegungen zu den Funktionen von EKS”

Stellen Sie sicher, dass benutzerdefinierte Ressourcen verfügbar sind

Nachdem die Funktion aktiviert wurde, stellen Sie sicher, dass benutzerdefinierte KRO-Ressourcen in Ihrem Cluster verfügbar sind.

Verwenden der Konsole

1. Navigieren Sie in der Amazon EKS-Konsole zu Ihrem Cluster
2. Wählen Sie die Registerkarte Ressourcen
3. Wählen Sie Erweiterungen
4. Wählen Sie CustomResourceDefinitions aus.

Der `ResourceGraphDefinition` Ressourcentyp sollte aufgeführt sein.

Verwenden Sie kubectl

```
kubectl api-resources | grep kro.run
```

Der `ResourceGraphDefinition` Ressourcentyp sollte aufgeführt sein.

Nächste Schritte

- the section called “Kro-Konzepte”- Verstehen Sie die KRO-Konzepte und die Zusammensetzung der Ressourcen
- the section called “Kro-Konzepte”- Erfahren Sie mehr über SimpleSchema CEL-Ausdrücke und Kompositionsmuster
- the section called “Mit Fähigkeiten arbeiten”- Verwalte deine Ressourcen für Kro-Fähigkeiten

Erstellen Sie eine Kro-Fähigkeit mit der CLI AWS

In diesem Thema wird beschrieben, wie Sie mit der CLI eine KRO-Funktion (Kube Resource Orchestrator) erstellen. AWS

Voraussetzungen

- AWS CLI — Version `2.12.3` oder höher. Führen Sie den Befehl aus, um Ihre Version zu überprüfen`aws --version`. Weitere Informationen finden Sie im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle unter Installation.
- **`kubectl`** – Ein Befehlszeilentool für die Arbeit mit Kubernetes-Clustern. Weitere Informationen finden Sie unter the section called “kubectl und eksctl einrichten”.

Schritt 1: Erstellen Sie eine IAM-Capability-Rolle

Erstellen Sie eine Vertrauensrichtliniendatei:

```
cat > kro-trust-policy.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "capabilities.eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
EOF
```

Erstellen Sie die IAM-Rolle:

```
aws iam create-role \
  --role-name KROCapabilityRole \
  --assume-role-policy-document file://kro-trust-policy.json
```

> **Note**
>
> Im Gegensatz zu ACK und Argo CD benötigt kro keine zusätzlichen IAM-Berechtigungen. kro arbeitet vollständig innerhalb Ihres Clusters und führt keine API-Aufrufe durch. AWS Die Rolle wird nur benötigt, um die Vertrauensstellung mit dem EKS-Capabilities Service herzustellen.

Schritt 2: Erstellen Sie die Kro-Fähigkeit

Erstellen Sie die Kro-Capability-Ressource auf Ihrem Cluster. *region-code*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet (z. B.`us-west-2`), und *my-cluster* durch Ihren Clusternamen.

```
aws eks create-capability \
  --region region-code \
  --cluster-name my-cluster \
```

```
  --capability-name my-kro \
  --type KRO \
  --role-arn arn:aws:iam::$(aws sts get-caller-identity --query Account --output
 text):role/KROCapabilityRole \
  --delete-propagation-policy RETAIN
```

Der Befehl wird sofort zurückgegeben, aber es dauert einige Zeit, bis die Funktion aktiv wird, da EKS die erforderliche Fähigkeitsinfrastruktur und die erforderlichen Komponenten erstellt. EKS installiert die benutzerdefinierten Kubernetes-Ressourcendefinitionen, die sich auf diese Funktion beziehen, in Ihrem Cluster, während dieser erstellt wird.

> **Note**
>
> Wenn Sie eine Fehlermeldung erhalten, dass der Cluster nicht existiert oder Sie keine Berechtigungen haben, überprüfen Sie Folgendes:
>
> - Der Clustername ist korrekt
> - Ihre AWS CLI ist für die richtige Region konfiguriert
> - Sie verfügen über die erforderlichen IAM-Berechtigungen

Schritt 3: Stellen Sie sicher, dass die Funktion aktiv ist

Warten Sie, bis die Funktion aktiv wird. *region-code*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet, und *my-cluster* ersetzen Sie es durch den Namen Ihres Clusters.

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-kro \
  --query 'capability.status' \
  --output text
```

Die Funktion ist bereit, wenn der Status angezeigt wird`ACTIVE`.

Sie können sich auch die vollständigen Funktionsdetails ansehen:

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
```

```
  --capability-name my-kro
```

Schritt 4: Erteilen Sie Berechtigungen zur Verwaltung von Kubernetes-Ressourcen

Wenn Sie eine Kro-Funktion erstellen, wird automatisch ein EKS-Zugriffseintrag mit dem erstelltAmazonEKSKROPolicy, sodass Kro und ihre Instanzen verwalten ResourceGraphDefinitions können. Es werden jedoch standardmäßig keine Berechtigungen zum Erstellen der zugrunde liegenden Kubernetes-Ressourcen (wie Bereitstellungen, Dienste usw.) gewährt, die in ConfigMaps Ihrem definiert sind. ResourceGraphDefinitions

Dieses absichtliche Design folgt dem Prinzip der geringsten Rechte ResourceGraphDefinitions — unterschiedliche Rechte erfordern unterschiedliche Berechtigungen. Beispiel: * Ein System ResourceGraphDefinition , das nur ConfigMaps Secrets erstellt, benötigt andere Berechtigungen als ein System, das Deployments und Services erstellt. * Ein System ResourceGraphDefinition , das ACK-Ressourcen erstellt, benötigt Berechtigungen für diese spezifischen benutzerdefinierten Ressourcen. * Manche Benutzer lesen ResourceGraphDefinitions möglicherweise nur vorhandene Ressourcen, ohne neue zu erstellen

Sie müssen die Berechtigungen, die kro benötigt, explizit auf der Grundlage der Ressourcen konfigurieren, die Sie verwalten ResourceGraphDefinitions werden.

Quick Setup

Verwenden AmazonEKSClusterAdminPolicy Sie für einen schnellen Einstieg in Test- oder Entwicklungsumgebungen:

Rufen Sie den ARN der Funktionsrolle ab:

```
CAPABILITY_ROLE_ARN=$(aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-kro \
  --query 'capability.roleArn' \
  --output text)
```

Ordnen Sie die Cluster-Admin-Richtlinie zu:

```
aws eks associate-access-policy \
  --region region-code \
  --cluster-name my-cluster \
```

```
  --principal-arn $CAPABILITY_ROLE_ARN \
  --policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy \
  --access-scope type=cluster
```

> **Important**
>
> Das `AmazonEKSClusterAdminPolicy` gewährt umfassende Berechtigungen zum Erstellen und Verwalten aller Kubernetes-Ressourcen, einschließlich der Möglichkeit, beliebige Ressourcentypen in allen Namespaces zu erstellen. Dies ist praktisch für die Entwicklung, sollte POCs aber nicht in der Produktion verwendet werden. Erstellen Sie für die Produktion benutzerdefinierte RBAC-Richtlinien, die nur die Berechtigungen gewähren, die für die spezifischen Ressourcen erforderlich sind, die Sie verwalten ResourceGraphDefinitions werden. Anleitungen zur Konfiguration von Berechtigungen mit den geringsten Rechten finden Sie unter und. the section called “Konfigurieren von -Berechtigungen” the section called “Überlegungen zu den Funktionen von EKS”

Schritt 5: Stellen Sie sicher, dass benutzerdefinierte Ressourcen verfügbar sind

Nachdem die Funktion aktiviert ist, stellen Sie sicher, dass benutzerdefinierte KRO-Ressourcen in Ihrem Cluster verfügbar sind:

```
kubectl api-resources | grep kro.run
```

Der `ResourceGraphDefinition` Ressourcentyp sollte aufgeführt sein.

Nächste Schritte

- the section called “Kro-Konzepte”- Verstehen Sie die KRO-Konzepte und die Zusammensetzung der Ressourcen
- the section called “Kro-Konzepte”- Erfahren Sie mehr über SimpleSchema CEL-Ausdrücke und Kompositionsmuster
- the section called “Mit Fähigkeiten arbeiten”- Verwalte deine Ressourcen für Kro-Fähigkeiten

Erstellen Sie eine Kro-Fähigkeit mit eksctl

In diesem Thema wird beschrieben, wie Sie mit eksctl eine Kro-Fähigkeit (Kube Resource Orchestrator) erstellen.

> **Note**
>
> Für die folgenden Schritte ist die Version eksctl oder höher erforderlich. `0.220.0` Führen Sie den Befehl aus, um Ihre Version zu überprüfen. `eksctl version`

Schritt 1: Erstellen Sie eine IAM-Capability-Rolle

Erstellen Sie eine Vertrauensrichtliniendatei:

```
cat > kro-trust-policy.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "capabilities.eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
EOF
```

Erstellen Sie die IAM-Rolle:

```
aws iam create-role \
  --role-name KROCapabilityRole \
  --assume-role-policy-document file://kro-trust-policy.json
```

> **Note**
>
> Im Gegensatz zu ACK und Argo CD benötigt kro keine zusätzlichen IAM-Berechtigungen, die über die Vertrauensrichtlinie hinausgehen. Kro arbeitet ausschließlich innerhalb Ihres Clusters und führt keine API-Aufrufe durch. AWS

Schritt 2: Erstellen Sie die Kro-Fähigkeit

Erstellen Sie die Kro-Fähigkeit mit eksctl. *region-code*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet, und *my-cluster* ersetzen Sie es durch den Namen Ihres Clusters.

```
eksctl create capability \
  --region region-code \
  --cluster my-cluster \
  --name my-kro \
  --type KRO \
  --role-arn arn:aws:iam::[.replaceable]111122223333:role/KROCapabilityRole
```

Der Befehl kehrt sofort zurück, aber es dauert einige Zeit, bis die Funktion aktiv wird.

Schritt 3: Stellen Sie sicher, dass die Funktion aktiv ist

Überprüfen Sie den Status der Fähigkeit. *region-code*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet, und *my-cluster* ersetzen Sie es durch den Namen Ihres Clusters.

```
eksctl get capability \
  --region region-code \
  --cluster my-cluster \
  --name my-kro
```

Die Funktion ist bereit, wenn der Status angezeigt wird`ACTIVE`.

Schritt 4: Erteilen Sie Berechtigungen zur Verwaltung von Kubernetes-Ressourcen

Standardmäßig kann Kro nur deren Instanzen erstellen ResourceGraphDefinitions und verwalten. Damit kro die zugrunde liegenden Kubernetes-Ressourcen, die in Ihrem definiert sind, erstellen und verwalten kann ResourceGraphDefinitions, verknüpfen Sie die `AmazonEKSClusterAdminPolicy` Zugriffsrichtlinie mit dem Zugriffseintrag der Funktion.

Rufen Sie den ARN der Funktionsrolle ab:

```
CAPABILITY_ROLE_ARN=$(aws eks describe-capability \
  --region region-code \
  --cluster my-cluster \
  --name my-kro \
  --query 'capability.roleArn' \
```

```
  --output text)
```

Ordnen Sie die Cluster-Admin-Richtlinie zu:

```
aws eks associate-access-policy \
  --region region-code \
  --cluster my-cluster \
  --principal-arn $CAPABILITY_ROLE_ARN \
  --policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy \
  --access-scope type=cluster
```

> **Important**
>
> Die `AmazonEKSClusterAdminPolicy` gewährt umfassende Berechtigungen zur Erstellung und Verwaltung aller Kubernetes-Ressourcen und soll den Einstieg vereinfachen. Für den produktiven Einsatz sollten Sie restriktivere RBAC-Richtlinien erstellen, die nur die Berechtigungen gewähren, die für die spezifischen Ressourcen erforderlich sind, die Sie verwalten werden. ResourceGraphDefinitions Anleitungen zur Konfiguration von Berechtigungen mit den geringsten Rechten finden Sie unter und. the section called “Konfigurieren von -Berechtigungen” the section called “Überlegungen zu den Funktionen von EKS”

Schritt 5: Stellen Sie sicher, dass benutzerdefinierte Ressourcen verfügbar sind

Nachdem die Funktion aktiviert ist, stellen Sie sicher, dass benutzerdefinierte KRO-Ressourcen in Ihrem Cluster verfügbar sind:

```
kubectl api-resources | grep kro.run
```

Der `ResourceGraphDefinition` Ressourcentyp sollte aufgeführt sein.

Nächste Schritte

- the section called “Kro-Konzepte”- Verstehen Sie die KRO-Konzepte und die Zusammensetzung der Ressourcen
- the section called “Kro-Konzepte”- Erfahren Sie mehr über SimpleSchema CEL-Ausdrücke und Kompositionsmuster
- the section called “Mit Fähigkeiten arbeiten”- Verwalte deine Ressourcen für Kro-Fähigkeiten

Kro-Konzepte

kro ermöglicht es Plattformteams, benutzerdefinierte Kubernetes zu erstellen, die mehrere Ressourcen zu Abstraktionen auf APIs höherer Ebene zusammensetzen. In diesem Thema wird ein praktisches Beispiel vorgestellt und anschließend die Kernkonzepte erklärt, die Sie verstehen müssen, wenn Sie mit der EKS-Funktion für kro arbeiten.

Erste Schritte mit Kro

Nachdem Sie die Kro-Fähigkeit erstellt haben (siehethe section called “Erstellen Sie eine Kro-Fähigkeit”), können Sie APIs mit der Erstellung benutzerdefinierter Anwendungen ResourceGraphDefinitions in Ihrem Cluster beginnen.

Hier ist ein vollständiges Beispiel, das eine einfache Abstraktion einer Webanwendung erstellt:

```
apiVersion: kro.run/v1alpha1
kind: ResourceGraphDefinition
metadata:
  name: webapplication
spec:
  schema:
    apiVersion: v1alpha1
    kind: WebApplication
    group: kro.run
    spec:
      name: string | required=true
      image: string | default="nginx:latest"
      replicas: integer | default=3
  resources:
  - id: deployment
    template:
      apiVersion: apps/v1
      kind: Deployment
      metadata:
        name: ${schema.spec.name}
      spec:
        replicas: ${schema.spec.replicas}
        selector:
          matchLabels:
            app: ${schema.spec.name}
        template:
          metadata:
```

```
          labels:
            app: ${schema.spec.name}
        spec:
          containers:
          - name: app
            image: ${schema.spec.image}
            ports:
            - containerPort: 80
  - id: service
    template:
      apiVersion: v1
      kind: Service
      metadata:
        name: ${schema.spec.name}
      spec:
        selector:
          app: ${schema.spec.name}
        ports:
        - protocol: TCP
          port: 80
          targetPort: 80
```

Nach der Anwendung ResourceGraphDefinition können Anwendungsteams Webanwendungen mithilfe Ihrer vereinfachten API erstellen:

```
apiVersion: kro.run/v1alpha1
kind: WebApplication
metadata:
  name: my-app
spec:
  name: my-app
  replicas: 5
```

kro erstellt das Deployment und den Service automatisch mit der entsprechenden Konfiguration. Da `image` nicht angegeben ist, wird der Standardwert `nginx:latest` aus dem Schema verwendet.

Schlüsselkonzepte

> **Important**
>
> kro validiert zur Zeit ResourceGraphDefinitions der Erstellung, nicht zur Laufzeit. Wenn Sie eine RGD erstellen, validiert Kro die CEL-Syntax, überprüft Ausdrücke anhand der

tatsächlichen Kubernetes-Schemas, verifiziert das Vorhandensein von Feldern und erkennt zirkuläre Abhängigkeiten. Das bedeutet, dass Fehler sofort bei der Erstellung der RGD erkannt werden, bevor Instanzen bereitgestellt werden.

ResourceGraphDefinition

A ResourceGraphDefinition (RGD) definiert eine benutzerdefinierte Kubernetes-API durch Angabe von:

- Schema — Die API-Struktur unter Verwendung des SimpleSchema Formats (Feldnamen, Typen, Standardwerte, Validierung)
- Ressourcen — Vorlagen für das zugrunde liegende Kubernetes oder AWS zu erstellende Ressourcen
- Abhängigkeiten — Wie Ressourcen zueinander in Beziehung stehen (wird automatisch anhand von Feldreferenzen erkannt)

Wenn Sie eine RGD anwenden, registriert Kro eine neue benutzerdefinierte Ressourcendefinition (CRD) in Ihrem Cluster. Anwendungsteams können dann Instanzen Ihrer benutzerdefinierten API erstellen, und kro kümmert sich um die Erstellung und Verwaltung aller zugrunde liegenden Ressourcen.

Weitere Informationen finden Sie in der Kro-Dokumentation unter ResourceGraphDefinition Überblick.

SimpleSchema formatieren

SimpleSchema bietet eine vereinfachte Möglichkeit, API-Schemas zu definieren, ohne dass OpenAPI-Kenntnisse erforderlich sind:

```
schema:
  apiVersion: v1alpha1
  kind: Database
  spec:
    name: string | required=true description="Database name"
    size: string | default="small" enum=small,medium,large
    replicas: integer | default=1 minimum=1 maximum=5
```

SimpleSchema unterstützt`string`, `integer`, `boolean`, und `number` Typen mit Einschränkungen wie`required`,`default`,`minimum`/`maximum`, `enum`, und. `pattern`

Weitere Informationen finden Sie SimpleSchemain der KRO-Dokumentation.

CEL-Ausdrücke

kro verwendet Common Expression Language (CEL), um Werte dynamisch zu referenzieren und bedingte Logik hinzuzufügen. CEL-Ausdrücke sind in ${ und eingeschlossen } und können auf zwei Arten verwendet werden:

Eigenständige Ausdrücke — Der gesamte Feldwert ist ein einziger Ausdruck:

```
spec:
  replicas: ${schema.spec.replicaCount}  # Expression returns integer
  labels: ${schema.spec.labelMap}        # Expression returns object
```

Das Ausdrucksergebnis ersetzt den gesamten Feldwert und muss dem erwarteten Feldtyp entsprechen.

Zeichenkettenvorlagen — Ein oder mehrere Ausdrücke, die in eine Zeichenfolge eingebettet sind:

```
metadata:
  name: "${schema.spec.prefix}-${schema.spec.name}"  # Multiple expressions
  annotation: "Created by ${schema.spec.owner}"      # Single expression in string
```

Alle Ausdrücke in Zeichenkettenvorlagen müssen Zeichenketten zurückgeben.
Wird verwendet`string()`, um andere Typen zu konvertieren:`"replicas-${string(schema.spec.count)}"`.

Feldverweise — Greifen Sie auf Instanzspezifikationswerte zu, indem Sie`schema.spec`:

```
template:
  metadata:
    name: ${schema.spec.name}-deployment
    namespace: ${schema.metadata.namespace}  # Can also reference metadata
  spec:
    replicas: ${schema.spec.replicas}
```

Optionaler Feldzugriff — Wird ? für Felder verwendet, die möglicherweise nicht existieren:

```
# For ConfigMaps or Secrets with unknown structure
value: ${configmap.data.?DATABASE_URL}

# For optional status fields
```

```
ready: ${deployment.status.?readyReplicas > 0}
```

Wenn das Feld nicht existiert, wird der Ausdruck zurückgegeben `null` und schlägt nicht fehl.

Bedingte Ressourcen — Beziehen Sie Ressourcen nur ein, wenn die Bedingungen erfüllt sind:

```
resources:
- id: ingress
  includeWhen:
    - ${schema.spec.enableIngress == true}
  template:
    # ... ingress configuration
```

Das `includeWhen` Feld akzeptiert eine Liste von booleschen Ausdrücken. Alle Bedingungen müssen erfüllt sein, damit die Ressource erstellt werden kann. Derzeit `includeWhen` können nur `schema.spec` Felder referenziert werden.

Transformationen — Transformieren Sie Werte mit ternären Operatoren und Funktionen:

```
template:
  spec:
    resources:
      requests:
        memory: ${schema.spec.size == "small" ? "512Mi" : "2Gi"}

    # String concatenation
    image: ${schema.spec.registry + "/" + schema.spec.imageName}

    # Type conversion
    port: ${string(schema.spec.portNumber)}
```

Ressourcenübergreifende Referenzen — Referenzwerte aus anderen Ressourcen:

```
resources:
- id: bucket
  template:
    apiVersion: s3.services.k8s.aws/v1alpha1
    kind: Bucket
    spec:
      name: ${schema.spec.name}-data

- id: configmap
```

```
  template:
    apiVersion: v1
    kind: ConfigMap
    data:
      BUCKET_NAME: ${bucket.spec.name}
      BUCKET_ARN: ${bucket.status.ackResourceMetadata.arn}
```

Wenn Sie in einem CEL-Ausdruck auf eine andere Ressource verweisen, wird automatisch eine Abhängigkeit erstellt. kro stellt sicher, dass die referenzierte Ressource zuerst erstellt wird.

Weitere Informationen finden Sie in der KRO-Dokumentation unter CEL-Ausdrücke.

Abhängigkeiten von Ressourcen

kro leitet Abhängigkeiten automatisch aus CEL-Ausdrücken ab — Sie geben die Reihenfolge nicht an, sondern beschreiben Beziehungen. Wenn eine Ressource mithilfe eines CEL-Ausdrucks auf eine andere Ressource verweist, erstellt Kro eine Abhängigkeit und bestimmt die richtige Erstellungsreihenfolge.

```
resources:
- id: bucket
  template:
    apiVersion: s3.services.k8s.aws/v1alpha1
    kind: Bucket
    spec:
      name: ${schema.spec.name}-data

- id: notification
  template:
    apiVersion: s3.services.k8s.aws/v1alpha1
    kind: BucketNotification
    spec:
      bucket: ${bucket.spec.name}  # Creates dependency: notification depends on bucket
```

Der Ausdruck `${bucket.spec.name}` erzeugt eine Abhängigkeit. kro erstellt einen gerichteten azyklischen Graphen (DAG) aller Ressourcen und ihrer Abhängigkeiten und berechnet dann eine topologische Reihenfolge für die Erstellung.

Erstellungsreihenfolge: Ressourcen werden in topologischer Reihenfolge erstellt (Abhängigkeiten zuerst).

Parallele Erstellung: Ressourcen ohne Abhängigkeiten werden gleichzeitig erstellt.

Löschreihenfolge: Ressourcen werden in umgekehrter topologischer Reihenfolge gelöscht (abhängige Objekte zuerst).

Zirkuläre Abhängigkeiten: Nicht zulässig — kro lehnt ResourceGraphDefinitions bei der Validierung zirkuläre Abhängigkeiten ab.

Sie können die berechnete Erstellungsreihenfolge einsehen:

```
kubectl get resourcegraphdefinition my-rgd -o jsonpath='{.status.topologicalOrder}'
```

Weitere Informationen finden Sie unter Graph-Inferenz in der KRO-Dokumentation.

Komponieren mit ACK

kro arbeitet nahtlos mit der EKS-Funktion für ACK zusammen, um AWS Ressourcen mit Kubernetes-Ressourcen zusammenzustellen:

```
resources:
# Create {aws} S3 bucket with ACK
- id: bucket
  template:
    apiVersion: s3.services.k8s.aws/v1alpha1
    kind: Bucket
    spec:
      name: ${schema.spec.name}-files

# Inject bucket details into Kubernetes ConfigMap
- id: config
  template:
    apiVersion: v1
    kind: ConfigMap
    data:
      BUCKET_NAME: ${bucket.spec.name}
      BUCKET_ARN: ${bucket.status.ackResourceMetadata.arn}

# Use ConfigMap in application deployment
- id: deployment
  template:
    apiVersion: apps/v1
    kind: Deployment
    spec:
      template:
        spec:
```

```
          containers:
          - name: app
            envFrom:
            - configMapRef:
                name: ${config.metadata.name}
```

Mit diesem Muster können Sie AWS Ressourcen erstellen, ihre Details (ARNs, URLs, Endpunkte) extrahieren und sie in Ihre Anwendungskonfiguration einfügen — alles wird als eine Einheit verwaltet.

Weitere Kompositionsmuster und Beispiele für Fortgeschrittene finden Sie unter. the section called “Überlegungen”

Nächste Schritte

- the section called “Überlegungen”- Erfahren Sie mehr über EKS-spezifische Muster, RBAC und die Integration mit ACK und Argo CD
- kro-Dokumentation — Umfassende Kro-Dokumentation mit erweiterten CEL-Ausdrücken, Validierungsmustern und Problemlösungen

Kro-Berechtigungen konfigurieren

Im Gegensatz zu ACK und Argo CD benötigt kro keine IAM-Berechtigungen. kro arbeitet vollständig in Ihrem Kubernetes-Cluster und führt keine API-Aufrufe durch. AWS Steuern Sie den Zugriff auf Kro-Ressourcen mithilfe des Standard-Kubernetes-RBAC.

Wie funktionieren Berechtigungen mit Kro

kro verwendet zwei Arten von Kubernetes-Ressourcen mit unterschiedlichen Bereichen:

ResourceGraphDefinitions: Ressourcen im Clusterbereich, die benutzerdefinierte definieren. APIs In der Regel von Plattformteams verwaltet, die organisatorische Standards entwerfen und einhalten.

Instanzen: Benutzerdefinierte Ressourcen mit Namespace-Gültigkeitsbereich, die aus erstellt wurden. ResourceGraphDefinitions Kann von Anwendungsteams mit entsprechenden RBAC-Berechtigungen erstellt werden.

Standardmäßig verfügt die KRO-Funktion über die Zugriffsrichtlinie über Berechtigungen zum Verwalten ResourceGraphDefinitions und Verwalten ihrer Instanzen. `AmazonEKSKROPolicy` Für Kro sind jedoch zusätzliche Berechtigungen erforderlich, um die zugrunde liegenden Kubernetes-Ressourcen zu erstellen und zu verwalten, die in Ihrem System definiert sind

ResourceGraphDefinitions (z. B. Bereitstellungen, Dienste oder ACK-Ressourcen). Sie müssen diese Berechtigungen über Zugriffsrichtlinien oder Kubernetes RBAC gewähren. Einzelheiten zur Erteilung dieser Berechtigungen finden Sie unter Kro Arbitrary Resource Permissions.

Berechtigungen für Plattformteams

Plattformteams benötigen Berechtigungen zum Erstellen und Verwalten ResourceGraphDefinitions.

Beispiel ClusterRole für Plattformteams:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: kro-platform-admin
rules:
- apiGroups: ["kro.run"]
  resources: ["resourcegraphdefinitions"]
  verbs: ["*"]
```

Binden Sie sich an Mitglieder des Plattform-Teams:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: platform-team-kro-admin
subjects:
- kind: Group
  name: platform-team
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: kro-platform-admin
  apiGroup: rbac.authorization.k8s.io
```

Berechtigungen des Anwendungsteams

Anwendungsteams benötigen Berechtigungen, um Instanzen benutzerdefinierter Ressourcen in ihren Namespaces zu erstellen.

Beispielrolle für Anwendungsteams:

```
apiVersion: rbac.authorization.k8s.io/v1
```

```
kind: Role
metadata:
  name: kro-app-developer
  namespace: my-app
rules:
- apiGroups: ["kro.run"]
  resources: ["webapps", "databases"]
  verbs: ["create", "get", "list", "update", "delete", "patch"]
```

An Mitglieder des Anwendungsteams binden:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: app-team-kro-developer
  namespace: my-app
subjects:
- kind: Group
  name: app-team
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: kro-app-developer
  apiGroup: rbac.authorization.k8s.io
```

> **Note**
>
> Die API-Gruppe in der Rolle (`kro.run`in diesem Beispiel) muss der in Ihrem ResourceGraphDefinition Schema `apiVersion` definierten entsprechen.

Schreibgeschützter Zugriff

Gewähren Sie schreibgeschützten Zugriff auf Ansicht ResourceGraphDefinitions und Instanzen ohne Änderungsberechtigungen.

Schreibgeschützt: ClusterRole

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
```

```
  name: kro-viewer
rules:
- apiGroups: ["kro.run"]
  resources: ["resourcegraphdefinitions"]
  verbs: ["get", "list", "watch"]
```

Schreibgeschützte Rolle für Instanzen:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: kro-instance-viewer
  namespace: my-app
rules:
- apiGroups: ["kro.run"]
  resources: ["webapps", "databases"]
  verbs: ["get", "list", "watch"]
```

Zugriff auf mehrere Namespaces

Gewähren Sie Anwendungsteams mithilfe von with Zugriff auf mehrere Namespaces. ClusterRoles RoleBindings

ClusterRole für den Zugriff auf mehrere Namespaces:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: kro-multi-namespace-developer
rules:
- apiGroups: ["kro.run"]
  resources: ["webapps"]
  verbs: ["create", "get", "list", "update", "delete"]
```

An bestimmte Namespaces binden:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: app-team-dev-access
  namespace: development
subjects:
```

```
- kind: Group
  name: app-team
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: kro-multi-namespace-developer
  apiGroup: rbac.authorization.k8s.io
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: app-team-staging-access
  namespace: staging
subjects:
- kind: Group
  name: app-team
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: kro-multi-namespace-developer
  apiGroup: rbac.authorization.k8s.io
```

Best Practices

Prinzip der geringsten Rechte: Erteilen Sie nur die Mindestberechtigungen, die für die jeweiligen Aufgaben der einzelnen Teams erforderlich sind.

Verwenden Sie Gruppen statt einzelner Benutzer: Binden Sie Rollen eher an Gruppen als an einzelne Benutzer, um die Verwaltung zu vereinfachen.

Getrennte Plattform- und Anwendungsaspekte: Plattformteams verwalten ResourceGraphDefinitions, Anwendungsteams verwalten Instanzen.

Namespace-Isolierung: Verwenden Sie Namespaces, um verschiedene Teams oder Umgebungen zu isolieren, wobei RBAC den Zugriff auf jeden Namespace steuert.

Schreibgeschützter Zugriff für Auditzwecke: Gewähren Sie Sicherheits- und Compliance-Teams zu Prüfungszwecken nur Lesezugriff.

Nächste Schritte

- the section called “Kro-Konzepte”- Verstehen Sie die KRO-Konzepte und die Zusammensetzung der Ressourcen

- the section called “Kro-Konzepte”- Verstehen Sie SimpleSchema CEL-Ausdrücke und Kompositionsmuster
- the section called “Überlegungen zu den Funktionen von EKS”- Informieren Sie sich über bewährte Sicherheitsverfahren im Hinblick auf Funktionen

Überlegungen zu Kro für EKS

In diesem Thema werden wichtige Überlegungen zur Verwendung der EKS-Funktion für KRO behandelt, einschließlich der Frage, wann die Ressourcenzusammensetzung, RBAC-Muster und die Integration mit anderen EKS-Funktionen verwendet werden sollten.

Wann sollte Kro verwendet werden

kro wurde für die Erstellung wiederverwendbarer und benutzerdefinierter Infrastrukturmuster entwickelt APIs , die das komplexe Ressourcenmanagement vereinfachen.

Verwenden Sie kro, wenn Sie:

- Erstellen Sie Self-Service-Plattformen mit vereinfachten Funktionen APIs für Anwendungsteams
- Standardisieren Sie die Infrastrukturmuster für alle Teams (Datenbank + Backup + Überwachung)
- Verwalten Sie Ressourcenabhängigkeiten und geben Sie Werte zwischen Ressourcen weiter
- Erstellen Sie benutzerdefinierte Abstraktionen, die die Komplexität der Implementierung verbergen
- Stellen Sie mehrere ACK-Ressourcen zu übergeordneten Bausteinen zusammen
- Ermöglichen Sie GitOps Workflows für komplexe Infrastruktur-Stacks

Verwenden Sie Kro nicht, wenn:

- Verwaltung einfacher, eigenständiger Ressourcen (verwenden Sie ACK- oder Kubernetes-Ressourcen direkt)
- Sie benötigen eine dynamische Laufzeitlogik (kro ist deklarativ, nicht zwingend erforderlich)
- Ressourcen haben keine Abhängigkeiten oder eine gemeinsame Konfiguration

kro zeichnet sich durch die Schaffung von „gepflasterten Pfaden“ aus — eigensinnige, wiederverwendbare Muster, die es Teams leicht machen, komplexe Infrastrukturen korrekt bereitzustellen.

RBAC-Muster

kro ermöglicht die Trennung von Problemen zwischen Plattformteams, die Instanzen erstellen, ResourceGraphDefinitions und Anwendungsteams, die Instanzen erstellen.

Verantwortlichkeiten des Plattformteams

Plattformteams erstellen und verwalten ResourceGraphDefinitions (RGDs), die benutzerdefiniert definieren APIs.

Erforderliche Berechtigungen:

- Erstellen, aktualisieren, löschen ResourceGraphDefinitions
- Verwalten Sie die zugrunde liegenden Ressourcentypen (Bereitstellungen, Dienste, ACK-Ressourcen)
- Zugriff auf alle Namespaces, in denen sie verwendet werden RGDs

Beispiel ClusterRole für ein Plattformteam:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: platform-kro-admin
rules:
- apiGroups: ["kro.run"]
  resources: ["resourcegraphdefinitions"]
  verbs: ["*"]
- apiGroups: ["*"]
  resources: ["*"]
  verbs: ["get", "list", "watch"]
```

Eine ausführliche RBAC-Konfiguration finden Sie unter. the section called “Konfigurieren von - Berechtigungen”

Verantwortlichkeiten des Anwendungsteams

Anwendungsteams erstellen Instanzen von benutzerdefinierten Ressourcen, die von definiert sind, RGDs ohne die zugrundeliegende Komplexität verstehen zu müssen.

Erforderliche Berechtigungen:

- Instanzen von benutzerdefinierten Ressourcen erstellen, aktualisieren und löschen
- Lesezugriff auf ihren Namespace
- Kein Zugriff auf zugrunde liegende Ressourcen oder RGDs

Vorteile:

- Teams verwenden einfache, hochrangige APIs
- Plattformteams kontrollieren die Implementierungsdetails
- Geringeres Risiko einer Fehlkonfiguration
- Schnelleres Onboarding für neue Teammitglieder

Integration mit anderen EKS-Funktionen

Zusammenstellung von ACK-Ressourcen

Kro ist besonders leistungsfähig, wenn es mit ACK kombiniert wird, um Infrastrukturmuster zu erstellen.

Allgemeine Muster:

- Anwendung mit Speicher: S3-Bucket + SQS-Warteschlange + Benachrichtigungskonfiguration
- Datenbank-Stack: RDS-Instanz + Parametergruppe + Sicherheitsgruppe + Secrets Manager Secret
- Netzwerk: VPC + Subnetze + Routentabellen + Sicherheitsgruppen + NAT-Gateways
- Datenverarbeitung mit Speicher: EC2 Instanz + EBS-Volumes + IAM-Instanzprofil

kro kümmert sich um die Reihenfolge der Abhängigkeiten, übergibt Werte zwischen Ressourcen (wie Zeichenketten ARNs und Verbindungszeichenfolgen) und verwaltet den gesamten Lebenszyklus als eine einzige Einheit.

Beispiele für das Zusammenstellen von ACK-Ressourcen finden Sie unter. the section called “ACK-Konzepte”

GitOps mit Argo CD

Verwenden Sie die EKS-Funktion für Argo CD, um sowohl RGDs Instanzen als auch Instanzen aus Git-Repositorys bereitzustellen.

Organisation des Repositorys:

- Plattform-Repo: Enthält, die vom Plattformteam ResourceGraphDefinitions verwaltet werden
- Anwendungs-Repos: Enthalten Instanzen von benutzerdefinierten Ressourcen, die von App-Teams verwaltet werden
- Gemeinsames Repo: Enthält RGDs sowohl als auch Instanzen für kleinere Organisationen

Überlegungen:

- RGDs Vor den Instanzen bereitstellen (Argo CD Sync Waves kann helfen)
- Verwenden Sie separate Argo-CD-Projekte für Plattform- und Anwendungsteams
- Das Plattformteam kontrolliert den Zugriff auf das RGD-Repository
- Anwendungsteams haben nur Lesezugriff auf RGD-Definitionen

Weitere Informationen zu Argo CD finden Sie unter. the section called “Arbeiten mit Argo CD”

Organisieren ResourceGraphDefinitions

Organisieren Sie RGDs nach Zweck, Komplexität und Eigentümerschaft.

Nach Zweck:

- Infrastruktur: Datenbank-Stacks, Netzwerke, Speicher
- Anwendung: Web-Apps APIs, Batch-Jobs
- Plattform: Gemeinsame Dienste, Überwachung, Protokollierung

Nach Komplexität:

- Einfach: 2-3 Ressourcen mit minimalen Abhängigkeiten
- Moderat: 5-10 Ressourcen mit einigen Abhängigkeiten
- Komplex: Mehr als 10 Ressourcen mit komplexen Abhängigkeiten

Benennungskonventionen:

- Verwenden Sie beschreibende Namen:`webapp-with-database`, `s3-notification-queue`
- Fügen Sie bei wichtigen Änderungen die Version in den Namen ein: `webapp-v2`
- Verwenden Sie konsistente Präfixe für verwandte RGDs:`platform-` , `app-`

Namespace-Strategie:

- RGDs sind clusterbezogen (nicht namespaces)
- Instanzen haben einen Namespace
- Verwenden Sie Namespace-Selektoren, RGDs um zu steuern, wo Instanzen erstellt werden können

Versionierung und Updates

Planen Sie die RGD-Entwicklung und die Instanzmigration ein.

RGD-Aktualisierungen:

- Wichtige Änderungen: RGD wurde bereits aktualisiert (optionale Felder und neue Ressourcen mit includeWhen hinzufügen)
- Bahnbrechende Änderungen: Erstelle eine neue RGD mit einem anderen Namen (webapp-v2)
- Veraltet: Veraltete Version mit Anmerkungen markieren, Zeitplan für die Migration RGDs mitteilen

Migration von Instanzen:

- Erstellen Sie neue Instanzen mit aktualisiertem RGD
- Stellen Sie sicher, dass neue Instanzen korrekt funktionieren
- Löschen Sie alte Instanzen
- kro verarbeitet die Aktualisierungen der zugrunde liegenden Ressourcen automatisch

Bewährte Verfahren:

- Testen Sie RGD-Änderungen zunächst in Umgebungen außerhalb der Produktion
- Verwenden Sie für größere Änderungen die semantische Versionierung in RGD-Namen
- Dokumentieren Sie wichtige Änderungen und Migrationspfade
- Stellen Sie Migrationsbeispiele für Anwendungsteams bereit

Validierung und Testen

Validieren RGDs Sie vor der Bereitstellung in der Produktion.

Strategien zur Validierung:

- Schemavalidierung: kro validiert die RGD-Struktur automatisch
- Testinstanzen: Erstellen Sie Testinstanzen in Entwicklungs-Namespaces
- Integrationstests: Stellen Sie sicher, dass zusammengesetzte Ressourcen zusammenarbeiten
- Durchsetzung von Richtlinien: Setzen Sie Zugangscontroller ein, um Unternehmensstandards durchzusetzen

Häufig zu testende Probleme:

- Abhängigkeiten und Reihenfolge von Ressourcen
- Werteübergabe zwischen Ressourcen (CEL-Ausdrücke)
- Bedingte Einbeziehung von Ressourcen (includeWhen)
- Weitergabe des Status anhand der zugrunde liegenden Ressourcen
- RBAC-Berechtigungen für die Instanzerstellung

Upstream-Dokumentation

Ausführliche Informationen zur Verwendung von kro finden Sie unter

- Erste Schritte mit Kro — Erstellen ResourceGraphDefinitions
- CEL-Ausdrücke — CEL-Ausdrücke schreiben
- kro Guides - Kompositionsmuster für Fortgeschrittene
- Problembehandlung — Problembehebung und Debugging

Nächste Schritte

- the section called “Konfigurieren von -Berechtigungen”- Konfigurieren Sie RBAC für Plattform- und Anwendungsteams
- the section called “Kro-Konzepte”- Verstehen Sie die KRO-Konzepte und den Ressourcenlebenszyklus
- the section called “Fehlerbehebung”- Beheben Sie Kro-Probleme
- the section called “ACK-Konzepte”- Erfahren Sie mehr über ACK-Ressourcen für die Komposition
- the section called “Arbeiten mit Argo CD”- Bereitstellung RGDs und Instanzen mit GitOps

Probleme mit KRO-Funktionen beheben

Dieses Thema enthält Anleitungen zur Fehlerbehebung für die EKS-Funktion für KRO, einschließlich Funktionsprüfungen, RBAC-Berechtigungen, CEL-Ausdrucksfehler und Probleme bei der Ressourcenzusammensetzung.

> **Note**
>
> EKS-Funktionen werden vollständig verwaltet und außerhalb Ihres Clusters ausgeführt. Sie haben keinen Zugriff auf Controller-Logs oder den `kro-system` Namespace. Die Fehlerbehebung konzentriert sich auf den Funktionsstatus, die RBAC-Konfiguration und den Ressourcenstatus.

Die Funktion ist AKTIV, funktioniert aber ResourceGraphDefinitions nicht

Wenn Ihre Kro-Fähigkeit `ACTIVE` den Status anzeigt, aber ResourceGraphDefinitions keine zugrunde liegenden Ressourcen erstellt werden, überprüfen Sie den Zustand der Fähigkeit, die RBAC-Berechtigungen und den Ressourcenstatus.

Überprüfen Sie den Zustand der Fähigkeit:

Sie können Funktionszustands- und Statusprobleme in der EKS-Konsole oder über die AWS CLI anzeigen.

Konsole:

1. Öffnen Sie die Amazon EKS-Konsole unter https://console.aws.amazon.com/eks/ home#/clusters.
2. Wählen Sie Ihren Clusternamen aus.
3. Wählen Sie den Registerkarte Beobachtbarkeit.
4. Wählen Sie Cluster überwachen aus.
5. Wählen Sie die Registerkarte Funktionen, um den Zustand und den Status aller Funktionen anzuzeigen.

AWS CLI:

```
# View capability status and health
aws eks describe-capability \
```

```
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-kro

# Look for issues in the health section
```

Häufige Ursachen:

- RBAC-Berechtigungen fehlen: Kro hat keine Berechtigungen, um zugrunde liegende Kubernetes-Ressourcen zu erstellen
- Ungültige CEL-Ausdrücke: Syntaxfehler in ResourceGraphDefinition
- Ressourcenabhängigkeiten: Abhängige Ressourcen sind nicht bereit
- Schemavalidierung: Die Instanz entspricht nicht den RGD-Schemaanforderungen

Überprüfen Sie die RBAC-Berechtigungen:

```
# Check if capability has cluster admin policy
kubectl get accessentry -A | grep kro
```

Wenn die Funktion nicht über die erforderlichen Berechtigungen verfügt, verknüpfen Sie sie `AmazonEKSClusterAdminPolicy` mit dem Zugriffseintrag der KRO-Funktion, oder erstellen Sie restriktivere RBAC-Richtlinien für den Produktionsgebrauch. Details dazu finden Sie unter the section called “Konfigurieren von -Berechtigungen”.

Status überprüfen: ResourceGraphDefinition

```
# List all RGDs
kubectl get resourcegraphdefinition

# Describe specific RGD
kubectl describe resourcegraphdefinition my-rgd

# Check for validation errors
kubectl get resourcegraphdefinition my-rgd -o jsonpath='{.status.conditions}'
```

ResourceGraphDefinitions haben drei wichtige Statusbedingungen:

- `ResourceGraphAccepted`- Ob das RGD die Validierung bestanden hat (CEL-Syntax, Typprüfung, Existenz des Felds)

- `KindReady`- Ob die CRD für Ihre benutzerdefinierte API generiert und registriert wurde
- `ControllerReady`- Ob Kro aktiv nach Instanzen Ihrer benutzerdefinierten API Ausschau hält

Falls ja `ResourceGraphAcceptedFalse`, überprüfen Sie die Zustandsmeldung auf Validierungsfehler wie unbekannte Felder, nicht übereinstimmende Typen oder zirkuläre Abhängigkeiten.

Instanzen wurden erstellt, aber die zugrunde liegenden Ressourcen werden nicht angezeigt

Wenn benutzerdefinierte Ressourceninstanzen existieren, die zugrunde liegenden Kubernetes-Ressourcen (Bereitstellungen, Dienste, ConfigMaps) jedoch nicht erstellt werden, überprüfen Sie, ob kro über die erforderlichen Berechtigungen verfügt, und überprüfen Sie, ob Kompositionsfehler vorliegen.

Überprüfen Sie den Instanzstatus:

```
# Describe the instance (replace with your custom resource kind and name)
kubectl describe custom-kind
         my-instance

# View instance events
kubectl get events --field-selector involvedObject.name=my-instance

# Check instance status conditions
kubectl get custom-kind
         my-instance -o jsonpath='{.status.conditions}'

# Check instance state
kubectl get custom-kind
         my-instance -o jsonpath='{.status.state}'
```

Instanzen haben ein `state` Feld, das den Status auf hoher Ebene anzeigt:

- `ACTIVE`— Die Instanz wird erfolgreich ausgeführt
- `IN_PROGRESS`- Die Instanz wird verarbeitet oder abgeglichen
- `FAILED`- Die Instanz konnte nicht abgeglichen werden
- `DELETING`- Die Instanz wird gelöscht
- `ERROR`- Bei der Verarbeitung ist ein Fehler aufgetreten

Instanzen haben außerdem vier Statusbedingungen:

- `InstanceManaged`- Finalizer und Labels sind richtig gesetzt
- `GraphResolved`- Laufzeitdiagramm erstellt und Ressourcen aufgelöst
- `ResourcesReady`- Alle Ressourcen wurden erstellt und sind bereit
- `Ready`- Allgemeiner Zustand der Instanz (wird nur erreicht`True`, wenn alle Unterbedingungen erfüllt sind`True`)

Konzentrieren Sie sich auf den `Ready` Zustand, um den Zustand der Instanz zu bestimmen. Falls `Ready` ja`False`, überprüfen Sie anhand der Unterbedingungen, welche Phase fehlgeschlagen ist.

Überprüfen Sie die RBAC-Berechtigungen:

Für die Kro-Funktion sind Berechtigungen erforderlich, um die zugrunde liegenden Kubernetes-Ressourcen zu erstellen, die in Ihrem definiert sind. ResourceGraphDefinitions

```
# Check if the capability has the AmazonEKSClusterAdminPolicy
kubectl get accessentry -A | grep kro
```

Wenn Berechtigungen fehlen, verknüpfen Sie sie `AmazonEKSClusterAdminPolicy` mit dem Zugriffseintrag der Kro-Fähigkeit oder erstellen Sie restriktivere RBAC-Richtlinien für den Produktionsgebrauch. Details dazu finden Sie unter the section called “Konfigurieren von - Berechtigungen”.

Fehler bei CEL-Ausdrücken

Fehler bei CEL-Ausdrücken werden bei der ResourceGraphDefinition Erstellung erkannt, nicht bei der Erstellung von Instances. Kro validiert die gesamte CEL-Syntax, überprüft Ausdrücke anhand von Kubernetes-Schemas und überprüft das Vorhandensein von Feldern, wenn Sie die RGD erstellen.

Häufige Fehler bei der CEL-Validierung:

- Undefinierter Feldverweis: Verweisen auf ein Feld, das im Schema oder in der Ressource nicht vorhanden ist
- Typkonflikt: Der Ausdruck gibt einen falschen Typ zurück (z. B. eine Zeichenfolge, bei der eine Ganzzahl erwartet wurde)
- Ungültige Syntax: Fehlende Klammern, Anführungszeichen oder Operatoren im CEL-Ausdruck
- Unbekannter Ressourcentyp: Verweist auf eine CRD, die im Cluster nicht vorhanden ist

Überprüfen Sie den RGD-Validierungsstatus:

```
# Check if RGD was accepted
kubectl get resourcegraphdefinition my-rgd -o jsonpath='{.status.conditions[?
(@.type=="ResourceGraphAccepted")]}'

# View detailed validation errors
kubectl describe resourcegraphdefinition my-rgd
```

Falls ja `ResourceGraphAcceptedFalse`, enthält die Zustandsmeldung den Validierungsfehler.

Beispiel für gültige CEL-Ausdrücke:

```
# Reference schema field
${schema.spec.appName}

# Conditional expression
${schema.spec.replicas > 1}

# String template (expressions must return strings)
name: "${schema.spec.appName}-service"

# Standalone expression (can be any type)
replicas: ${schema.spec.replicaCount}

# Resource reference
${deployment.status.availableReplicas}

# Optional field access (returns null if field doesn't exist)
${configmap.data.?DATABASE_URL}
```

Ressourcenabhängigkeiten werden nicht aufgelöst

kro leitet automatisch Abhängigkeiten aus CEL-Ausdrücken ab und erstellt Ressourcen in der richtigen Reihenfolge. Wenn Ressourcen nicht wie erwartet erstellt werden, überprüfen Sie die Reihenfolge der Abhängigkeiten und die Verfügbarkeit der Ressourcen.

Berechnete Erstellungsreihenfolge anzeigen:

```
# See the order kro will create resources
kubectl get resourcegraphdefinition my-rgd -o jsonpath='{.status.topologicalOrder}'
```

Dies zeigt die berechnete Reihenfolge auf der Grundlage von CEL-Ausdrucksverweisen zwischen Ressourcen.

Überprüfen Sie die Verfügbarkeit der Ressourcen:

```
# View instance status to see which resources are ready
kubectl get custom-kind
         my-instance -o jsonpath='{.status}'

# Check specific resource status
kubectl get deployment my-deployment -o jsonpath='{.status.conditions}'
```

Überprüfen Sie die ReadyWhen-Bedingungen (falls verwendet):

Das Feld `readyWhen` ist optional. Wenn nicht angegeben, gelten Ressourcen unmittelbar nach der Erstellung als bereit. Wenn Sie `readyWhen` Bedingungen definiert haben, stellen Sie sicher, dass diese korrekt prüfen, ob die Ressourcen bereit sind:

```
resources:
  - id: deployment
    readyWhen:
      - ${deployment.status.availableReplicas == deployment.spec.replicas}
```

Überprüfen Sie die Ressourcenereignisse:

```
# View events for the underlying resources
kubectl get events -n namespace --sort-by='.lastTimestamp'
```

Fehler bei der Schemavalidierung

Wenn Instanzen aufgrund von Fehlern bei der Schemavalidierung nicht erstellt werden können, überprüfen Sie, ob die Instanz die RGD-Schemaanforderungen erfüllt.

Überprüfen Sie die Validierungsfehler:

```
# Attempt to create instance and view error
kubectl apply -f instance.yaml

# View existing instance validation status
kubectl describe custom-kind
```

```
        my-instance | grep -A 5 "Validation"
```

Häufige Probleme bei der Validierung:

- Erforderliche Felder fehlen: Die Instanz stellt nicht alle erforderlichen Schemafelder bereit
- Typkonflikt: Es wird eine Zeichenfolge bereitgestellt, bei der eine Ganzzahl erwartet wird
- Ungültiger Aufzählungswert: Es wird ein Wert verwendet, der nicht in der Liste der zulässigen Werte steht
- Musterkonflikt: Die Zeichenfolge stimmt nicht mit dem Regex-Muster überein

RGD-Schema überprüfen:

```
# View the schema definition
kubectl get resourcegraphdefinition my-rgd -o jsonpath='{.spec.schema}'
```

Stellen Sie sicher, dass Ihre Instanz alle erforderlichen Felder mit den richtigen Typen bereitstellt.

Nächste Schritte

- the section called “Überlegungen”- Überlegungen und bewährte Methoden zu KRO
- the section called “Konfigurieren von -Berechtigungen”- Konfigurieren Sie RBAC für Plattform- und Anwendungsteams
- the section called “Kro-Konzepte”- Verstehen Sie die KRO-Konzepte und den Ressourcenlebenszyklus
- the section called “Problembehebung bei Funktionen”- Allgemeine Hinweise zur Problembehebung

Vergleich der EKS-Funktionen für Kro mit selbstverwaltetem Kro

Die EKS-Funktion für Kro bietet dieselbe Funktionalität wie selbstverwaltetes Kro, bietet jedoch erhebliche betriebliche Vorteile. Einen allgemeinen Vergleich zwischen EKS-Funktionen und selbstverwalteten Lösungen finden Sie unter. the section called “Überlegungen”

Die EKS-Funktion für Kro verwendet dieselben Upstream-Kro-Controller und ist vollständig kompatibel mit Upstream-Kro. ResourceGraphDefinitions, CEL-Ausdrücke und Ressourcenzusammensetzung funktionieren identisch. Eine vollständige Kro-Dokumentation und Beispiele finden Sie in der Kro-Dokumentation.

Migrationspfad

Sie können ohne Ausfallzeiten von selbstverwaltetem KRO zur verwalteten Funktion migrieren.

> **Important**
>
> Stellen Sie vor der Migration sicher, dass auf Ihrem selbstverwalteten Kro-Controller dieselbe Version wie die EKS-Funktion für Kro ausgeführt wird. Überprüfen Sie die Funktionsversion in der EKS-Konsole oder verwenden Sie Ihre selbstverwaltete Installation `aws eks describe-capability` und führen Sie anschließend ein entsprechendes Upgrade durch. Dadurch werden Kompatibilitätsprobleme während der Migration vermieden.

1. Aktualisieren Sie Ihren selbstverwalteten Kro-Controller so, dass er ihn `kube-system` für Leasingverträge zur Wahl von Führungskräften verwenden kann:

   ```
   helm upgrade --install kro \
     oci://ghcr.io/awslabs/kro/kro-chart \
     --namespace kro \
     --set leaderElection.namespace=kube-system
   ```

   Dadurch wird der Leasingvertrag des Controllers auf diesen übertragen`kube-system`, sodass sich die verwaltete Kapazität darauf abstimmen kann.
2. Erstellen Sie die Kro-Fähigkeit auf Ihrem Cluster (siehethe section called “Erstellen Sie eine Kro-Fähigkeit”)
3. Die verwaltete Funktion erkennt bestehende Instanzen ResourceGraphDefinitions und übernimmt den Abgleich
4. Skalieren Sie selbstverwaltete KRO-Bereitstellungen schrittweise oder entfernen Sie sie:

   ```
   helm uninstall kro --namespace kro
   ```

Dieser Ansatz ermöglicht eine sichere Koexistenz beider Controller während der Migration. Die verwaltete Funktion übernimmt automatisch Instanzen ResourceGraphDefinitions und Instanzen, die zuvor von einem selbstverwalteten KRO verwaltet wurden, sodass ein kontinuierlicher Abgleich ohne Konflikte gewährleistet ist.

Nächste Schritte

- the section called “Erstellen Sie eine Kro-Fähigkeit”- Erstellen Sie eine Kro-Fähigkeitsressource
- the section called “Kro-Konzepte”- Verstehen Sie die KRO-Konzepte und die Zusammensetzung der Ressourcen

Problembehebung bei EKS-Funktionen

Dieses Thema enthält allgemeine Anleitungen zur Fehlerbehebung bei EKS-Funktionen, einschließlich Funktionsprüfungen, häufig auftretenden Problemen und Links zur funktionsspezifischen Problembehebung.

> **Note**
>
> Die EKS-Funktionen werden vollständig verwaltet und außerhalb Ihres Clusters ausgeführt. Sie haben keinen Zugriff auf Controller-Logs oder Controller-Namespaces. Die Fehlerbehebung konzentriert sich auf den Zustand der Funktionen, den Ressourcenstatus und die Konfiguration.

Allgemeiner Ansatz zur Fehlerbehebung

Gehen Sie bei der Behebung von Problemen mit EKS-Funktionen wie folgt vor:

1. Funktionsstatus überprüfen: Dient `aws eks describe-capability` zum Anzeigen des Funktionsstatus und der Integritätsprobleme
2. Überprüfen Sie den Ressourcenstatus: Überprüfen Sie die von Ihnen erstellten Kubernetes-Ressourcen (CRDs) auf Statusbedingungen und Ereignisse
3. Überprüfen Sie die IAM-Berechtigungen: Stellen Sie sicher, dass die Capability Role über die erforderlichen Berechtigungen verfügt
4. Konfiguration überprüfen: Stellen Sie sicher, dass die funktionsspezifische Konfiguration korrekt ist

Überprüfen Sie den Zustand der Funktionen

Alle EKS-Funktionen stellen Statusinformationen über die EKS-Konsole und die `describe-capability` API bereit.

Konsole:

1. Öffnen Sie die Amazon EKS-Konsole unter https://console.aws.amazon.com/eks/ home#/clusters.
2. Wählen Sie Ihren Clusternamen aus.
3. Wählen Sie den Registerkarte Beobachtbarkeit.
4. Wählen Sie Cluster überwachen aus.
5. Wählen Sie die Registerkarte Funktionen, um den Zustand und den Status aller Funktionen anzuzeigen.

Auf der Registerkarte Funktionen wird Folgendes angezeigt:

- Name und Typ der Fähigkeit
- Aktueller Status
- Gesundheitsprobleme, mit Beschreibung

AWS CLI:

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-capability-name
```

Die Antwort enthält:

- Status: Aktueller Funktionsstatus (`CREATING`,`ACTIVE`,`UPDATING`,`DELETING`,`CREATE_FAILED`,`UPDATE_FAILED`)
- Health: Gesundheitsinformationen, einschließlich aller Probleme, die von der Fähigkeit erkannt wurden

Allgemeine Status der Fähigkeiten

ERSTELLUNG: Die Fähigkeit wird eingerichtet.

AKTIV: Die Funktion läuft und ist einsatzbereit. Wenn Ressourcen nicht wie erwartet funktionieren, überprüfen Sie den Ressourcenstatus und die IAM-Berechtigungen.

AKTUALISIERUNG: Die Konfigurationsänderungen werden übernommen. Warten Sie, bis der Status wieder angezeigt wird`ACTIVE`.

CREATE_FAILED oder UPDATE_FAILED: Beim Setup oder Update ist ein Fehler aufgetreten. Einzelheiten finden Sie im Abschnitt „Gesundheit“. Häufige Ursachen:

- Die Vertrauensrichtlinie für IAM-Rollen ist falsch oder fehlt
- Die IAM-Rolle ist nicht vorhanden oder es kann nicht darauf zugegriffen werden
- Probleme mit dem Clusterzugriff
- Ungültige Konfigurationsparameter

Überprüfen Sie den Kubernetes-Ressourcenstatus

EKS-Funktionen erstellen und verwalten benutzerdefinierte Kubernetes-Ressourcendefinitionen (CRDs) in Ihrem Cluster. Überprüfen Sie bei der Fehlerbehebung den Status der Ressourcen, die Sie erstellt haben:

```
# List resources of a specific type
kubectl get resource-kind -A

# Describe a specific resource to see conditions and events
kubectl describe resource-kind
         resource-name -n namespace

# View resource status conditions
kubectl get resource-kind
         resource-name -n namespace -o jsonpath='{.status.conditions}'

# View events related to the resource
kubectl get events --field-selector involvedObject.name=resource-name -n namespace
```

Die Bedingungen für den Ressourcenstatus geben Aufschluss über:

- Ob die Ressource bereit ist
- Alle aufgetretenen Fehler
- Aktueller Stand der Versöhnung

Überprüfen Sie die IAM-Berechtigungen und den Clusterzugriff

Viele Funktionsprobleme sind auf Probleme mit IAM-Berechtigungen oder eine fehlende Cluster-Zugriffskonfiguration zurückzuführen. Überprüfen Sie sowohl die Berechtigungen für die Capability Role als auch die Clusterzugriffseinträge.

Überprüfen Sie die IAM-Rollenberechtigungen

Stellen Sie sicher, dass die Capability-Rolle über die erforderlichen Berechtigungen verfügt:

```
# List attached managed policies
aws iam list-attached-role-policies --role-name my-capability-role

# List inline policies
aws iam list-role-policies --role-name my-capability-role

# Get specific policy details
aws iam get-role-policy --role-name my-capability-role --policy-name policy-name

# View the role's trust policy
aws iam get-role --role-name my-capability-role --query 'Role.AssumeRolePolicyDocument'
```

Die Vertrauensrichtlinie muss dem `capabilities.eks.amazonaws.com` Dienstprinzipal Folgendes ermöglichen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "capabilities.eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Überprüfen Sie die EKS-Zugriffseinträge und Zugriffsrichtlinien

Für alle Funktionen sind die richtigen EKS-Zugriffseinträge und -Zugriffsrichtlinien auf dem Cluster erforderlich, auf dem sie ausgeführt werden.

Stellen Sie sicher, dass der Zugriffseintrag vorhanden ist:

```
aws eks list-access-entries \
  --cluster-name my-cluster \
  --region region-code
```

Suchen Sie in der Liste nach dem ARN für die Capability Role. Fehlt die Fähigkeit, kann sie nicht auf den Cluster zugreifen.

Überprüfen Sie die dem Eintrag beigefügten Zugriffsrichtlinien:

```
aws eks list-associated-access-policies \
  --cluster-name my-cluster \
  --principal-arn arn:aws:iam::111122223333:role/my-capability-role \
  --region region-code
```

Für alle Funktionen sind entsprechende Zugriffsrichtlinien erforderlich:

- ACK: Benötigt Berechtigungen zum Erstellen und Verwalten von Kubernetes-Ressourcen
- kro: Benötigt Berechtigungen zum Erstellen und Verwalten von Kubernetes-Ressourcen
- Argo CD: Benötigt Berechtigungen zum Erstellen und Verwalten von Anwendungen und erfordert Zugriffseinträge auf Remote-Zielclustern für Bereitstellungen mit mehreren Clustern

Für Argo CD-Bereitstellungen mit mehreren Clustern:

Stellen Sie bei der Bereitstellung auf Remote-Clustern sicher, dass die Capability Role auf jedem Zielcluster über einen Zugriffseintrag verfügt:

```
# Check Access Entry on target cluster
aws eks describe-access-entry \
  --cluster-name target-cluster \
  --principal-arn arn:aws:iam::111122223333:role/argocd-capability-role \
  --region region-code
```

Wenn der Access-Eintrag auf einem Zielcluster fehlt, kann Argo CD keine Anwendungen darauf bereitstellen. Einzelheiten the section called “Registrieren Sie Cluster” zur Konfiguration finden Sie unter.

Funktionsspezifische Problembehebung

Eine ausführliche Anleitung zur Problembehebung, die für jeden Funktionstyp spezifisch ist, finden Sie unter:

- the section called “Fehlerbehebung”- Problembehandlung bei der Erstellung von ACK-Ressourcen, IAM-Berechtigungen und kontenübergreifendem Zugriff
- the section called “Fehlerbehebung”- Problembehebung bei der Anwendungssynchronisierung, Repository-Authentifizierung und Multi-Cluster-Bereitstellungen
- the section called “Fehlerbehebung”- Problembehandlung ResourceGraphDefinitions, CEL-Ausdrücke und RBAC-Berechtigungen

Häufig auftretende Probleme in allen Funktionen

Die Fähigkeit steckt im Status CREATING fest

Wenn eine Fähigkeit länger als erwartet im `CREATING` Status verbleibt:

1. Überprüfen Sie den Funktionsstatus auf bestimmte Probleme in der Konsole (Observability > Monitor Cluster > Registerkarte Capabilities) oder mithilfe der AWS CLI:

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-capability-name \
  --query 'capability.health'
```

2. Stellen Sie sicher, dass die IAM-Rolle existiert und über die richtige Vertrauensrichtlinie verfügt
3. Stellen Sie sicher, dass Ihr Cluster zugänglich und fehlerfrei ist
4. Suchen Sie nach Problemen auf Clusterebene, die die Einrichtung von Funktionen verhindern könnten

Ressourcen werden nicht erstellt oder aktualisiert

Wenn die Funktion aktiviert ist`ACTIVE`, aber Ressourcen nicht erstellt oder aktualisiert werden:

1. Überprüfen Sie den Ressourcenstatus auf Fehlerbedingungen

2. Überprüfen Sie die IAM-Berechtigungen für die spezifischen AWS Dienste (ACK) oder Repositorys (Argo CD)
3. Überprüfen Sie die RBAC-Berechtigungen für die Erstellung der zugrunde liegenden Ressourcen (kro)
4. Überprüfen Sie die Ressourcenspezifikationen auf Validierungsfehler

Der Zustand der Fähigkeiten weist Probleme auf

Es werden gesundheitliche Probleme `describe-capability` angezeigt:

1. Lesen Sie die Problembeschreibungen sorgfältig durch — sie weisen häufig auf das spezifische Problem hin
2. Gehen Sie der Ursache nach (IAM-Berechtigungen, Konfigurationsfehler usw.)
3. Die Funktion wird automatisch wiederhergestellt, sobald das Problem behoben ist

Nächste Schritte

- the section called “Mit Fähigkeiten arbeiten”- Ressourcen für Fähigkeiten verwalten
- the section called “Fehlerbehebung”- ACK-spezifische Fehlerbehebung
- the section called “Fehlerbehebung”- CD-spezifische Problembehebung für Argo
- the section called “Fehlerbehebung”- Kro-spezifische Problembehebung
- the section called “Überlegungen zu den Funktionen von EKS”- Bewährte Sicherheitsmethoden für Funktionen

Organisieren und Überwachen von Cluster-Ressourcen

In diesem Kapitel finden Sie die folgenden Themen, die Ihnen beim Verwalten Ihres Clusters helfen. Sie können auch Informationen über Ihre Kubernetes-Ressourcen mit der AWS-Managementkonsole anzeigen.

- Das Kubernetes-Dashboard ist eine allgemeine, webbasierte Benutzeroberfläche für Kubernetes-Cluster. Es ermöglicht Benutzern, im Cluster ausgeführte Anwendungen zu verwalten und Fehler zu beheben sowie den Cluster selbst zu verwalten. Weitere Informationen finden Sie unter Kubernetes-Dashboard im GitHub-Repository.
- the section called “Kennzahlen-Server”Der Kubernetes-Metrik-Server ist ein Aggregator für Ressourcenverbrauchsdaten in Ihrem Cluster. Sie wird nicht standardmäßig in Ihrem Cluster bereitgestellt, wird jedoch von Kubernetes-Add-Ons wie dem Kubernetes-Dashboard und the section called “Horizontal Pod Autoscaler” verwendet. In diesem Thema erfahren Sie, wie der Metrics Server installiert wird.
- the section called “Apps mit Helm bereitstellen”Der Helm-Paketmanager für Kubernetes unterstützt Sie bei der Installation und Verwaltung von Anwendungen in Ihrem Kubernetes-Cluster. Dieses Thema hilft Ihnen bei der Installation und Ausführung der Helm-Binärdateien, sodass Sie Charts mit der Helm-CLI auf Ihrem lokalen Computer installieren und verwalten können.
- the section called “Markieren Ihrer -Ressourcen” - Um Sie bei der Verwaltung Ihrer Amazon EKS-Ressourcen zu unterstützen, können Sie jeder Ressource eigene Metadaten in Form von Tags zuweisen. In diesem Thema werden Tags (Markierungen) und deren Erstellung beschrieben.
- the section called “Servicekontingente” - Das AWS-Konto verfügt über Standardkontingente (früher als Limits bezeichnet) für jeden AWS-Service. Erfahren Sie mehr über Amazon EKS-Ressourcenkontingente, und wie Sie sie erhöhen.

Überwachung und Optimierung der Kosten für Amazon-EKS-Cluster

Die Kostenüberwachung ist ein wesentlicher Aspekt der Verwaltung Ihrer Kubernetes-Cluster in Amazon EKS. Indem Sie Einblick in Ihre Cluster-Kosten erhalten, können Sie die Ressourcennutzung optimieren, Budgets festlegen und datenbasierte Entscheidungen zu Ihren Bereitstellungen treffen. Amazon EKS bietet zwei Lösungen zur Kostenüberwachung, jede mit ihren eigenen einzigartigen Vorteilen, die Ihnen dabei helfen, Ihre Kosten effektiv zu verfolgen und zuzuordnen:

Daten zur Aufteilung der AWS-Fakturierungskosten für Amazon EKS – Dieses native Feature lässt sich nahtlos in die AWS-Fakturierungskonsole integrieren, sodass Sie Kosten mithilfe derselben vertrauten Benutzeroberfläche und derselben Workflows analysieren und zuweisen können, die Sie auch für andere AWS-Service verwenden. Mit der aufgeteilten Kostenzuordnung erhalten Sie Einblicke in Ihre Kubernetes-Kosten neben Ihren anderen AWS-Ausgaben, wodurch es einfacher ist, die Kosten in Ihrer gesamten AWS-Umgebung ganzheitlich zu optimieren. Sie können auch vorhandene AWS-Fakturierung-Feature wie Kostenkategorien und Erkennung von Kostenanomalien nutzen, um Ihre Kostenmanagement-Fähigkeiten weiter zu verbessern. Weitere Informationen finden Sie unter Grundlagen zu Daten zur aufgeteilten Kostenzuordnung im Benutzerhandbuch zur AWS-Fakturierung.

Kubecost – Amazon EKS unterstützt Kubecost, ein Kubernetes-Kostenüberwachungstool. Kubecost bietet einen Feature-reichen, Kubernetes-nativen Ansatz für die Kostenüberwachung und stellt differenzierte Kostenaufschlüsselungen nach Kubernetes-Ressourcen, Empfehlungen zur Kostenoptimierung sowie sofort einsatzbereite Dashboards und Berichte zur Verfügung. Durch die Integration mit dem AWS-Kosten- und Nutzungsbericht ruft Kubecost außerdem genaue Preisdaten ab und stellt so sicher, dass Sie einen genauen Überblick über Ihre Amazon-EKS-Kosten erhalten. Weitere Informationen zur Installation von Kubecost. Informationen zum Erhalt eines kostenlosen Kubecost-Abonnements finden Sie auf der Seite zu Kubecost AWS Marketplace.

Kosten pro Pod in der AWS Abrechnung mit geteilter Kostenzuweisung anzeigen

Kostenüberwachung mithilfe von Daten zur AWS geteilten Kostenzuweisung für Amazon EKS

Sie können Daten zur AWS geteilten Kostenzuweisung für Amazon EKS verwenden, um eine detaillierte Kostentransparenz für Ihre Amazon EKS-Cluster zu erhalten. Auf diese Weise können Sie die Kosten und die Nutzung Ihrer Kubernetes-Anwendungen analysieren, optimieren und zurückbelasten. Sie weisen die Anwendungskosten einzelnen Geschäftseinheiten und Teams auf der Grundlage der EC2 CPU- und Speicherressourcen von Amazon zu, die von Ihrer Kubernetes-Anwendung verbraucht werden. Die aufgeteilten Kostenzuordnungsdaten für Amazon EKS bieten Einblick in die Kosten pro Pod und ermöglichen Ihnen die Aggregation der Kostendaten pro Pod mithilfe von Namespace, Cluster und anderen Kubernetes-Primitiven. Nachfolgend finden Sie Beispiele für Kubernetes-Primitive, die Sie zur Analyse der Amazon-EKS-Kostenzuordnungsdaten verwenden können.

- Clustername

- Bereitstellung
- Namespace
- Knoten
- Name der Workload
- Workload-Typ

Benutzerdefinierte Kostenzuordnungs-Tags werden ebenfalls unterstützt. Weitere Informationen zur Verwendung von Daten zur geteilten Kostenzuweisung finden Sie unter Grundlegendes zu Daten zur geteilten Kostenzuweisung im AWS Billing User Guide.

Kosten- und Nutzungsberichte einrichten

Sie können die Option Geteilte Kostenzuweisungsdaten für EKS in der Cost Management Console, in der AWS Befehlszeilenschnittstelle oder im aktivieren AWS SDKs.

Verwenden Sie für die Aufteilung der Kostenzuordnungsdaten Folgendes:

1. Melden Sie sich für die Aufteilung der Kostenzuordnungsdaten an. Weitere Informationen finden Sie unter Aktivieren von Daten zur geteilten Kostenzuweisung im Benutzerhandbuch AWS für Kosten- und Nutzungsberichte.
2. Fügen Sie die Daten in einen neuen oder vorhandenen Bericht ein.
3. Zeigen Sie den Bericht an. Sie können die Konsole Fakturierung und Kostenmanagement verwenden oder die Berichtsdateien in Amazon Simple Storage Service anzeigen.

Kubecost installieren

Amazon EKS unterstützt Kubecost, mit dem Sie Ihre Kosten aufgeschlüsselt nach Kubernetes-Ressourcen, einschließlich Pods, Knoten, Namespaces und Labels, überwachen können. Dieses Thema behandelt die Installation von Kubecost und den Zugriff auf das Kubecost-Dashboard.

Amazon EKS bietet ein AWS optimiertes Kubecost-Paket für die Transparenz der Cluster-Kosten. Sie können Ihre bestehenden AWS Supportverträge verwenden, um Support zu erhalten. Weitere Informationen zu den verfügbaren Versionen von Kubecost finden Sie unter the section called “Weitere Informationen zu Kubecost”.

> **Note**
>
> Kubecost v3 bietet wichtige architektonische Verbesserungen, darunter eine deutlich schnellere Leistung und erweiterte Automatisierungsfunktionen. Erfahren Sie mehr über Kubecost v3.
> Kubecost v2 führt mehrere wichtige neue Features ein. Weitere Informationen zu Kubecost v2.

Weitere Informationen zu Kubecost finden Sie in der Kubecost-Dokumentation und den Häufig gestellten Fragen.

Installation des Amazon-EKS-optimierten Kubecost-Pakets

Sie können eines der folgenden Verfahren verwenden, um das Amazon-EKS-optimierte Kubecost-Paket zu installieren:

- Es wird empfohlen, vor dem Start den Abschnitt Kubecost – Architekturübersicht zu lesen, um zu verstehen, wie Kubecost in Amazon EKS funktioniert.
- Wenn Sie Amazon EKS zum ersten Mal verwenden, empfehlen wir Ihnen, das Amazon-EKS-Add-On für die Installation zu verwenden, da es die Installation des Amazon-EKS-optimierten Kubecost-Pakets vereinfacht. Weitere Informationen finden Sie unter Bereitstellung von Kubecost in einem Amazon-EKS-Cluster mithilfe des Amazon-EKS-Add-Ons.
- Um die Installation anzupassen, können Sie Ihr Amazon-EKS-optimiertes Kubecost-Paket mit Helm konfigurieren. Weitere Informationen finden Sie unter Bereitstellung von Kubecost in einem Amazon-EKS-Cluster mithilfe von Helm in der Kubecost-Dokumentation.

> **Important**
>
> Für Kubecost v3 wurde die Position des Helm-Diagramms auf geändert. `public.ecr.aws/kubecost/kubecost` Wenn Sie ein Upgrade von Version 2 durchführen, aktualisieren Sie Ihre Helm-Repository-Referenzen entsprechend.

> **Note**
>
> Für Multi-Cluster-Bereitstellungen mit Kubecost v3 benötigen Sie S3-kompatiblen Objektspeicher (AWS S3 für EKS-Kunden) für die Speicherung von Metriken. Dies ersetzt den in Version 2 verwendeten Prometheus-kompatiblen Speicher. Weitere Informationen finden Sie unter Multi-Cluster-Installation in der Kubecost-Dokumentation.

Zugriff auf das Kubecost-Dashboard

Sobald die Einrichtung des Amazon-EKS-optimierten Kubecost-Pakets abgeschlossen ist, sollten Sie Zugriff auf das Kubecost-Dashboard haben. Weitere Informationen finden Sie unter the section called “Zugriff auf das Kubecost-Dashboard”.

Zugriff auf das Kubecost-Dashboard

Voraussetzungen

1. Stellen Sie sicher, dass sich die zugehörigen Pods im Status „Ausgeführt“ befinden.

```
kubectl get pods --namespace kubecost
```

Zugriff auf das Kubecost-Dashboard

1. Aktivieren Sie auf Ihrem Gerät die Port-Weiterleitung, um das Kubecost-Dashboard anzuzeigen.
   - Wenn kubecost v3 mit helm installiert ist:

     ```
     kubectl port-forward deployment/kubecost-frontend 9090 --namespace kubecost
     ```

   - Wenn kubecost v1 oder v2 mit helm installiert ist:

     ```
     kubectl port-forward deployment/kubecost-cost-analyzer 9090 --namespace kubecost
     ```

   - Wenn kubecost mit dem Amazon EKS-Add-on installiert wurde:

     ```
     kubectl port-forward deployment/cost-analyzer 9090 --namespace kubecost
     ```

Alternativ können Sie den AWS Load Balancer Controller verwenden, um Kubecost verfügbar zu machen und Amazon Cognito für die Authentifizierung, Autorisierung und Benutzerverwaltung zu verwenden. Weitere Informationen finden Sie unter Verwendung des Application Load Balancer und Amazon Cognito, um Benutzer für Ihre Kubernetes-Web-Apps zu authentifizieren.

2. Öffnen Sie auf demselben Gerät, auf dem Sie den vorherigen Schritt ausgeführt haben, einen Webbrowser und geben Sie die folgende Adresse ein.

```
http://localhost:9090
```

   Die Kubecost-Übersichtsseite wird in Ihrem Browser angezeigt. Es kann 5–10 Minuten (oder länger) dauern, bis Kubecost Metriken erfasst hat, abhängig von der Größe Ihres Clusters. Sie können Ihre Amazon-EKS-Ausgaben anzeigen, einschließlich der kumulierten Cluster-Kosten, der zugehörigen Kubernetes-Komponenten-Kosten und der monatlichen Gesamtkosten.

3. Um die Kosten auf Clusterebene zu verfolgen, kennzeichnen Sie Ihre Amazon-EKS-Ressourcen für die Abrechnung mit Tags. Weitere Informationen finden Sie unter the section called “Markieren von Ressourcen für die Fakturierung”.
   - Cost allocation (Kostenzuordnung) – Zeigen Sie die monatlichen Amazon-EKS-Kosten und kumulierten Kosten für jeden Ihrer Namespaces und andere Dimensionen der letzten sieben Tage an. Dies ist hilfreich, um zu verstehen, welche Teile Ihrer Anwendung zu den Amazon-EKS-Ausgaben beitragen.
   - Anlagen — Sehen Sie sich die Kosten der AWS Infrastrukturressourcen an, die Ihren Amazon EKS-Ressourcen zugeordnet sind.

Weitere Informationen zu Kubecost

Amazon EKS bietet ein AWS optimiertes Kubecost-Paket für die Transparenz der Cluster-Kosten. Amazon EKS unterstützt Kubecost, mit dem Sie Ihre Kosten aufgeschlüsselt nach Kubernetes-Ressourcen, einschließlich Pods, Knoten, Namespaces und Labels, überwachen können.

Dieses Thema behandelt die verfügbaren Versionen von Kubecost und die Unterschiede zwischen den verfügbaren Stufen. EKS unterstützt Kubecost Version 1, Version 2 und Version 3. Jede Version ist in verschiedenen Stufen verfügbar. Sie können das für Amazon EKS optimierte Kubecost-Paket ohne zusätzliche Kosten für Ihre Amazon-EKS-Cluster verwenden. Möglicherweise wird Ihnen die Nutzung verbundener AWS Dienste wie Amazon Managed Service für Prometheus in Rechnung

gestellt. Sie können auch Ihre bestehenden AWS Supportverträge verwenden, um Support zu erhalten.

Als Kubernetes-Plattformadministrator und Finanzleitung können Sie Kubecost verwenden, um eine Aufschlüsselung der Amazon-EKS-Gebühren zu visualisieren, Kosten zuzuweisen und Organisationseinheiten wie Anwendungsteams Gebühren zu berechnen. Sie können Ihren internen Teams und Geschäftseinheiten transparente und genaue Kostendaten zur Verfügung stellen, die auf der tatsächlichen AWS Abrechnung basieren. Darüber hinaus können Sie angepasste Empfehlungen für die Kostenoptimierung erhalten, die auf der Infrastrukturumgebung und den Nutzungsmustern innerhalb ihrer Cluster basieren. Weitere Informationen zu Kubecost finden Sie in der Kubecost-Dokumentation.

Was ist der Unterschied zwischen dem benutzerdefinierten Paket von Kubecost und der kostenlosen Version von Kubecost (auch bekannt als)? OpenCost

AWS und Kubecost haben zusammengearbeitet, um eine maßgeschneiderte Version von Kubecost anzubieten. Diese Version enthält eine Teilmenge kommerzieller Features ohne zusätzliche Kosten. In den folgenden Tabellen finden Sie die Feature, die im benutzerdefinierten Paket von Kubecost enthalten sind.

Kubecost v3

Was ist der Unterschied zwischen Kubecost v2 und v3?

Kubecost 3.0 ist ein wichtiges Architektur-Upgrade, das eine deutlich schnellere Leistung, verbesserte Skalierbarkeit und proaktive Optimierungsmöglichkeiten bietet. Die bedeutendste Änderung ist die Migration zu einer ClickHouse Datenbank, die DuckDB aus Version 2.8 ersetzt, die wesentlich schnellere Abfragen und eine zuverlässigere Leistung im großen Maßstab bietet. Kubecost 3.0 führt außerdem einen einheitlichen Agenten ein, der Kubecost- und Cloudability-Funktionen kombiniert, wodurch die Prometheus-Abhängigkeit entfällt und der Speicherbedarf reduziert wird, während die Kompatibilität gewahrt bleibt. OpenCost

> **Important**
>
> Lesen Sie die Kubecost-Dokumentation, bevor Sie auf Version 3 aktualisieren. Die Migration von Version 2 erfordert eine sorgfältige Planung und kann sich auf die Verfügbarkeit von Berichten während der Umstellung auswirken. Die Position der Helm-Karte wurde geändert auf`public.ecr.aws/kubecost/kubecost`.

Wichtigste architektonische Verbesserungen in Version 3:

- ClickHouse Datenbank: Ersetzt DuckDB für deutlich schnellere Abfragen und bessere Skalierbarkeit
- Unified Agent: Kombiniert Kubecost- und Cloudability-Funktionen und macht so die Abhängigkeit von Prometheus überflüssig
- S3-kompatibler Speicher für Multi-Cluster: Für Multi-Cluster-Bereitstellungen verwendet v3 S3-kompatiblen Objektspeicher (AWS S3 für EKS-Kunden) anstelle von Prometheus kompatiblem Speicher wie Amazon Managed Service für Prometheus. Der FinOps Agent ruft Metriken aus der Kubernetes-API ab und überträgt sie in einen S3-kompatiblen Speicher. Anschließend ruft der Aggregator diese Daten ab, führt Ableitungsschritte durch und zeigt die Ergebnisse im Frontend an. Weitere Informationen finden Sie im Leitfaden zur Installation mehrerer Cluster und sekundärer Cluster in der Kubecost-Dokumentation.
- Reduzierter Speicherbedarf: Deutlich geringerer Ressourcenbedarf bei gleichbleibender Funktionalität
- Vereinfachte Architektur: Einzelcontainer-Pod-Topologie für unabhängige Skalierung und verbesserte Ausfallsicherheit
- Verbesserte Automatisierung: Automatisierte Dimensionierung von Container-Anfragen mit Erkennung mehrerer Cluster und benutzerdefinierten Profilen

Vorteile des für Amazon EKS optimierten Pakets in Version 3:

Das für Amazon EKS optimierte Kubecost-Paket ist weiterhin ohne zusätzliche Kosten erhältlich und von dem neuen Ausgabenlimit von 100.000 USD ausgenommen, das im kostenlosen Kontingent von Kubecost v3 eingeführt wurde. EKS-Benutzer behalten unabhängig von der Höhe der Ausgaben vollen Zugriff auf alle Kubernetes-Ausgabenfunktionen.

Vergleich der Haupt-Features:

Feature	Kubecost — kostenlos es Kontingent 3.0	Amazon EKS-optim iertes Kubecost Bundle 3.0	Kubecost Enterprise 3.0
Transparenz der Cluster-Kosten	Unbegrenzte Anzahl von Clustern, begrenzt auf Ausgaben in Höhe	Einheitlicher Multi-Clu ster ohne Ausgabenl imits	Einheitliche und unbegrenzte Anzahl von Clustern in einer unbegrenzten Anzahl

Feature	Kubecost — kostenlos es Kontingent 3.0	Amazon EKS-optim iertes Kubecost Bundle 3.0	Kubecost Enterprise 3.0
	von 100.000 USD über einen Zeitraum von 30 Tagen		von Umgebungen (d. h. Multi-Cloud)
Datenbank-Backend	ClickHouse (lokal)	ClickHouse mit S3-kompatiblem Speicher für Multi-Cluster-Metr iken	ClickHouse mit benutzerdefinierten Datenbankoptionen
Leistung	Erheblich schnellere Abfragen im Vergleich zu Version 2	Erheblich schnellere Abfragen im Vergleich zu v2	Erheblich schnellere Abfragen im Vergleich zu v2
Speicherbedarf	Reduziert gegenüber v2 (keine Prometheu s-Abhängigkeit)	Reduziert gegenüber v2 (keine Prometheu s-Abhängigkeit)	Reduziert gegenüber v2 (keine Prometheu s-Abhängigkeit)
Automatisierte Dimensionierung von Container-Anfragen	Verfügbar (limitiert auf 250 Kerne)	Ohne Kernbesch ränkungen erhältlich	Ohne Kernbesch ränkungen erhältlich
Ausgabenlimits	100.000 USD über 30 Tage	Keine Ausgabenlimits	Keine Ausgabenb eschränkungen
Automatisierung mehrerer Cluster	Begrenzt	Umfassender Überblick über mehrere Cluster mit sicherem Messaging	Umfassendes Bewusstsein für mehrere Cluster mit sicherem Messaging

Kubecost v2

Was ist der Unterschied zwischen Kubecost v1 und v2?

Kubecost 2.0 stellt eine wesentliche Verbesserung gegenüber früheren Versionen dar und enthält wichtige neue Feature, darunter ein brandneues API-Backend. Beachten Sie, dass Allocation und

Assets vollständig abwärtskompatibel APIs sind. Lesen Sie die Kubecost-Dokumentation, um einen problemlosen Übergang zu gewährleisten. Die vollständige Liste der Verbesserungen finden Sie in der Ankündigung zu Kubecost v2.0 und den vollständigen Versionshinweisen.

> **Important**
>
> Lesen Sie vor dem Upgrade die Kubecost-Dokumentation. Durch ein Upgrade kann die Berichtsverfügbarkeit beeinträchtigt werden.

Vergleich der Haupt-Features:

Feature	Kostenloses Kontingent Kubecost 2.0	Amazon-EKS-optimiertes Kubecost-Paket 2.0	Kubecost Enterprise 2.0
Transparenz der Cluster-Kosten	Unbegrenzte Cluster mit bis zu 250 Kernen	Einheitlicher Multi-Cluster ohne Kernbeschränkungen bei Integration mit Amazon Managed Service für Prometheus	Einheitliche und unbegrenzte Anzahl von Clustern in einer unbegrenzten Anzahl von Umgebungen (d. h. Multi-Cloud)
Bereitstellung	Vom Benutzer gehostet	Vom Benutzer gehostet	Vom Benutzer gehostet, von Kubecost gehostet (dedizierter Mandant), SaaS
Unterstützte Datenbanken	Lokales Prometheus	Amazon Managed Service für Prometheus oder lokales Prometheus	Alle Prometheus-Varianten und benutzerdefinierten Datenbanken
Unterstützung für die Datenbank-Aufbewahrung (Rohmetriken)	15 Tage	Unbegrenzte historische Daten	Unbegrenzte historische Daten

Feature	Kostenloses Kontingent Kubecost 2.0	Amazon-EKS-optimiertes Kubecost-Paket 2.0	Kubecost Enterprise 2.0
Kubecost-API und UI-Aufbewahrung (ETL)	15 Tage	15 Tage	Unbegrenzt
Sichtbarkeit in der Hybrid Cloud	-	Amazon EKS und Amazon-EKS-Anywhere-Cluster	Multi-Cloud und Hybrid-Cloud
Benachrichtigungen und wiederkehrende Berichte	Wird nur auf dem primären Cluster unterstützt, begrenzt auf 250 Kerne	Effizienzwarnungen, Budgetwarnungen, Warnungen bei Ausgabenänderungen und mehr werden unterstützt in allen Clustern	Effizienzwarnungen, Budgetwarnungen, Warnungen bei Ausgabenänderungen und mehr werden unterstützt in allen Clustern
Gespeicherte Berichte	-	Berichte mithilfe von Metriken der letzten 15 Tage	Berichte mithilfe unbegrenzter Verlaufsdaten und Metriken
Integration der Cloud-Abrechnung	Wird nur auf dem primären Cluster unterstützt, begrenzt auf 250 Kerne	Unterstützung benutzerdefinierter Preisgestaltung für AWS (einschließlich mehrerer Cluster und mehrerer Konten)	Benutzerdefinierte Preisgestaltung für jede Cloud
Empfehlungen für Einsparungen	Wird nur auf dem primären Cluster unterstützt, begrenzt auf 250 Kerne	Primäre Cluster-Erkenntnisse, aber es gibt keine 250-Kern-Grenze	Erkenntnisse aus mehreren Clustern
Unternehmensführung: Prüfungen	-	-	Prüfen Sie historische Kostenereignisse

Feature	Kostenloses Kontingent Kubecost 2.0	Amazon-EKS-optimiertes Kubecost-Paket 2.0	Kubecost Enterprise 2.0
Unterstützung von Single Sign-On (SSO)	-	Unterstützung von Amazon Cognito	Okta, Auth0, PingID und alles KeyCloak andere benutzerdefinierte
Rollenbasierte Zugriffskontrolle (RBAC) mit SAML 2.0	-	-	Okta, Auth0, PingID und alles andere benutzerdefinierte KeyCloak
Training und Onboarding für Unternehmen	-	-	Schulung und Onboarding mit umfassendem Service FinOps
Teams	-	-	Ja

Neue Funktionen:

Für die folgenden Features gelten metrische Beschränkungen:

- Kubecost Aggregator
- Netzwerküberwachung
- Kubecost-Aktionen
- Sammlungen
- Anomalie-Erkennung
- Richtige Dimensionierung von Containeranfragen
- Kubecost-Prognose
- Automatische Vervollständigung für Filterung und Aggregation

Metrische Grenzwerte:

Metrik	Kostenloses Kontingent Kubecost 2.0	Amazon-EKS-optimiertes Kubecost-Paket 2.0	Kubecost Enterprise 2.0
Cluster-Größe	Unbegrenzte Cluster mit bis zu 250 Kernen	Unbegrenzt	Unbegrenzt
Beibehaltung von Metriken	15 Tage	15 Tage	Unbegrenzt
Multi-Cluster-Support	Nicht verfügbar	Available (Verfügbar)	Available (Verfügbar)
Kernbegrenzungen	250 Kerne pro Cluster	Keine Kernbegrenzungen	Keine Kernbegrenzungen

Kubecost v1

Feature	Kostenloses Kontingent für Kubecost	Amazon-EKS-optimiertes Kubecost-Paket	Kubecost Enterprise
Bereitstellung	Vom Benutzer gehostet	Vom Benutzer gehostet	Vom Benutzer gehostet oder von Kubecost gehostet (SaaS)
Anzahl der unterstützten Cluster	Unbegrenzt	Unbegrenzt	Unbegrenzt
Unterstützte Datenbanken	Lokales Prometheus	Lokales Prometheus oder Amazon Managed Service für Prometheus	Prometheus, Amazon Managed Service für Prometheus, Cortex oder Thanos
Unterstützung für die Aufbewahrung von Datenbanken	15 Tage	Unbegrenzte historische Daten	Unbegrenzte historische Daten

Feature	Kostenloses Kontingent für Kubecost	Amazon-EKS-optimiertes Kubecost-Paket	Kubecost Enterprise
Kubecost-API-Aufbewahrung (ETL)	15 Tage	15 Tage	Unbegrenzte historische Daten
Transparenz der Cluster-Kosten	Einzelne Cluster	Einheitliches Multi-Cluster	Einheitliches Multi-Cluster
Sichtbarkeit in der Hybrid Cloud	-	Amazon EKS und Amazon-EKS-Anywhere-Cluster	Multi-Cloud- und Hybrid-Cloud-Unterstützung
Benachrichtigungen und wiederkehrende Berichte	-	Unterstützung von Effizienzwarnungen, Budgetwarnungen, Ausgabenänderungswarnungen und mehr	Unterstützung von Effizienzwarnungen, Budgetwarnungen, Ausgabenänderungswarnungen und mehr
Gespeicherte Berichte	-	Berichte mit Daten aus 15 Tagen	Berichte mit unbegrenzten historischen Daten
Integration der Cloud-Abrechnung	Für jeden einzelnen Cluster erforderlich	Unterstützung bei kundenspezifischen Preisen für AWS (einschließlich mehrerer Cluster und mehrerer Konten)	Unterstützung bei kundenspezifischen Preisen für AWS (einschließlich mehrerer Cluster und mehrerer Konten)
Empfehlungen für Einsparungen	Erkenntnisse aus einzelnen Clustern	Erkenntnisse aus einzelnen Clustern	Erkenntnisse aus mehreren Clustern
Unternehmensführung: Prüfungen	-	-	Prüfen Sie historische Kostenereignisse

Feature	Kostenloses Kontingent für Kubecost	Amazon-EKS-optimiertes Kubecost-Paket	Kubecost Enterprise
Unterstützung von Single Sign-On (SSO)	-	Unterstützung von Amazon Cognito	Okta, Auth0, PingID, KeyCloak
Rollenbasierte Zugriffskontrolle (RBAC) mit SAML **2.0**	-	-	Okta, Auth0, PingID, Keycloak
Training und Onboarding für Unternehmen	-	-	Schulung und FinOps Onboarding mit umfassendem Service

Häufig gestellte Fragen

Lesen Sie die folgenden häufig gestellten Fragen und Antworten zur Verwendung von Kubecost mit Amazon EKS.

Was ist das Kubecost-API-Feature zur Datenaufbewahrung (ETL)?

Das Kubecost-ETL-Feature aggregiert und organisiert Kennzahlen, um Kostentransparenz auf verschiedenen Granularitätsebenen zu gewährleisten (z. B. `namespace-level`, `pod-level` und `deployment-level`). Für das Amazon-EKS-optimierte Kubecost-Paket erhalten Kunden Daten und Erkenntnisse aus Metriken der letzten 15 Tage.

Was ist die Funktion für Benachrichtigungen und wiederkehrende Berichte? Welche Benachrichtigungen und Berichte beinhaltet sie?

Mithilfe von Kubecost-Warnmeldungen können Teams in Echtzeit Aktualisierungen zu den Ausgaben für Kubernetes und die Cloud erhalten. Wiederkehrende Berichte ermöglichen es Teams, individuelle Ansichten der historischen Kubernetes- und Cloud-Ausgaben zu erhalten. Beide sind über die Kubecost-Benutzeroberfläche oder Helm-Werte konfigurierbar. Sie unterstützen E-Mail, Slack und Microsoft Teams.

Was beinhalten gespeicherte Berichte?

Gespeicherte Kubecost-Berichte sind vordefinierte Ansichten von Kosten- und Effizienzkennzahlen. Sie beinhalten die Kosten nach Cluster, Namespace, Label und mehr.

Was ist Cloud-Abrechnungsintegration?

Durch die Integration mit der AWS Abrechnung APIs kann Kubecost out-of-cluster Kosten anzeigen (z. B. Amazon S3). Darüber hinaus ermöglicht es Kubecost den Abgleich von Kubecost's In-Cluster-Prognosen mit den tatsächlichen Abrechnungsdaten, um Spotnutzung, Savings Plans und Unternehmensrabatte zu berücksichtigen.

Was beinhalten Sparempfehlungen?

Kubecost bietet Einblicke und Automatisierung, um Benutzern bei der Optimierung ihrer Kubernete-Infrastruktur und Ausgaben.

Ist diese Funktion kostenpflichtig?

Nein. Sie können das Amazon-EKS-optimierte Kubecost-Paket ohne zusätzliche Kosten nutzen. Wenn Sie zusätzliche Kubecost-Funktionen wünschen, die nicht enthalten sind, können Sie eine Enterprise-Lizenz von Kubecost über den AWS Marketplace oder direkt bei Kubecost erwerben.

Ist Support für das Amazon-EKS-optimierte Kubecost-Paket verfügbar?

Ja, nur wenn Sie das Amazon-EKS-optimierte Kubecost-Paket verwenden.

Wie erhalte ich Support für das Amazon-EKS-optimierte Kubecost-Paket?

Sie können einen Support-Fall beim AWS Support-Team unter Kontakt eröffnen AWS.

Benötige ich eine Lizenz, um die von der Amazon-EKS-Integration bereitgestellten Kubecost-Feature zu nutzen?

Nein.

Kann ich Kubecost in den AWS Kosten- und Nutzungsbericht integrieren, um genauere Berichte zu erhalten?

Ja. Sie können Kubecost so konfigurieren, dass Daten aus dem AWS Kosten- und Nutzungsbericht aufgenommen werden, um eine genaue Kostentransparenz zu erhalten, einschließlich Rabatte, Spot-Preise, Preise für Reserved Instances und andere. Weitere Informationen finden Sie unter AWS - Cloud-Fakturierung-Integration in der Kubecost-Dokumentation.

Unterstützt diese Version das Kostenmanagement von selbstverwalteten Kubernetes-Clustern auf Amazon? EC2

Nein. Das Amazon-EKS-optimierte Kubecost-Paket ist nur mit Amazon-EKS-Clustern kompatibel.

Kann Kubecost die Kosten für Amazon EKS auf AWS Fargate verfolgen?

Kubecost bietet beste Möglichkeiten, die Cluster-Kosten für Amazon EKS auf Fargate sichtbar zu machen, jedoch mit geringerer Genauigkeit als mit Amazon EKS auf Amazon. EC2 Dies ist hauptsächlich auf den Unterschied zurückzuführen, wie Ihnen Ihre Nutzung in Rechnung gestellt wird. Mit Amazon EKS in Fargate werden Ihnen die verbrauchten Ressourcen in Rechnung gestellt. Mit Amazon EKS auf EC2 Amazon-Knoten werden Ihnen die bereitgestellten Ressourcen in Rechnung gestellt. Kubecost berechnet die Kosten eines EC2 Amazon-Knotens auf der Grundlage der Knotenspezifikation, die CPU, RAM und kurzlebigen Speicher umfasst. Mit Fargate werden die Kosten basierend auf den angeforderten Ressourcen für die Fargate-Pods berechnet.

Wie erhalte ich Aktualisierungen und neue Versionen von Kubecost?

Sie können Ihre Kubecost-Version mithilfe der standardmäßigen Helm-Upgrade-Verfahren aktualisieren. Für Kubecost v3 sind die neuesten Versionen im neuen Helm-Diagramm verfügbar. `public.ecr.aws/kubecost/kubecost` Frühere Versionen (v1 und v2) sind weiterhin in der Amazon ECR Public Gallery verfügbar.

> **Important**
>
> Beachten Sie beim Upgrade auf Kubecost v3, dass sich die Position des Helm-Diagramms von zu geändert hat. `public.ecr.aws/kubecost/cost-analyzer` `public.ecr.aws/kubecost/kubecost` Aktualisieren Sie Ihre Helm-Repository-Referenzen entsprechend.

Wird die **`kubectl-cost`**-CLI unterstützt? Wie installiere ich diese?

Ja. `Kubectl-cost` ist ein Open-Source-Tool von Kubecost (Apache 2.0-Lizenz), das CLI-Zugriff auf Kubernetes-Kennzahlen zur Kostenzuordnungen bietet. Informationen zur Installation `kubectl-cost` finden Sie unter Installation auf GitHub.

Wird die Kubecost-Benutzeroberfläche unterstützt? Wie greife ich darauf zu?

Kubecost bietet ein Web-Dashboard, auf das Sie über `kubectl`-Portweiterleitung, eine Eingangsregel oder einen Load Balancer zugreifen können. Sie können auch den Load AWS Balancer Controller verwenden, um Kubecost verfügbar zu machen, und Amazon Cognito für

Authentifizierung, Autorisierung und Benutzerverwaltung verwenden. Weitere Informationen finden Sie im Blog unter So verwenden Sie Application Load Balancer und Amazon Cognito zur Authentifizierung von Benutzern für Ihre Kubernetes-Web-Apps. AWS

Wirkt sich das neue Ausgabenlimit von 100.000$ in Kubecost v3 auf Amazon EKS-Benutzer aus?

Nein. Das mit dem kostenlosen Kontingent Kubecost v3 eingeführte Ausgabenlimit von 100.000 USD über 30 Tage gilt nicht für Nutzer des Kubecost-Bundles, die für Amazon EKS optimiert sind. EKS-Benutzer behalten unabhängig von der Höhe der Ausgaben vollen Zugriff auf alle Kubernetes-Ausgabenfunktionen.

Was sind die wichtigsten Leistungsverbesserungen in Kubecost v3?

Kubecost v3 bietet erhebliche Leistungsverbesserungen durch sein ClickHouse Datenbank-Backend, das im Vergleich zu der in Version 2.8 verwendeten DuckDB erheblich schnellere Abfragen ermöglicht. Darüber hinaus beseitigt die einheitliche Agentenarchitektur die Abhängigkeit von Prometheus, wodurch der Speicherbedarf reduziert wird und gleichzeitig die volle Funktionalität und OpenCost Kompatibilität erhalten bleibt.

Welches Speicher-Backend verwendet Kubecost v3 für Bereitstellungen mit mehreren Clustern?

Kubecost v3 verwendet S3-kompatiblen Objektspeicher (AWS S3 für EKS-Kunden) für die Speicherung von Metriken in mehreren Clustern und ersetzt damit den in Version 2 verwendeten Prometheus-kompatiblen Speicher. Der FinOps Agent sammelt Metriken aus der Kubernetes-API und überträgt sie in einen S3-kompatiblen Speicher. Der Aggregator ruft dann diese Daten ab, führt Kostenberechnungen durch und zeigt die Ergebnisse im Frontend an. Ausführliche Anweisungen zur Einrichtung mehrerer Cluster finden Sie im Leitfaden zur Installation mehrerer Cluster und sekundärer Cluster in der Kubecost-Dokumentation.

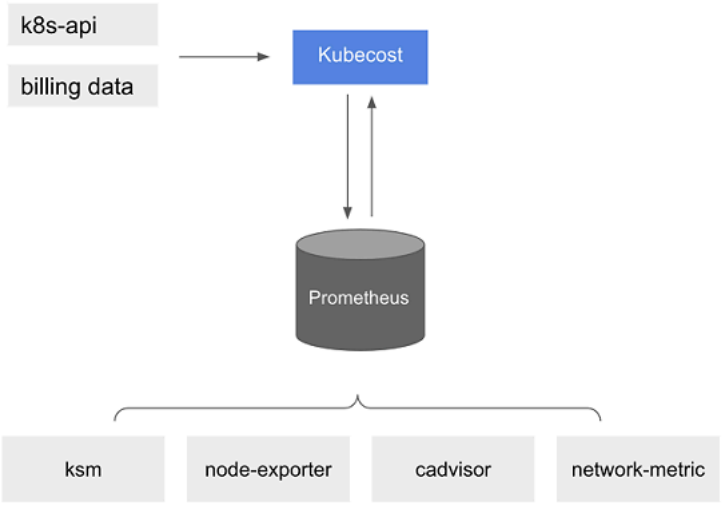
Kann ich direkt von Kubecost v1 auf v3 aktualisieren?

Nein. Ein direktes Upgrade von Version 1 auf Version 3 wird nicht unterstützt. Sie müssen zuerst auf Version 2 und dann auf Version 3 migrieren. Ausführliche Hinweise zur Migration finden Sie in der Kubecost-Dokumentation, da der Prozess eine sorgfältige Planung erfordert und sich auf die Verfügbarkeit von Berichten während der Umstellung auswirken kann.

Zusätzliche Kubecost-Feature

- Die folgenden Funktionen sind in Kubecost v1, v2 und v3 verfügbar.
  - Kostenkennzahlen exportieren — Die optimierte Kostenüberwachung von Amazon EKS wird mit Kubecost bereitgestellt. In Version 1 und Version 2 ist Kubecost in Prometheus integriert, um

Metriken zu speichern und zu verarbeiten. In Version 3 verwendet Kubecost eine ClickHouse Datenbank, um die Leistung erheblich zu verbessern und gleichzeitig die Kompatibilität aufrechtzuerhalten. OpenCost Bei Multi-Cluster-Bereitstellungen in Version 3 werden Metriken in S3-kompatiblem Objektspeicher (AWS S3 für EKS-Kunden) statt in Prometheus-kompatiblem Speicher gespeichert. Kubecost liest Metriken, führt Berechnungen zur Kostenzuweisung durch und stellt Daten über seine Benutzeroberfläche bereit. APIs Die Architektur variiert je nach Version, bietet jedoch eine konsistente Funktionalität.



Sie können Abfragen schreiben, um Kubecost-Daten zur weiteren Analyse in Ihr aktuelles Business Intelligence-System aufzunehmen. Sie können es auch als Datenquelle für Ihr aktuelles Grafana-Dashboard verwenden, mit dem Ihre internen Teams vertraut sind, um die Amazon-EKS-Cluster-Kosten anzuzeigen. Weitere Informationen zum Schreiben von Abfragen finden Sie in der OpenCost Konfigurationsdokumentation oder verwenden Sie die Grafana-JSON-Beispielmodelle im Kubecost Github-Repository als Referenz.

- AWS Integration von Kosten- und Nutzungsberichten — Um Berechnungen der Kostenzuweisung für Ihren Amazon EKS-Cluster durchzuführen, ruft Kubecost die öffentlichen Preisinformationen für AWS Dienste und AWS Ressourcen aus der AWS Preislisten-API ab. Sie können Kubecost auch in den AWS Kosten- und Nutzungsbericht integrieren, um die Genauigkeit der für Ihr Konto spezifischen Preisinformationen zu verbessern. AWS Zu diesen Informationen gehören Rabattprogramme für Unternehmen, Nutzung von reservierten Instances, Savings Plans und Spot-Nutzung. Weitere Informationen zur Funktionsweise der Integration von AWS Kosten- und Nutzungsberichten finden Sie unter AWS Cloud Billing Integration in der Kubecost-Dokumentation.

Anzeigen der Ressourcennutzung mit Kubernetes Metrics Server

Kubernetes Metrics Server ist ein Aggregator für Daten zur Ressourcennutzung in Ihrem Cluster und wird standardmäßig nicht in Amazon-EKS-Clustern bereitgestellt. Weitere Informationen finden Sie unter Kubernetes Metrics Server auf. GitHub Der Metriken-Server wird häufig von anderen Kubernetes-Add-Ons verwendet, beispielsweise für die Skalierung von Pod-Bereitstellungen mit Horizontal Pod Autoscaler oder für das Kubernetes-Dashboard. Weitere Informationen finden Sie unter Ressourcenmetrikpipeline in der Kubernetes-Dokumentation. In diesem Thema wird erläutert, wie Sie den Kubernetes-Metrik-Server in Ihrem Amazon EKS-Cluster bereitstellen.

> **Important**
>
> Die Metriken sind für point-in-time Analysen bestimmt und stellen keine genaue Quelle für historische Analysen dar. Sie können nicht als Überwachungslösung oder für andere Zwecke verwendet werden, die nichts mit Auto Scaling zu tun haben. Weitere Informationen zu Überwachungstools finden Sie unter Cluster überwachen.

Überlegungen

- Wenn Sie Kubernetes Metrics Server mithilfe des Manifests manuell auf Fargate-Knoten bereitstellen, konfigurieren Sie die `metrics-server`-Bereitstellung so, dass ein anderer Port als der Standard-Port `10250` verwendet wird. Dieser Port ist für Fargate reserviert. Die Amazon-EKS-Add-On-Version des Metriken-Servers ist für die Verwendung von Port `10251` vorkonfiguriert.
- Stellen Sie sicher, dass Sicherheitsgruppen und Netzwerk `10250` den Port zwischen den `metrics-server` Pods und allen anderen Knoten und Pods ACLs zulassen. Kubernetes Metrics Server verwendet weiterhin Port `10250`, um Metriken von anderen Endpunkten im Cluster zu erfassen. Wenn Sie auf Fargate-Knoten bereitstellen, lassen Sie sowohl den konfigurierten alternativen Metrik-Server-Port als auch den Port `10250` zu.

Bereitstellung als Community-Add-On mit Amazon-EKS-Add-Ons

Neu: Sie können Metrics Server jetzt über die AWS Konsole oder Amazon EKS als Community-Add-on bereitstellen APIs.

Mit der AWS Konsole bereitstellen

1. Öffnen Sie Ihren EKS-Cluster in der AWS Konsole
2. Wählen Sie auf der Registerkarte „Add-Ons“ die Option Weitere Add-Ons abrufen.
3. Wählen Sie im Abschnitt „Community-Add-Ons“ die Option Metrik-Server und anschließend Weiter
4. EKS ermittelt die geeignete Version des Add-Ons für Ihren Cluster. Sie können die Version über das Dropdown-Menü Version ändern.
5. Wählen Sie Weiter und anschließend Erstellen, um das Add-On zu installieren.

Weitere Ressourcen

Weitere Informationen zu the section called “Community-Erweiterungen”.

Sie installieren oder aktualisieren Community-Add-Ons auf dieselbe Weise wie andere Amazon-EKS-Add-Ons.

- the section called “Add-On erstellen”
- the section called “Add-On aktualisieren”
- the section called “Add-On entfernen”

Bereitstellung mit Manifest

Neu: Sie können Metrics Server jetzt über die AWS Konsole oder Amazon EKS als Community-Add-on bereitstellen APIs. Diese Installationsanweisungen für Manifeste werden archiviert.

1. Stellen Sie den Metrics Server mit dem folgenden Befehl bereit:

   ```
   kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml
   ```

   Wenn Sie Fargate verwenden, müssen Sie diese Datei ändern. In der Standardkonfiguration verwendet der Metrikserver den Port 10250. Dieser Port ist auf Fargate reserviert. Ersetzen Sie Verweise auf Port 10250 in components.yaml durch einen anderen Port, beispielsweise 10251.
2. Überprüfen Sie mit dem folgenden Befehl, ob die `metrics-server`-Bereitstellung die gewünschte Anzahl der Pods umfasst.

```
kubectl get deployment metrics-server -n kube-system
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME             READY   UP-TO-DATE   AVAILABLE   AGE
metrics-server   1/1     1            1           6m
```

3. Überprüfen Sie die Funktionsfähigkeit des Metrik-Servers, indem Sie die Ressourcenauslastung (CPU/Speicher) der Knoten anzeigen.

```
kubectl top nodes
```

4. Sollten Sie die Fehlermeldung `Error from server (Forbidden)` erhalten, ist es erforderlich, Ihre Kubernetes-RBAC-Konfiguration zu aktualisieren. Ihre Kubernetes-RBAC-Identität benötigt ausreichende Berechtigungen, um Cluster-Metriken lesen zu können. Sehen Sie sich die mindestens erforderlichen Kubernetes-API-Berechtigungen zum Lesen von Metriken an. GitHub Erfahren Sie, wie Sie AWS IAM-Identitäten wie Rollen Zugriff auf Kubernetes gewähren. APIs

Bereitstellung von Anwendungen mit Helm in Amazon EKS

Der Helm-Paketmanager für Kubernetes unterstützt Sie bei der Installation und Verwaltung von Anwendungen in Ihrem Kubernetes-Cluster. Weitere Informationen finden Sie in der Helm-Dokumentation. Dieses Thema hilft Ihnen bei der Installation und Ausführung der Helm-Binärdateien, sodass Sie Charts mit der Helm-CLI auf Ihrem lokalen System installieren und verwalten können.

> **Important**
>
> Bevor Sie Helm-Charts in Ihrem Amazon-EKS-Cluster installieren können, müssen Sie `kubectl` so konfigurieren, dass es für Amazon EKS funktioniert. Wenn Sie dies noch nicht getan haben, lesen Sie the section called “Zugriff auf den Cluster mit kubectl”, bevor Sie fortfahren. Wenn der folgende Befehl für Ihren Cluster erfolgreich ist, ist die Konfiguration korrekt.
>
> ```
> kubectl get svc
> ```

1. Führen Sie den entsprechenden Befehl für das Client-Betriebssystem aus.

- Wenn Sie macOS mit Homebrew verwenden, installieren Sie die Binärdateien mit dem folgenden Befehl.

```
brew install helm
```

- Weitere Installationsoptionen finden Sie unter Installieren von Helm in den Helm-Dokumenten.

> **Note**
>
> Wenn Sie eine Nachricht bekommen, dass zunächst `openssl` installiert werden muss, können Sie es mit dem folgenden Befehl installieren.

```
sudo yum install openssl
```

1. Um die neue Binärdatei in Ihren `PATH` aufzunehmen, schließen Sie Ihr aktuelles Terminalfenster und öffnen Sie ein neues.
2. Sehen Sie die Version von Helm, die Sie installiert haben.

```
helm version --template='{{ .Version }}{{ "\n" }}'
```

   Eine Beispielausgabe sieht wie folgt aus.

```
v3.17.2
```

3. Stellen Sie sicher, dass die installierte Version mit Ihrer Cluster-Version kompatibel ist Weitere Informationen finden Sie unter Unterstützte Versionsabweichung. Wenn Sie beispielsweise mit `3.17.x` arbeiten, sollte die unterstützte Kubernetes-Version nicht außerhalb des Bereichs von `1.29.x` ~ `1.32.x` liegen.
4. An dieser Stelle können Sie beliebige Helm-Befehle ausführen (z. B. `helm install` *`chart-name`*), um Helm-Charts in Ihrem Cluster zu installieren, zu ändern, zu löschen oder abzufragen. Wenn Sie Helm zum ersten Mal verwenden und kein bestimmtes Diagramm installieren möchten, können Sie Folgendes tun:
   - Experimentieren, indem Sie ein Beispiel-Chart installieren. Ein Beispiel-Chart installieren in der Helm- Schnellstartanleitung lesen.

- Erstellen Sie ein Beispieldiagramm und übertragen Sie es an Amazon ECR. Weitere Informationen finden Sie unter Übertragen eines Helm-Diagramms im Amazon-Elastic-Container-Registry-Benutzerhandbuch.
- Installieren Sie ein Amazon-EKS-Diagramm aus dem eks-charts-GitHub-Repository oder von ArtifactHub.

Organisation von Amazon-EKS-Ressourcen mit Tags

Sie können Tags verwenden, die Sie bei der Verwaltung Ihrer Amazon-EKS-Ressourcen unterstützen. Dieses Thema bietet einen Überblick über die Tags -Funktion und zeigt, wie Sie Tags erstellen können.

Themen

- Grundlagen zu Tags (Markierungen)
- Markieren Ihrer -Ressourcen
- Tag-Einschränkungen
- Markieren von Ressourcen für die Fakturierung
- Arbeiten mit Tags über die Konsole
- Arbeiten mit Tags mittels CLI, API oder eksctl

> **Note**
>
> Tags sind eine Art von Metadaten, die von Kubernetes-Labels und Annotationen getrennt sind. Weitere Informationen zu diesen anderen Metadatentypen finden Sie in folgenden Abschnitten der Kubernetes-Dokumentation:
>
> - Labels und Selektoren
> - Anmerkungen

Grundlagen zu Tags (Markierungen)

Ein Tag ist eine Markierung, die Sie einer AWS-Ressource zuordnen. Jedes Tag (Markierung) besteht aus einem Schlüssel und einem optionalen Wert.

Sie können mit Tags auch Ihre AWS-Ressourcen kategorisieren. Sie können Ressourcen beispielsweise nach Zweck, Inhaber oder Umgebung kategorisieren. Wenn Sie viele Ressourcen desselben Typs haben, können Sie die Tags, die Sie einer bestimmten Ressource zugewiesen haben, verwenden, um diese Ressource schnell zu identifizieren. Sie können beispielsweise eine Reihe von Tags für Ihre Amazon-EKS-Cluster definieren, um Ihnen dabei zu helfen, den Eigentümer und die Stack-Ebene jedes einzelnen Clusters nachzuverfolgen. Sie sollten für jeden Ressourcentyp einen konsistenten Satz von Tag-Schlüsseln entwickeln. Anschließend können Sie die Ressourcen basierend auf den hinzugefügten Tags suchen und filtern.

Nachdem Sie ein Tag hinzugefügt haben, können Sie jederzeit Tag-Schlüssel und -Werte bearbeiten oder Tags aus einer Ressource entfernen. Wenn Sie eine Ressource löschen, werden alle Tags (Markierungen) der Ressource ebenfalls gelöscht.

Tags haben keine semantische Bedeutung für Amazon EKS und werden ausschließlich als Zeichenfolgen interpretiert. Sie können den Wert eines Tags auf eine leere Zeichenfolge setzen. Sie können jedoch den Wert eines Tags nicht auf null setzen. Wenn Sie ein Tag (Markierung) mit demselben Schlüssel wie ein vorhandener Tag (Markierung) für die Ressource hinzufügen, wird der alte Wert mit dem neuen überschrieben.

Wenn Sie AWS Identity and Access Management (IAM) verwenden, können Sie steuern, welche Benutzer in Ihrem AWS-Konto Tags verwalten dürfen.

Markieren Ihrer -Ressourcen

Die folgenden Amazon-EKS-Ressourcen unterstützen Tags:

- Cluster
- Verwaltete Knotengruppen
- Fargate-Profile

So wenden Sie Tags auf Ressourcen an:

- Wenn Sie die Amazon-EKS-Konsole verwenden, können Sie Tags jederzeit auf neue oder vorhandene Ressourcen anwenden. Dies können Sie über dasTagsauf der entsprechenden Ressourcenseite. Weitere Informationen finden Sie unter the section called “Arbeiten mit Tags über die Konsole”.
- Wenn Sie `eksctl` verwenden, können Sie Tags auf Ressourcen anwenden, wenn sie mit der `--tags`-Option erstellt wurden.

- Wenn Sie die AWS-CLI, die Amazon-EKS-API oder ein AWS-SDK verwenden, können Sie Tags mithilfe des `tags`-Parameters auf der entsprechenden API-Aktion auf neue Ressourcen anwenden. Sie können auch über die `TagResource`-API Tags auf Ressourcen anwenden. Weitere Informationen finden Sie unter TagResource.

Wenn Sie einige Aktionen zur Erstellung von Ressourcen verwenden, können Sie gleichzeitig mit der Erstellung auch Tags für die Ressource angeben. Wenn Tags nicht angewendet werden können, während die Ressource erstellt wird, kann die Ressource nicht erstellt werden. Dieser Mechanismus stellt sicher, dass Ressourcen, die Sie mit Tags versehen möchten, entweder mit den von Ihnen angegebenen Tags oder gar nicht erstellt werden. Wenn Sie Ressourcen bei ihrer Erstellung mit Tags versehen, müssen Sie nach der Erstellung der Ressource keine benutzerdefinierten Tagging-Skripte ausführen.

Tags werden nicht auf andere Ressourcen bertragen, die der von Ihnen erstellten Ressource zugeordnet sind. Fargate-Profil-Tags werden beispielsweise nicht zu weiteren dem Fargate-Profil zugeordneten Ressourcen propagiert, z. B. zu den Pods, die mit ihm geplant sind.

Tag-Einschränkungen

Für Tags gelten die folgenden Einschränkungen:

- Maximal 50 Tags können einer Ressource zugeordnet werden.
- Tag-Schlüssel können nicht für eine Ressource wiederholt werden. Jeder Tag-Schlüssel muss eindeutig sein, und jeder Schlüssel darf nur einen Wert besitzen.
- Schlüssel können aus bis zu 128 Zeichen im UTF-8-Format bestehen.
- Werte können aus bis zu 256 Zeichen im UTF-8-Format bestehen.
- Wenn mehrere AWS-Services und Ressourcen Ihr Tagging-Schema verwenden, beschränken Sie die Arten von Zeichen, die Sie verwenden. Einige Services haben möglicherweise Einschränkungen für zulässige Zeichen. Allgemein erlaubte Zeichen sind: Buchstaben, Zahlen, Leerzeichen und die folgenden Sonderzeichen: `+ - = . _ : / @`.
- Bei Tag-Schlüsseln und -Werten muss die Groß- und Kleinschreibung beachtet werden.
- Verwenden Sie weder `aws:`, `AWS:` noch oder andere Schreibweisen hiervon mit Kombinationen aus Groß- und Kleinbuchstaben als Präfix für Schlüssel oder Werte. Diese sind nur für die AWS-Verwendung reserviert. Sie können keine Tag-Schlüssel oder Werte mit diesem Präfix bearbeiten oder löschen. Tags mit diesem Präfix werden nicht zum Limit für Tags pro Ressource gezählt.

Markieren von Ressourcen für die Fakturierung

Wenn Sie Tags auf Amazon-EKS-Cluster anwenden, können Sie sie für die Kostenzuweisung in Ihren Kosten- und Nutzungsberichten verwenden. Die Messdaten in Ihren Kosten- und Nutzungsberichten zeigen die Nutzung für alle Ihre Amazon-EKS-Cluster an. Weitere Informationen finden Sie im AWS-Kosten- und Nutzungsbericht im Benutzerhandbuch zur AWS-Fakturierung.

Mit dem AWS generierten Kostenzuordnungs-Tag, insbesondere `aws:eks:cluster-name`, können Sie die Amazon-EC2-Instance-Kosten nach einzelnen Amazon-EKS-Clustern im Cost Explorer (Kosten-Explorer) aufschlüsseln. Dieses Tag erfasst jedoch nicht die Kosten der Steuerebene. Das Tag wird automatisch zu Amazon-EC2-Instances hinzugefügt, die Teil eines Amazon-EKS-Clusters sind. Dieses Verhalten tritt unabhängig davon auf, ob die Instances mit von Amazon EKS verwalteten Knotengruppen, Karpenter oder direkt mit Amazon EC2 bereitgestellt werden. Dieses spezielle Tag wird nicht auf das Limit von 50 Tags angerechnet. Um das Tag zu verwenden, muss der Kontoinhaber es in der AWS-Fakturierungskonsole oder mithilfe der API aktivieren. Wenn ein Besitzer des AWS-Organizations-Verwaltungskontos das Tag aktiviert, wird es für alle Mitgliedskonten der Organisation ebenfalls aktiviert.

Sie können Ihre Rechnungsinformationen auch auf der Grundlage von Ressourcen organisieren, die dieselben Tag-Schlüsselwerte haben. Beispielsweise können Sie mehrere Ressourcen mit einem bestimmten Anwendungsnamen taggen und dann Ihre Fakturierungsinformationen organisieren. Auf diese Weise können Sie die Gesamtkosten dieser Anwendung über mehrere Services hinweg sehen. Weitere Informationen zum Einrichten eines Kostenzuordnungsberichts mit Markierungen finden Sie unter Monatlicher Kostenzuordnungsbericht im Benutzerhandbuch zur AWS-Fakturierung.

> **Note**
>
> Wenn Sie die Berichterstellung gerade erst aktiviert haben, werden die Daten für den aktuellen Monat nach 24 Stunden bereitgestellt.

Cost Explorer ist ein Reporting-Tool, das als Teil des kostenlosen Kontingents von AWS verfügbar ist. Sie können Cost Explorer verwenden, um Diagramme Ihrer Amazon-EKS-Ressourcen der letzten 13 Monate anzuzeigen. Sie können auch prognostizieren, wie viel Sie wahrscheinlich für die nächsten drei Monate ausgeben werden. Sie können Muster sehen, wie viel Sie für AWS-Ressourcen im Zeitablauf ausgeben. Sie können es zum Beispiel verwenden, um Bereiche zu identifizieren, die eine genauere Untersuchung erfordern, sowie um Trends auszumachen, die Ihnen helfen, Ihre Kosten

zu verstehen. Sie können auch Zeitbereiche für die Daten angeben und die Daten nach Tagen oder Monate anzeigen lassen.

Arbeiten mit Tags über die Konsole

Mithilfe der Amazon-EKS-Konsole können Sie die Tags im Zusammenhang mit neuen oder vorhandenen Clustern und verwalteten Knotengruppen verwalten.

Wenn Sie eine ressourcenspezifische Seite in der Amazon-EKS-Konsole auswählen, wird auf der Seite eine Liste der Ressourcen angezeigt. Wenn Sie beispielsweise Clusters (Cluster) im Navigationsbereich auswählen, zeigt die Konsole eine Liste der Amazon EKS-Cluster an. Wenn Sie eine Ressource aus einer dieser Listen auswählen (z. B. einen bestimmten Cluster) und die Ressource Tags unterstützt, können Sie deren Tags auf der Registerkarte Tags anzeigen und verwalten.

Sie können auch den Tag-Editor in der AWS-Managementkonsole verwenden, der Ihnen eine einheitliche Möglichkeit gibt, Ihre Tags zu verwalten. Weitere Informationen zum Markieren Ihrer AWS-Ressourcen mit Tag-Editor finden Sie im AWS-Tag-Editor-Benutzerhandbuch.

Hinzufügen von Tags zu einer Ressource bei der Erstellung

Sie können Tags zu Amazon EKS-Clustern und verwalteten Knotengruppen hinzufügen, wenn Sie sie erstellen. Weitere Informationen finden Sie unter the section called “Erstellen eines Clusters”.

Hinzufügen und Löschen von Tags für eine Ressource

Sie können die Tags, die mit Ihren Clustern verbunden sind, direkt auf der Seite der Ressource hinzufügen oder löschen.

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie in der Navigationsleiste die gewünschte AWS-Region aus.
3. Wählen Sie im linken Navigationsbereich die Option Cluster aus.
4. Wählen Sie einen bestimmten Cluster aus.
5. Wählen Sie die Registerkarte Tags und dann Manage tags (Tags verwalten).
6. Fügen Sie auf der Seite Manage tags (Tags verwalten) Ihre Tags bei Bedarf hinzu oder löschen Sie sie.
   - Um einen Tag hinzuzufügen, wählen Sie Add tag (Tag hinzufügen). Geben Sie dann den Schlüssel und den Wert für jedes Tag an.

- Um ein Tag zu entfernen, wählen Sie Remove tag (Tag entfernen) aus.

7. Wiederholen Sie diesen Vorgang für jedes Tag, das Sie hinzufügen oder löschen möchten.
8. Wählen Sie zum Abschluss Update (Aktualisieren) aus.

Arbeiten mit Tags mittels CLI, API oder `eksctl`

Verwenden Sie die folgenden AWS-CLI-Befehle oder Amazon-EKS-API-Operationen, um die Tags für Ihre Ressourcen hinzuzufügen, zu aktualisieren, aufzulisten und zu löschen. Sie können `eksctl` nur dann zum Hinzufügen von Tags verwenden, wenn Sie gleichzeitig neue Ressourcen mit einem Befehl erstellen.

Aufgabe	AWS-CLI	AWS-Tools für Windows PowerShell	API-Aktion
Fügen Sie einen oder mehrere Tags hinzu oder überschreiben Sie sie.	tag-resource	Add-EKSResourceTag	TagResource
Löschen Sie ein oder mehrere Tags.	untag-resource	Remove-EKSResourceTag	UntagResource

Die folgenden Beispiele zeigen, wie man Tags an Ressourcen mithilfe der AWS-CLI hinzufügt oder entfernt.

Beispiel 1: Markieren eines bestehenden Clusters

Der folgende Befehl markiert einen vorhandenen Cluster.

```
aws eks tag-resource --resource-arn resource_ARN --tags team=devs
```

Beispiel 2: Entfernen von Tags von einem bestehenden Cluster

Der folgende Befehl löscht ein Tag von einem bestehenden Cluster.

```
aws eks untag-resource --resource-arn resource_ARN --tag-keys tag_key
```

Beispiel 3: Tags für eine Ressource auflisten

Der folgende Befehl listet die Tags auf, die einer vorhandenen Ressource zugeordnet sind.

```
aws eks list-tags-for-resource --resource-arn resource_ARN
```

Wenn Sie einige Aktionen zur Erstellung von Ressourcen verwenden, können Sie gleichzeitig mit der Erstellung der Ressource auch Tags angeben. Die folgenden Aktionen unterstützen das Taggen beim Erstellen einer Ressource.

Aufgabe	AWS-CLI	AWS-Tools für Windows PowerShell	API-Aktion	eksctl
Erstellen eines Clusters	create-cluster	New-EKSCluster	CreateCluster	`create cluster`
Erstellen einer verwalteten Knotengruppe	create-nodegroup	New-EKSNodegroup	CreateNodegroup	`create nodegroup`
Erstellen eines Fargate-Profils	create-fargate-profile	New-EKSFargateProfile	CreateFargateProfile.html	`create fargatepr ofile`

- Wenn Sie beim Erstellen einer verwalteten Knotengruppe auch die Amazon-EC2-Instances mit Tags versehen möchten, erstellen Sie die verwaltete Knotengruppe mithilfe einer Startvorlage. Weitere Informationen finden Sie unter the section called “ EC2 Amazon-Instances taggen”. Wenn Ihre Instances bereits vorhanden sind, können Sie die Instanzen manuell mit Tags versehen. Weitere Informationen finden Sie unter Tagging Ihrer Ressourcen im Amazon-EC2-Benutzerhandbuch.

Service Quotas für Amazon EKS und Fargate anzeigen und verwalten

Amazon EKS hat Service Quotas integriert, einen AWS Service, mit dem Sie Ihre Kontingente von einem zentralen Ort aus einsehen und verwalten können. Weitere Informationen zu Service Quotas finden Sie unter Was sind Service Quotas im Benutzerhandbuch für Service Quotas. Mit der Integration von Service Quotas können Sie mithilfe der und AWS CLI schnell den Wert Ihrer Amazon EKS- und AWS Fargate-Servicekontingenten AWS-Managementkonsole ermitteln.

EKS-Servicekontingente finden Sie in der AWS-Managementkonsole

1. Öffnen Sie die Service Quotas-Konsole.
2. Wählen Sie im linken Navigationsbereich AWS Dienste aus.
3. Suchen Sie in der AWS Serviceliste nach Amazon Elastic Kubernetes Service (Amazon EKS) oder AWS Fargate und wählen Sie diese aus.

   In der Liste der Service-Kontingente finden Sie den Namen des Service-Kontingents, den angewendeten Wert (falls verfügbar), das AWS Standardkontingent und ob der Kontingentwert anpassbar ist.
4. Wählen Sie den Kontingentnamen, um zusätzliche Informationen zu einem Service Quota anzuzeigen, z. B. seine Beschreibung.
5. (Optional) Um eine Kontingenterhöhung zu beantragen, wählen Sie das Kontingent, das Sie erhöhen möchten, und dann Request quota increase (Kontingenterhöhung beantragen) aus, geben Sie die erforderlichen Informationen ein, und wählen Sie dann Request (Beantragen) aus.

Weitere Informationen zum Umgang mit Servicekontingenten mithilfe von finden Sie im Service Quotas User Guide. AWS-Managementkonsole Informationen zum Beantragen einer Kontingenterhöhung finden Sie unter Beantragen einer Kontingenterhöhung im Service-Quotas-Benutzerhandbuch.

EKS-Dienstkontingente mit der AWS CLI anzeigen

Führen Sie den folgenden Befehl aus, um Ihre Amazon-EKS-Kontingente anzuzeigen.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
```

```
    --service-code eks \
    --output table
```

Führen Sie den folgenden Befehl aus, um Ihre Fargate-Kontingente anzuzeigen.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code fargate \
    --output table
```

> **Note**
>
> Das zurückgegebene Kontingent ist die Anzahl der Amazon ECS-Aufgaben oder Amazon EKS-Pods, die in diesem Konto in der aktuellen AWS Region gleichzeitig auf Fargate ausgeführt werden können.

Weitere Informationen zum Arbeiten mit Servicekontingenten mithilfe der AWS CLI finden Sie unter service-quota in der AWS CLI-Befehlsreferenz. Informationen zum Anfordern einer Erhöhung des Kontingents finden Sie unter dem request-service-quota-increaseBefehl in der AWS CLI-Befehlsreferenz.

Amazon-EKS-Service-Quotas

AWS empfiehlt die Verwendung von AWS-Managementkonsole , um Ihre aktuellen Kontingente einzusehen. Weitere Informationen finden Sie unter the section called “EKS-Servicekontingente finden Sie in der AWS-Managementkonsole”.

Die standardmäßigen EKS-Servicekontingente finden Sie unter Amazon Elastic Kubernetes Service Service-Endpunkte und Kontingente in der AWS Allgemeinen Referenz.

Diese Servicekontingente werden unter Amazon Elastic Kubernetes Service (Amazon EKS) in der Service-Quotas-Konsole aufgeführt. Informationen zum Anfordern einer Kontingenterhöhung für Werte, die als anpassbar angezeigt werden, finden Sie unter Beantragen einer Kontingenterhöhung im Benutzerhandbuch von Service Quotas.

> **Note**
>
> Anpassungen der folgenden Komponenten werden in Service Quotas nicht unterstützt: * Pod Identity-Zuordnungen pro Cluster. Grenzwerte finden Sie unter. the section called “Pod Identity” * CIDRs für Remote Node Networks oder Remote Pod Networks für Hybridknoten. Informationen zu den Beschränkungen finden Sie unter the section called “Hybridknoten”.

AWS Fargate-Servicekontingente

Der AWS Fargate-Dienst in der Service Quotas-Konsole listet mehrere Service Quotas auf. Sie können Alarme konfigurieren, mit denen Sie benachrichtigt werden, wenn sich Ihre Nutzung einem Servicekontingent nähert. Weitere Informationen finden Sie unter the section called “Einen CloudWatch Alarm zur Überwachung der Fargate-Ressourcennutzungsmetriken erstellen”.

Neue AWS Konten haben möglicherweise niedrigere anfängliche Kontingente, die sich im Laufe der Zeit erhöhen können. Fargate überwacht ständig die Kontonutzung in jeder AWS Region und erhöht dann automatisch die Kontingente je nach Nutzung. Außerdem können Sie eine Kontingenterhöhung für Werte anfordern, die als anpassbar angezeigt werden. Weitere Informationen finden Sie unter Beantragen einer Kontingenterhöhung im Service-Quotas-Benutzerhandbuch.

AWS empfiehlt, sie zu verwenden, um Ihre aktuellen AWS-Managementkonsole Kontingente einzusehen. Weitere Informationen finden Sie unter the section called “EKS-Servicekontingente finden Sie in der AWS-Managementkonsole”.

Die standardmäßigen AWS Fargate on EKS-Dienstkontingente finden Sie unter Fargate-Dienstkontingente in der AWS Allgemeinen Referenz.

> **Note**
>
> Fargate erzwingt zusätzlich Ratenkontingente für Amazon-ECS-Aufgaben und den Start von Amazon-EKS-Pods. Weitere Informationen finden Sie unter Drosselungskontingente für AWS Fargate im Handbuch zu Amazon ECS.

Sicherheit in Amazon EKs

Cloud-Sicherheit AWS hat höchste Priorität. Als AWS Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame Verantwortung von Ihnen AWS und Ihnen. Das Modell der geteilten Verantwortung beschreibt dies als Sicherheit der Cloud und Sicherheit in der Cloud:

- Sicherheit der Cloud — AWS ist verantwortlich für den Schutz der Infrastruktur, die AWS Dienste in der AWS Cloud ausführt. AWS Ist für Amazon EKS für die Kubernetes-Steuerebene verantwortlich, die die Knoten und `etcd` die Datenbank der Kontrollebene umfasst. Auditoren von Drittanbietern testen und überprüfen die Effektivität unserer Sicherheitsmaßnahmen im Rahmen der AWS - Compliance-Programme regelmäßig. Informationen zu den Compliance-Programmen, die für Amazon EKS gelten, finden Sie unter AWS -Services im Geltungsbereich nach Compliance-Programm.
- Sicherheit in der Cloud – Ihre Verantwortlichkeit umfasst die folgenden Bereiche.
  - Die Sicherheitskonfiguration der Datenebene, einschließlich der Konfiguration der Sicherheitsgruppen, die den Durchgang des Datenverkehrs aus der Amazon-EKS-Steuerebene in die VPC des Kunden zulassen.
  - Die Konfiguration der Knoten und der Container selbst.
  - Das Betriebssystem (einschließlich Aktualisierungen und Sicherheits-Patches) des Knotens
  - Andere zugehörige Anwendungssoftware:
    - Einrichten und Verwalten von Kontrollelementen für das Netzwerk (wie beispielsweise Firewall-Regeln)
    - Verwalten des Identity and Access Management auf Plattformebene, entweder mit oder zusätzlich zu IAM
  - Die Vertraulichkeit Ihrer Daten, die Anforderungen Ihres Unternehmens und geltende Gesetze und Vorschriften

Amazon EKS ist durch mehrere Compliance-Programme für regulierte und vertrauliche Anwendungen zertifiziert. Amazon EKS ist konform mit SOC, PCI, ISO, FedRAMP-Moderate, IRAP, C5, K-ISMS, ENS High, OSPAR, HITRUST CSF und ist ein HIPAA-fähiger Service. Weitere Informationen finden Sie unter Zugriff verwalten.

Diese Dokumentation zeigt Ihnen, wie Sie das Modell der übergreifenden Verantwortlichkeit bei der Verwendung von Amazon EKS einsetzen können. Die folgenden Themen zeigen Ihnen, wie Sie Amazon EKS konfigurieren, um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Sie erfahren auch, wie Sie andere AWS Services nutzen können, mit denen Sie Ihre Amazon EKS-Ressourcen überwachen und sichern können.

> **Note**
>
> Linux-Container bestehen aus Kontrollgruppen (Cgroups) und Namespaces, die dazu beitragen, den Zugriff eines Containers einzuschränken, aber alle Container verwenden denselben Linux-Kernel wie die Host-Amazon-Instance. EC2 Es wird dringend davon abgeraten, einen Container als Root-Benutzer (UID 0) auszuführen oder einem Container Zugriff auf Hostressourcen oder Namespaces wie das Hostnetzwerk oder den Host-PID-Namespace zu gewähren, da dies die Effektivität der von Containern bereitgestellten Isolation verringert.

Themen

- Amazon-EKS-Cluster mit bewährten Methoden sichern
- Analyse von Schwachstellen in Amazon EKS
- Compliance-Validierung für Amazon-EKS-Cluster
- Sicherheitsüberlegungen für Amazon Elastic Kubernetes Service
- Sicherheitsüberlegungen für Kubernetes
- Sicherheitsüberlegungen für Amazon EKS Auto Mode
- Sicherheitsüberlegungen für EKS-Funktionen
- Identitäts- und Zugriffsverwaltung für Amazon EKS

Amazon-EKS-Cluster mit bewährten Methoden sichern

Die bewährten Sicherheitsmethoden für Amazon EKS finden unter Bewährte Methoden für die Sicherheit im Leitfaden zu bewährten Methoden für Amazon EKS.

Analyse von Schwachstellen in Amazon EKS

Sicherheit ist ein wichtiger Aspekt bei der Konfiguration und Wartung von Kubernetes-Clustern und -Anwendungen. Nachfolgend finden Sie Ressourcen zur Analyse der Sicherheitskonfiguration Ihrer EKS-Cluster, Ressourcen zur Überprüfung auf Schwachstellen sowie Integrationen mit AWS-Services, die diese Analyse für Sie durchführen.

Der Center for Internet Security (CIS)-Benchmark für Amazon EKS

Der Kubernetes Benchmark des Center for Internet Security (CIS) bietet Anleitungen für Amazon-EKS-Sicherheitskonfigurationen. Die Benchmark:

- Gilt für Amazon EC2-Knoten (sowohl verwaltete als auch selbstverwaltete), bei denen Sie für die Sicherheitskonfigurationen von Kubernetes-Komponenten verantwortlich sind.
- Bietet eine von der Community genehmigte Standardmethode, um sicherzustellen, dass Sie Ihren Kubernetes-Cluster und Ihre Knoten sicher konfiguriert haben, wenn Sie Amazon EKS verwenden.
- Besteht aus vier Abschnitten; Protokollierungskonfiguration der Steuerebene, Knotensicherheitskonfigurationen, Richtlinien und verwaltete Services.
- Unterstützt alle Kubernetes-Versionen, die derzeit in Amazon EKS verfügbar sind und können mitkube-bench, ein Standard-Open-Source-Tool zur Überprüfung der Konfiguration mit dem CIS-Benchmark auf Kubernetes-Clustern.

Weitere Informationen hierzu finden Sie unterEinführung des CIS Amazon EKS Benchmarkaus.

Eine automatisierte `aws-sample`-Pipeline zur Aktualisierung Ihrer Knotengruppe mit einer CIS-geprüften AMI finden Sie unter EKS-optimierte AMI-Hardening-Pipeline.

Amazon-EKS-Plattformversionen

Amazon-EKS-Plattformversionen definieren die Funktionalitäten der Cluster-Steuerebene (z. B. welche Kubernetes-API-Server-Flags aktiviert sind) sowie die aktuelle Kubernetes-Patch-Version. Neue Cluster werden mit der neuesten Plattformversion bereitgestellt. Weitere Details finden Sie unter EKS-Plattformversionen.

Sie können einen Amazon EKS-Cluster auf neuere Kubernetes-Versionen aktualisieren. Wenn neue Kubernetes-Versionen in Amazon EKS verfügbar werden, empfehlen wir Ihnen, Ihre Cluster proaktiv auf die neueste verfügbare Version zu aktualisieren. Weitere Informationen zu den Kubernetes-Versionen in EKS finden Sie unter Von Amazon EKS unterstützte Versionen.

Liste der Betriebssystem-Schwachstellen

AL2023-Schwachstellenliste

Sie können Sicherheits- oder Datenschutzereignisse für Amazon Linux 2023 im Amazon Linux-Sicherheitszentrum verfolgen oder den zugehörigen RSS-Feed abonnieren. Sicherheits- und Datenschutzereignisse enthalten eine Übersicht über das Problem sowie Pakete und Anweisungen zum Aktualisieren Ihrer Instances, um das Problem zu beheben.

Schwachstellenliste für Amazon Linux 2

Sie können Sicherheits- oder Datenschutzereignisse für Amazon Linux 2 im Amazon Linux-Sicherheitszentrum verfolgen oder den zugehörigen RSS-Feed abonnieren. Sicherheits- und Datenschutzereignisse enthalten eine Übersicht über das Problem sowie Pakete und Anweisungen zum Aktualisieren Ihrer Instances, um das Problem zu beheben.

Knoten-Erkennung mit Amazon Inspector

Mit Amazon Inspector können Sie eine Prüfung auf nicht gewünschte Netzwerkzugänglichkeit Ihrer Arbeitsknoten und auf Schwachstellen auf diesen Amazon-EC2-Instances vornehmen.

Cluster- und Knoten-Erkennung mit Amazon GuardDuty

Der Amazon GuardDuty-Service zur Erkennung von Bedrohungen unterstützt Sie beim Schutz Ihrer Konten, Container, Workloads und Daten in Ihrer AWS-Umgebung. Neben anderen Features bietet GuardDuty die folgenden zwei Features, die potenzielle Bedrohungen für Ihre EKS-Cluster erkennen: EKS-Schutz und Laufzeit-Überwachung.

Weitere Informationen finden Sie unter the section called “Amazon GuardDuty”.

Compliance-Validierung für Amazon-EKS-Cluster

Informationen dazu, ob ein AWS-Service im Rahmen bestimmter Compliance-Programme angeboten wird, finden Sie unter AWS-Services im Rahmen von Compliance-Programmen. Wählen Sie dazu das gewünschte Compliance-Programm aus. Allgemeine Informationen finden Sie unter AWS-Compliance-Programme.

Die Auditberichte von Drittanbietern lassen sich mit AWS Artifact herunterladen. Weitere Informationen finden Sie unter Herunterladen von Berichten in AWS-Artifact.

Ihre Verantwortung in Bezug auf die Compliance bei der Verwendung von AWS-Services ergibt sich aus der Vertraulichkeit der Daten, den Compliance-Zielen des Unternehmens sowie den einschlägigen Gesetzen und Vorschriften. AWS stellt die folgenden Ressourcen zur Sicherstellung der Compliance bereit:

- Compliance und Governance im Bereich Sicherheit – In diesen Anleitungen für die Lösungsimplementierung werden Überlegungen zur Architektur behandelt. Außerdem werden Schritte für die Bereitstellung von Sicherheits- und Compliance-Features beschrieben.
- Referenz für berechtigte HIPAA-Services – Listet berechtigte HIPAA-Services auf. Nicht alle AWS-Services sind HIPAA-konform.
- AWS-Compliance-Ressourcen – Diese Arbeitsbücher und Leitfäden könnten für Ihre Branche und Ihren Standort interessant sein.
- AWS-Compliance-Leitfäden für Kunden – Verstehen Sie das Modell der geteilten Verantwortung aus dem Blickwinkel der Einhaltung von Vorschriften. In den Leitfäden werden die bewährten Methoden zum Schutz von AWS-Services zusammengefasst und die Leitlinien den Sicherheitskontrollen in verschiedenen Frameworks (einschließlich des National Institute of Standards and Technology (NIST), des Payment Card Industry Security Standards Council (PCI) und der International Organization for Standardization (ISO)) zugeordnet.
- Auswertung von Ressourcen mit Regeln im AWS-Config-Entwicklerhandbuch – Der AWS-Config-Service bewertet, wie gut Ihre Ressourcenkonfigurationen mit internen Praktiken, Branchenrichtlinien und Vorschriften übereinstimmen.
- AWS Security Hub – Dieser AWS-Service bietet einen umfassenden Überblick über Ihren Sicherheitsstatus innerhalb von AWS. Security Hub verwendet Sicherheitskontrollen, um Ihre AWS-Ressourcen zu bewerten und Ihre Einhaltung von Sicherheitsstandards und bewährten Methoden zu überprüfen. Die Liste der unterstützten Services und Kontrollen finden Sie in der Security-Hub-Steuerelementreferenz.
- Amazon GuardDuty – Dieser AWS-Service erkennt potenzielle Bedrohungen für Ihre AWS-Konten, Workloads, Container und Daten, indem er Ihre Umgebung auf verdächtige und böswillige Aktivitäten überwacht. GuardDuty kann Ihnen helfen, verschiedenen Compliance-Anforderungen wie PCI DSS nachzukommen, indem es die Anforderungen zur Erkennung von Eindringlingen erfüllt, die in bestimmten Compliance-Frameworks vorgeschrieben sind.
- AWS Audit Manager – Dieser AWS-Service hilft Ihnen, Ihre AWS-Nutzung kontinuierlich zu überprüfen, um zu vereinfachen, wie Sie mit Risiken umgehen und Compliance sowie Branchenstandards einhalten.

Sicherheitsüberlegungen für Amazon Elastic Kubernetes Service

Nachfolgend finden Sie Überlegungen zur Sicherheit der Cloud, da diese Auswirkungen auf Amazon EKS haben.

Themen

- Infrastruktursicherheit in Amazon EKS
- Ausfallsicherheit in Amazon-EKS-Clustern verstehen
- Serviceübergreifende Vermeidung verwirrter Stellvertreter in Amazon EKS

Infrastruktursicherheit in Amazon EKS

Als verwalteter Service ist Amazon Elastic Kubernetes Service durch die globale Netzwerksicherheit von AWS geschützt. Informationen zu AWS-Sicherheitsservices und wie AWS die Infrastruktur schützt, finden Sie unter AWS-Cloud-Sicherheit. Informationen zum Entwerfen Ihrer AWS-Umgebung anhand der bewährten Methoden für die Infrastruktursicherheit finden Sie unter Infrastrukturschutz im Security Pillar AWS Well-Architected Framework.

Sie greifen mithilfe von AWS veröffentlichten API-Aufrufe über das Netzwerk auf Amazon EKS zu. Kunden müssen Folgendes unterstützen:

- Transport Layer Security (TLS). Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Verschlüsselungs-Suiten mit Perfect Forward Secrecy (PFS) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Die meisten modernen Systeme wie Java 7 und höher unterstützen diese Modi.

Außerdem müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der einem IAM-Prinzipal zugeordnet ist. Oder Sie können den AWS-Sicherheitstoken-Service (AWS STS) verwenden, um temporäre Sicherheitsanmeldeinformationen zum Signieren von Anfragen zu generieren.

Wenn Sie einen Amazon-EKS-Cluster erstellen, geben Sie die VPC-Subnetze an, die Ihr Cluster verwenden soll. Amazon EKS erfordert Subnetze in mindestens zwei Availability Zones. Wir empfehlen eine VPC mit öffentlichen und privaten Subnetzen, damit Kubernetes öffentliche Load Balancer in den öffentlichen Subnetzen erstellen kann, die den Datenverkehr auf Pods ausgleichen, die auf Knoten in privaten Subnetzen ausgeführt werden.

Weitere Informationen zu Überlegungen bezüglich der VPC finden Sie unter the section called “VPC- und Subnetz-Anforderungen”.

Wenn Sie Ihre VPC und Knotengruppen mit den AWS-CloudFormation-Vorlagen erstellen, die in der schrittweisen Anleitung Erste Schritte mit Amazon EKS bereitgestellt werden, werden die Steuerebene und die Sicherheitsgruppen des Knotens mit unseren empfohlenen Einstellungen konfiguriert.

Weitere Informationen zu Überlegungen bezüglich Sicherheitsgruppen finden Sie unter the section called “Anforderungen an Sicherheitsgruppen”.

Wenn Sie einen neuen Cluster erstellen, erstellt Amazon EKS einen Endpunkt für den verwalteten Kubernetes-API-Server, über den Sie mit Ihrem Cluster kommunizieren (mit Kubernetes-Verwaltungswerkzeugen wie `kubectl`). Standardmäßig ist dieser API-Server-Endpunkt öffentlich im Internet zugänglich und der Zugriff auf den API-Server wird durch eine Kombination aus AWS Identity and Access Management (IAM) und nativer Kubernetes Role Based Access Control (RBAC) gesichert.

Sie können den privaten Zugriff auf den Kubernetes-API-Server aktivieren, sodass die gesamte Kommunikation zwischen Ihren Knoten und dem API-Server innerhalb Ihrer VPC bleibt. Sie können die IP-Adressen einschränken, die über das Internet auf Ihren API-Server zugreifen können, oder den Internetzugriff auf den API-Server vollständig deaktivieren.

Weitere Informationen zum Ändern des Zugriffs auf Cluster-Endpunke finden Sie unter the section called “Ändern des Cluster-Endpunktzugriffs”.

Sie können Kubernetes-Netzwerkrichtlinien mit dem Amazon VPC CNI oder Tools von Drittanbietern wie Project Calico implementieren. Weitere Informationen zur Verwendung von Amazon VPC CNI für Netzwerkrichtlinien finden Sie unter the section called “Kubernetes-Richtlinien”. Project Calico ist ein Open-Source-Projekt eines Drittanbieters. Weitere Informationen finden Sie in der Dokumentation zu Project Calico.

Zugriff auf Amazon EKS über AWS PrivateLink

Sie können mit AWS PrivateLink eine private Verbindung zwischen Ihrer VPC und Amazon Elastic Kubernetes Service herstellen. Sie können auf Amazon EKS zugreifen, als wäre es in Ihrer VPC, ohne dass Sie ein Internet-Gateway, ein NAT-Gerät, eine VPN-Verbindung oder eine AWS-Direct-Connect-Verbindung verwenden müssen. Instances in Ihrer VPC benötigen keine öffentlichen IP-Adressen, um auf Amazon EKS zuzugreifen.

Sie stellen diese private Verbindung her, indem Sie einen Schnittstellenendpunkt erstellen, der von AWS PrivateLink unterstützt wird. Wir erstellen eine Endpunkt-Netzwerkschnittstelle in jedem Subnetz, das Sie für den Schnittstellen-Endpunkt aktivieren. Hierbei handelt es sich um vom Anforderer verwaltete Netzwerkschnittstellen, die als Eingangspunkt für den Datenverkehr dienen, der für Amazon EKS bestimmt ist.

Weitere Informationen finden Sie unter Zugriff auf AWS-Services über AWS PrivateLink im AWS-PrivateLink-Handbuch.

Bevor Sie beginnen

Stellen Sie zunächst sicher, dass Sie die folgenden Aufgaben durchgeführt haben:

- Zugriff auf einen AWS-Service über einen Schnittstellen-VPC-Endpunkt im AWS-PrivateLink-Handbuch überprüfen

Überlegungen

- Support und Einschränkungen: Amazon-EKS-Schnittstellenendpunkte ermöglichen sicheren Zugriff auf alle Amazon-EKS-API-Aktionen von Ihrem VPC aus, weisen jedoch bestimmte Einschränkungen auf: Sie unterstützen keinen Zugriff auf Kubernetes-APIs, da diese über einen separaten privaten Endpunkt verfügen. Sie können Amazon EKS nicht so konfigurieren, dass nur über den Schnittstellenendpunkt darauf zugegriffen werden kann.
- Preise: Für die Verwendung von Schnittstellenendpunkten für Amazon EKS fallen Standardgebühren für AWS PrivateLink an: Stundengebühren für jeden in jeder Availability Zone bereitgestellten Endpunkt, Datenverarbeitungsgebühren für den Datenverkehr über den Endpunkt. Weitere Informationen finden Sie unter Preise zu AWS PrivateLink.
- Sicherheit und Zugriffskontrolle: Wir empfehlen, die Sicherheit zu erhöhen und den Zugriff mit diesen zusätzlichen Konfigurationen zu steuern: Verwenden Sie VPC-Endpunktrichtlinien, um den Zugriff auf Amazon EKS über den Schnittstellenendpunkt zu steuern, ordnen Sie Sicherheitsgruppen den Endpunkt-Netzwerkschnittstellen zu, um den Datenverkehr zu verwalten, und verwenden Sie VPC-Ablaufprotokolle, um den IP-Datenverkehr zu und von den Schnittstellenendpunkten zu erfassen und zu überwachen, wobei die Protokolle in Amazon CloudWatch oder Amazon S3 veröffentlicht werden können. Weitere Informationen finden Sie unter Zugriff auf VPC-Endpunkte mithilfe von Endpunktrichtlinien steuern und Protokollierung von IP-Datenverkehr mithilfe von VPC-Ablaufprotokollen.
- Konnektivitätsoptionen: Schnittstellenendpunkte bieten flexible Konnektivitätsoptionen über den On-Premises-Zugriff (verbinden Sie Ihr lokales Rechenzentrum über AWS Direct Connect oder

AWS Site-to-Site VPN mit einem VPC über den Schnittstellenendpunkt) oder über die Inter-VPC-Konnektivität (verwenden Sie AWS Transit Gateway oder VPC Peering, um andere VPCs mit dem VPC über den Schnittstellenendpunkt zu verbinden, wobei der Datenverkehr innerhalb des AWS-Netzwerks bleibt).

- Support für IP-Versionen: Endpunkte, die vor August 2024 erstellt wurden, unterstützen nur IPv4 unter Verwendung von eks.region.amazonaws.com. Neue Endpunkte, die nach August 2024 erstellt wurden, unterstützen Dual-Stack-IPv4 und IPv6 (z. B. eks.region.amazonaws.com, eks.region.api.aws).
- Regionale Verfügbarkeit: AWS PrivateLink ist für die EKS-API ist in den Regionen Asien-Pazifik (Malaysia) (ap-southeast-5), Asien-Pazifik (Thailand) (ap-southeast-7), Mexiko (Zentral) (mx-central-1) und Asien-Pazifik (Taipeh) (ap-east-2) nicht verfügbar. AWS PrivateLink-Support für eks-auth (EKS Pod Identity) ist in der Region Asien-Pazifik (Malaysia) (ap-southeast-5) verfügbar.

Erstellen eines Schnittstellen-Endpunkts für Amazon EKS

Sie können einen Schnittstellenendpunkt für Amazon EKS entweder über die Amazon-VPC-Konsole oder die AWS-Befehlszeilenschnittstelle (AWS-CLI) erstellen. Weitere Informationen finden Sie unter Erstellen eines VPC-Endpunkts im AWS-PrivateLink-Leitfaden .

Erstellen Sie einen Schnittstellenendpunkt für Amazon EKS unter Verwendung der folgenden Service-Namen:

EKS-API

- com.amazonaws.region-code.eks
- com.amazonaws.region-code.eks-fips (für FIPS-kompatible Endpunkte)

EKS-Auth-API (EKS Pod Identity)

- com.amazonaws.region-code.eks-auth

Private DNS-Feature für Amazon-EKS-Schnittstellenendpunkte

Das private DNS-Feature, das standardmäßig für Schnittstellenendpunkte von Amazon EKS und anderen AWS-Services aktiviert ist, ermöglicht sichere und private API-Anfragen unter Verwendung der standardmäßigen regionalen DNS-Namen. Dieses Feature stellt sicher, dass API-Aufrufe über

den Schnittstellenendpunkt über das private AWS-Netzwerk geleitet werden, was die Sicherheit und Leistung verbessert.

Das private DNS-Feature wird automatisch aktiviert, wenn Sie einen Schnittstellenendpunkt für Amazon EKS oder andere AWS-Services erstellen. Zur Aktivierung, müssen Sie Ihre VPC korrekt konfigurieren, indem Sie bestimmte Attribute festlegen:

- enableDnsHostnames: Ermöglicht Instances innerhalb der VPC, DNS-Host-Namen zu verwenden.
- enableDnsSupport: Aktiviert die DNS-Auflösung im gesamten VPC.

Eine Schritt-für-Schritt-Anleitung zum Überprüfen oder Ändern dieser Einstellungen finden Sie unter Anzeigen und Aktualisieren der DNS-Attribute für Ihre VPC.

DNS-Namen und IP-Adresstypen

Wenn das private DNS-Feature aktiviert ist, können Sie bestimmte DNS-Namen verwenden, um eine Verbindung zu Amazon EKS herzustellen. Diese Optionen werden im Laufe der Zeit weiterentwickelt:

- eks.region.amazonaws.com: Der herkömmliche DNS-Name, der vor August 2024 nur zu IPv4-Adressen aufgelöst wird. Für vorhandene Endpunkte, die auf Dual-Stack aktualisiert wurden, wird dieser Name sowohl in IPv4- als auch in IPv6-Adressen aufgelöst.
- eks.region.api.aws: Dieser Dual-Stack-DNS-Name ist für neue Endpunkte verfügbar, die nach August 2024 erstellt wurden, und wird sowohl in IPv4- als auch in IPv6-Adressen aufgelöst.

Nach August 2024 werden neue Schnittstellenendpunkte mit zwei DNS-Namen bereitgestellt, und Sie haben die Möglichkeit, den Dual-Stack-IP-Adresstyp zu wählen. Für vorhandene Endpunkte wird durch die Aktualisierung auf Dual-Stack eks.region.amazonaws.com so geändert, dass sowohl IPv4 als auch IPv6 unterstützt werden.

Verwendung des privaten DNS-Features

Nach der Konfiguration kann das private DNS-Feature in Ihre Workflows integriert werden und bietet Ihnen folgende Möglichkeiten:

- API-Anfragen: Verwenden Sie die standardmäßigen regionalen DNS-Namen, entweder`eks.region.amazonaws.com` oder `eks.region.api.aws`, basierend auf der Konfiguration Ihres Endpunkts, um API-Anfragen an Amazon EKS zu senden.
- Anwendungskompatibilität: Ihre vorhandenen Anwendungen, die EKS-APIs aufrufen, erfordern keine Änderungen, um dieses Feature zu nutzen.

- AWS-CLI mit Dual-Stack: Informationen zur Verwendung der Dual-Stack-Endpunkte mit der AWS-CLI finden Sie in der Konfiguration für Dual-Stack- und FIPS-Endpunkte im AWS-SDK- und Tools-Referenzhandbuch..
- Automatische Weiterleitung: Jeder Aufruf des Standard-Service-Endpunkts von Amazon EKS wird automatisch über den Schnittstellenendpunkt weitergeleitet, wodurch eine private und sichere Verbindung gewährleistet wird.

Ausfallsicherheit in Amazon-EKS-Clustern verstehen

Im Zentrum der globalen AWS-Infrastruktur stehen die AWS-Regionen und Availability Zones (Verfügbarkeitszonen, AZs). AWS Regionen stellen mehrere physisch getrennte und isolierte Availability Zones bereit, die mit Netzwerken mit geringer Latenz, hohem Durchsatz und hochredundanten Vernetzungen verbunden sind. Mithilfe von Availability Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch Failover zwischen Availability Zones ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind besser hoch verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

Amazon EKS führt und skaliert die Kubernetes-Steuerungsebene über mehrere AWS-Availability-Zones hinweg, um eine hohe Verfügbarkeit zu gewährleisten. Amazon EKS skaliert Steuerebene-Instances automatisch basierend auf der Last, erkennt und ersetzt fehlerhafte Steuerebene-Instances und patcht die Steuerebene automatisch. Nachdem Sie ein Versionsupdate eingeleitet haben, aktualisiert Amazon EKS Ihre Steuerebene für Sie und behält die hohe Verfügbarkeit der Steuerebene während des Updates bei.

Diese Steuerebene besteht aus mindestens zwei API-Server-Instances und drei `etcd`-Instances, die über drei Availability Zones innerhalb einer AWS-Region hinweg ausgeführt werden. Amazon EKS:

- Überwacht aktiv die Last auf Kontrollebeneninstanzen und skaliert sie automatisch, um eine hohe Leistung zu gewährleisten.
- Erkennt und ersetzt automatisch fehlerhafte Instances der Steuerebene und startet sie bei Bedarf in den Availability Zones innerhalb der AWS-Region neu.
- Nutzt die Architektur von AWS-Regionen, um eine hohe Verfügbarkeit zu gewährleisten. Aus diesem Grund ist Amazon EKS in der Lage, ein SLA für die Verfügbarkeit von API-Server-Endpunkten bereitzustellen.

Weitere Informationen über AWS-Regionen und -Availability Zones finden Sie unter Globale AWS-Infrastruktur.

Serviceübergreifende Vermeidung verwirrter Stellvertreter in Amazon EKS

Das Problem des verwirrten Stellvertreters ist ein Sicherheitsproblem, bei dem eine Entität, die keine Berechtigung zur Durchführung einer Aktion hat, eine privilegiertere Entität zur Durchführung der Aktion zwingen kann. In AWS, dienstübergreifender Identitätswechsel kann zum Problem des verwirrten Stellvertreters führen. Ein serviceübergreifender Identitätswechsel kann auftreten, wenn ein Service (der Anruf-Service) einen anderen Service anruft (den aufgerufenen Service). Der Aufruf-Service kann so manipuliert werden, dass er seine Berechtigungen verwendet, um auf die Ressourcen eines anderen Kunden zu reagieren, auf die er sonst nicht zugreifen dürfte. Um dies zu verhindern, AWS bietet Tools, mit denen Sie Ihre Daten für alle Dienste mit Dienstprinzipalen schützen können, denen Zugriff auf Ressourcen in Ihrem Konto gewährt wurde.

Wir empfehlen die Verwendung der globalen Bedingungskontextschlüssel `aws:SourceArn` und `aws:SourceAccount` in Ressourcenrichtlinien, um die Berechtigungen einzuschränken, die Amazon Elastic Kubernetes Service (Amazon EKS) einem anderen Service für die Ressource gewährt.

`aws:SourceArn`

Verwenden Sie `aws:SourceArn`, um nur einer Ressource den serviceübergreifenden Zugriff zuzuordnen.

`aws:SourceAccount`

Verwenden Sie `aws:SourceAccount`, um jede Ressource in diesem Konto der serviceübergreifenden Nutzung zuzuordnen.

Der effektivste Weg, um sich vor dem Confused-Deputy-Problem zu schützen, ist die Verwendung des globalen Bedingungskontext-Schlüssels `aws:SourceArn` mit dem vollständigen ARN der Ressource. Wenn Sie die vollständige ARN der Ressource nicht kennen oder mehrere Ressourcen angeben, verwenden Sie den globalen Kontextbedingungsschlüssel `aws:SourceArn` mit Platzhalterzeichen (*) für die unbekannten Teile der ARN. Beispiel, `arn:aws:<servicename>:*:<123456789012>:*`.

Wenn der `aws:SourceArn`-Wert die Konto-ID nicht enthält, z. B. einen Amazon-S3-Bucket-ARN, müssen Sie sowohl `aws:SourceAccount` als auch `aws:SourceArn` verwenden, um Berechtigungen einzuschränken.

Serviceübergreifende Vermeidung verwirrter Stellvertreter in Amazon-EKS-Cluster-Rollen

Für jeden Cluster ist eine IAM-Rolle für Amazon-EKS-Cluster erforderlich. Von Amazon EKS verwaltete Kubernetes-Cluster verwenden diese Rolle zur Verwaltung von Knoten, und der veraltete Cloud-Anbieter nutzt diese Rolle, um Load Balancer mit Elastic Load Balancing für Services zu erstellen. Diese Cluster-Aktionen können sich nur auf dasselbe Konto auswirken. Daher empfehlen wir, jede Cluster-Rolle auf diesen Cluster und dieses Konto zu beschränken. Dies ist eine spezifische Anwendung der AWS Empfehlung, in Ihrem Konto den Grundsatz der geringsten Rechte zu befolgen.

Format der Quell-ARN

Der Wert von `aws:SourceArn` muss die ARN eines EKS-Clusters im Format `arn:aws:eks:`*`region`*`:`*`account`*`:cluster/`*`cluster-name`* sein. Beispiel, `arn:aws: eks:us-west-2:123456789012:cluster/my-cluster`.

Format für Vertrauensrichtlinie für EKS-Cluster-Rollen

Das folgende Beispiel zeigt, wie Sie die globalen Bedingungskontextschlüssel `aws:SourceArn` und `aws:SourceAccount` für Amazon EKS verwenden können, um das Problem mit verwirrtem Stellvertreter zu verhindern.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "ArnLike": {
          "aws:SourceArn": "arn:aws:eks:us-west-2:123456789012:cluster/my-cluster"
          },
        "StringEquals": {
            "aws:SourceAccount": "123456789012"
        }
      }
    }
  ]
```

```
}
```

Sicherheitsüberlegungen für Kubernetes

Nachfolgend finden Sie Überlegungen zur Sicherheit in der Cloud, da diese sich auf Kubernetes in Amazon-EKS-Clustern auswirken. Eine ausführliche Übersicht über Sicherheitskontrollen und -praktiken in Kubernetes finden Sie unter Cloudnative Sicherheit und Kubernetes in der Kubernetes-Dokumentation.

Themen

- Sicherung von Workloads mit Kubernetes-Zertifikaten
- Die von Amazon EKS erstellten RBAC-Rollen und -Benutzer verstehen
- Verschlüsselung von Kubernetes-Geheimnissen mit KMS in vorhandenen Clustern
- Verwendung von AWS-Secrets-Manager-Geheimnissen mit Amazon-EKS-Pods
- Standardmäßige Umschlagverschlüsselung für alle Kubernetes-API-Daten

Sicherung von Workloads mit Kubernetes-Zertifikaten

Die Kubernetes-Zertifikats-API automatisiert die Bereitstellung von X.509-Anmeldeinformationen. Die API verfügt über eine Befehlszeilenschnittstelle, über die Kubernetes-API-Clients X.509-Zertifikate von einer Zertifizierungsstelle (CA) anfordern und abrufen kann. Sie können die `CertificateSigningRequest` (CSR)-Ressource verwenden, um anzufordern, dass ein benannter Unterzeichner das Zertifikat signiert. Ihre Anträge werden entweder genehmigt oder abgelehnt, bevor sie signiert werden. Kubernetes unterstützt sowohl integrierte Unterzeichner als auch benutzerdefinierte Unterzeichner mit klar definierten Verhaltensweisen. Auf diese Weise können Kunden vorhersehen, was mit ihren CSRs passiert. Weitere Informationen zur Zertifikatsignatur finden Sie unter Signieren von Anforderungen.

Einer der integrierten Unterzeichner ist `kubernetes.io/legacy-unknown`. Die `v1beta1`-API der CSR-Ressource berücksichtigte diesen Unterzeichnertyp „legacy-unknown“. Die stabile `v1`-API von CSR lässt jedoch nicht zu, dass `signerName` auf `kubernetes.io/legacy-unknown` festgelegt wird.

Wenn Sie Amazon EKS CA zum Generieren von Zertifikaten verwenden möchten, müssen Sie einen benutzerdefinierten Unterzeichner verwenden. Um die CSR-`v1`-API-Version zu verwenden

und ein neues Zertifikat zu generieren, müssen Sie alle vorhandenen Manifeste und API-Clients migrieren. Bestehende Zertifikate, die mit der vorhandenen `v1beta1`-API erstellt wurden, sind gültig und funktionieren, bis sie ablaufen. Diese umfasst die folgenden Funktionen:

- Vertrauensverteilung: keine. Es gibt kein Standardvertrauen oder eine Standardverteilung für diesen Unterzeichner in einem Kubernetes-Cluster.
- Zulässige Themen: beliebig
- Zulässige x509-Erweiterungen: Berücksichtigt SubjectAltName und Schlüsselnutzungserweiterungen und verwirft andere Erweiterungen
- Zulässige Schlüsselnutzung: Darf keine anderen Nutzungen als [„Schlüsselverschlüsselung“, „digitale Signatur“, „Serverauth“] enthalten

> **Note**
>
> Das Signieren von Clientzertifikaten wird nicht unterstützt.

- Ablauf/Zertifikatslebensdauer: 1 Jahr (Standard und Maximum)
- CA-Bit zulässig/unzulässig: unzulässig

Beispiel CSR-Erstellung mit signerName

Diese Schritte veranschaulichen, wie Sie ein Bereitstellungszertifikat für DNS-Namen `myserver.default.svc` mit `signerName: beta.eks.amazonaws.com/app-serving` erstellen. Verwenden Sie dies als Leitfaden für Ihre eigene Umgebung.

1. Führen Sie den Befehl `openssl genrsa -out myserver.key 2048` aus, um einen privaten RSA-Schlüssel zu erzeugen.

```
openssl genrsa -out myserver.key 2048
```

2. Führen Sie den folgenden Befehl aus, um eine Zertifikatsanforderung zu erstellen.

```
openssl req -new -key myserver.key -out myserver.csr -subj "/CN=myserver.default.svc"
```

3. Generieren Sie einen `base64`-Wert für die CSR-Variablen für die Verwendung in einem späteren Schritt.

```
base_64=$(cat myserver.csr | base64 -w 0 | tr -d "
```

```
")
```

4. Führen Sie den folgenden Befehl aus, um eine Datei mit dem Namen `mycsr.yaml` zu erstellen. Im folgenden Beispiel ist `beta.eks.amazonaws.com/app-serving` der `signerName`.

```
cat >mycsr.yaml <<EOF
apiVersion: certificates.k8s.io/v1
kind: CertificateSigningRequest
metadata:
  name: myserver
spec:
  request: $base_64
  signerName: beta.eks.amazonaws.com/app-serving
  usages:
    - digital signature
    - key encipherment
    - server auth
EOF
```

5. Reichen Sie die CSR ein.

```
kubectl apply -f mycsr.yaml
```

6. Genehmigen Sie das Bereitstellungszertifikat.

```
kubectl certificate approve myserver
```

7. Stellen Sie sicher, dass das Zertifikat ausgestellt wurde.

```
kubectl get csr myserver
```

   Eine Beispielausgabe sieht wie folgt aus.

```
NAME       AGE     SIGNERNAME                            REQUESTOR          CONDITION
myserver   3m20s   beta.eks.amazonaws.com/app-serving   kubernetes-admin
 Approved,Issued
```

8. Exportieren Sie das ausgestellte Zertifikat.

```
kubectl get csr myserver -o jsonpath='{.status.certificate}'| base64 -d >
 myserver.crt
```

Die von Amazon EKS erstellten RBAC-Rollen und -Benutzer verstehen

Wenn Sie einen Kubernetes-Cluster erstellen, werden für das ordnungsgemäße Funktionieren von Kubernetes mehrere Kubernetes-Standardidentitäten in diesem Cluster erstellt. Amazon EKS erstellt Kubernetes-Identitäten für jede seiner Standardkomponenten. Die Identitäten stellen eine rollenbasierte Autorisierungskontrolle (RBAC) von Kubernetes für die Cluster-Komponenten bereit. Weitere Informationen finden Sie unter Verwendung der RBAC-Autorisierung in der Kubernetes-Dokumentation.

Wenn Sie optionale Add-Ons zu Ihrem Cluster installieren, werden Ihrem Cluster möglicherweise zusätzliche Kubernetes-Identitäten hinzugefügt. Weitere Informationen zu Identitäten, die in diesem Thema nicht behandelt werden, finden Sie in der Dokumentation des Add-Ons.

Sie können die Liste der von Amazon EKS erstellten Kubernetes-Identitäten auf Ihrem Cluster mit dem `kubectl` Befehlszeilentool AWS-Managementkonsole oder anzeigen. Alle Benutzeridentitäten erscheinen in den `kube` Audit-Logs, die Ihnen über Amazon CloudWatch zur Verfügung stehen.

AWS-Managementkonsole

Voraussetzung

Der von Ihnen verwendete IAM-Prinzipal muss über die unter Erforderliche Berechtigungen beschriebenen Berechtigungen verfügen.

Um von Amazon EKS erstellte Identitäten anzuzeigen, verwenden Sie den AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie in der Liste Clusters (Cluster) den Cluster aus, der die anzuzeigenden Identitäten enthält.
3. Wählen Sie die Registerkarte Resources (Ressourcen) aus.
4. Wählen Sie unter Resource types (Ressourcentypen) die Option Authorization (Autorisierung) aus.
5. Wählen Sie ClusterRoles, ClusterRoleBindings, Rollen oder RoleBindings. Alle Ressourcen, denen eks vorangestellt ist, wurden von Amazon EKS erstellt. Weitere von Amazon EKS erstellte Identitätsressourcen sind:
   - Der ClusterRoleund ClusterRoleBindingbenannte aws-node. Die aws-node-Ressourcen unterstützen das Amazon-VPC-CNI-Plugin für Kubernetes, das Amazon EKS auf allen Clustern installiert.

- Ein ClusterRolebenannter vpc-resource-controller-roleund ein benannter. ClusterRoleBindingvpc-resource-controller-rolebinding Diese Ressourcen unterstützen den Amazon-VPC-Ressourcencontroller, den Amazon EKS auf allen Clustern installiert.

Zusätzlich zu den Ressourcen, die Sie in der Konsole sehen, sind die folgenden speziellen Benutzeridentitäten in Ihrem Cluster vorhanden, obwohl sie in der Konfiguration des Clusters nicht sichtbar sind:

- **`eks:cluster-bootstrap`** – Wird für `kubectl`-Vorgänge während des Cluster-Bootstraps verwendet.
- **`eks:support-engineer`** – Wird für Cluster-Management-Operationen verwendet.

6. Wählen Sie eine bestimmte Ressource aus, um Details dazu anzuzeigen. Standardmäßig werden Ihnen Informationen in der Strukturierten Ansicht angezeigt. In der oberen rechten Ecke der Detailseite können Sie die Raw view (Rohansicht) auswählen, um alle Informationen für die Ressource anzuzeigen.

Kubectl

Voraussetzung

Die Entität, die Sie verwenden (AWS Identity and Access Management (IAM) oder OpenID Connect (OIDC)), um die Kubernetes-Ressourcen auf dem Cluster aufzulisten, muss von IAM oder Ihrem OIDC-Identitätsanbieter authentifiziert werden. Der Entität müssen Berechtigungen zur Verwendung der Kubernetes-Verben `get` und `list` für die `Role`-, `ClusterRole`-, `RoleBinding`- und `ClusterRoleBinding`-Ressourcen in Ihrem Cluster gewährt werden, mit denen die Entität arbeiten soll. Weitere Informationen zum Gewähren von Zugriff auf IAM-Entitäten auf Ihren Cluster finden Sie unter the section called “Kubernetes-API-Zugriff”. Weitere Informationen zum Gewähren von Zugriff auf Ihren Cluster durch Entitäten, die von Ihrem eigenen OIDC-Anbieter authentifiziert wurden, finden Sie unter the section called “OIDC-Anbieter verknüpfen”.

So zeigen Sie von Amazon EKS erstellte Identitäten mit dem **`kubectl`** an

Führen Sie den Region aus, die Sie anzeigen möchten. Alle zurückgegebenen Ressourcen, denen eks vorangestellt ist, wurden von Amazon EKS erstellt. Zusätzlich zu den Ressourcen, die in der Ausgabe der Befehle zurückgegeben werden, sind die folgenden speziellen Benutzeridentitäten in Ihrem Cluster vorhanden, obwohl sie in der Konfiguration des Clusters nicht sichtbar sind:

- **`eks:cluster-bootstrap`** – Wird für `kubectl`-Vorgänge während des Cluster-Bootstraps verwendet.

- **`eks:support-engineer`** – Wird für Cluster-Management-Operationen verwendet.

ClusterRoles— `ClusterRoles` sind auf Ihren Cluster beschränkt, sodass jede für eine Rolle erteilte Berechtigung für Ressourcen in jedem Kubernetes-Namespace auf dem Cluster gilt.

Der folgende Befehl gibt alle von Amazon EKS erstellten Kubernetes `ClusterRoles` in Ihrem Cluster zurück.

```
kubectl get clusterroles | grep eks
```

Zusätzlich zu dem in der Ausgabe zurückgegebenen `ClusterRoles`, dem etwas vorangestellt ist, sind die folgenden `ClusterRoles` vorhanden.

- **`aws-node`** – Diese `ClusterRole` unterstützt das Amazon-VPC-CNI-Plugin für Kubernetes, das Amazon EKS in allen Clustern installiert.
- **`vpc-resource-controller-role`** – Diese `ClusterRole` unterstützt den Amazon-VPC-Ressourcen-Controller, den Amazon EKS in allen Clustern installiert.

Um die Spezifikation für a zu sehen`ClusterRole`, ersetzen Sie den folgenden Befehl durch einen, der *`eks:k8s-metrics`* in der Ausgabe des vorherigen Befehls `ClusterRole` zurückgegeben wurde. Das folgende Beispiel gibt die Spezifikation für die zurück *`eks:k8s-metrics`*`ClusterRole`.

```
kubectl describe clusterrole eks:k8s-metrics
```

Eine Beispielausgabe sieht wie folgt aus.

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources         Non-Resource URLs  Resource Names  Verbs
  ---------         -----------------  --------------  -----
                    [/metrics]         []              [get]
  endpoints         []                 []              [list]
  nodes             []                 []              [list]
  pods              []                 []              [list]
  deployments.apps  []                 []              [list]
```

ClusterRoleBindings— `ClusterRoleBindings` sind auf Ihren Cluster beschränkt.

Der folgende Befehl gibt alle von Amazon EKS erstellten Kubernetes `ClusterRoleBindings` in Ihrem Cluster zurück.

```
kubectl get clusterrolebindings | grep eks
```

Zusätzlich zu dem in der Ausgabe zurückgegebenen `ClusterRoleBindings`, sind die folgenden `ClusterRoleBindings` vorhanden.

- **`aws-node`** – Diese `ClusterRoleBinding` unterstützt das Amazon-VPC-CNI-Plugin für Kubernetes, das Amazon EKS in allen Clustern installiert.
- **`vpc-resource-controller-rolebinding`** – Diese `ClusterRoleBinding` unterstützt den Amazon-VPC-Ressourcen-Controller, den Amazon EKS in allen Clustern installiert.

Um die Spezifikation für a zu sehen`ClusterRoleBinding`, ersetzen Sie *`eks:k8s-metrics`* den folgenden Befehl durch einen, der in der Ausgabe des vorherigen Befehls `ClusterRoleBinding` zurückgegeben wurde. Das folgende Beispiel gibt die Spezifikation für die zurück *`eks:k8s-metrics`*`ClusterRoleBinding`.

```
kubectl describe clusterrolebinding eks:k8s-metrics
```

Eine Beispielausgabe sieht wie folgt aus.

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  ClusterRole
  Name:  eks:k8s-metrics
Subjects:
  Kind  Name             Namespace
  ----  ----             ---------
  User  eks:k8s-metrics
```

Rollen – `Roles` sind auf einen Kubernetes-Namespace beschränkt. Alle mit `Roles` erstellten Amazon EKS sind auf den `kube-system`-Namespace beschränkt.

Der folgende Befehl gibt alle von Amazon EKS erstellten Kubernetes `Roles` in Ihrem Cluster zurück.

```
kubectl get roles -n kube-system | grep eks
```

Um die Spezifikation für a zu sehen`Role`, ersetzen Sie *`eks:k8s-metrics`* den folgenden Befehl durch den Namen von a, der in der Ausgabe des vorherigen Befehls `Role` zurückgegeben wurde. Das folgende Beispiel gibt die Spezifikation für die zurück *`eks:k8s-metrics`*`Role`.

```
kubectl describe role eks:k8s-metrics -n kube-system
```

Eine Beispielausgabe sieht wie folgt aus.

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources         Non-Resource URLs  Resource Names              Verbs
  ---------         -----------------  --------------              -----
  daemonsets.apps   []                 [aws-node]                  [get]
  deployments.apps  []                 [vpc-resource-controller]  [get]
```

RoleBindings— `RoleBindings` sind auf einen Kubernetes-Namespace beschränkt. Alle mit `RoleBindings` erstellten Amazon EKS sind auf den `kube-system`-Namespace beschränkt.

Der folgende Befehl gibt alle von Amazon EKS erstellten Kubernetes `RoleBindings` in Ihrem Cluster zurück.

```
kubectl get rolebindings -n kube-system | grep eks
```

Um die Spezifikation für a zu sehen`RoleBinding`, ersetzen Sie den folgenden Befehl durch einen, der *`eks:k8s-metrics`* in der Ausgabe des vorherigen Befehls `RoleBinding` zurückgegeben wurde. Das folgende Beispiel gibt die Spezifikation für die zurück *`eks:k8s-metrics`*`RoleBinding`.

```
kubectl describe rolebinding eks:k8s-metrics -n kube-system
```

Eine Beispielausgabe sieht wie folgt aus.

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  Role
  Name:  eks:k8s-metrics
```

```
Subjects:
  Kind  Name             Namespace
  ----  ----             ---------
  User  eks:k8s-metrics
```

Verschlüsselung von Kubernetes-Geheimnissen mit KMS in vorhandenen Clustern

> **Important**
>
> Dieses Verfahren gilt nur für EKS-Cluster, auf denen Kubernetes Version 1.27 oder niedriger ausgeführt wird. Wenn Sie Kubernetes Version 1.28 oder höher ausführen, werden Ihre Kubernetes-Geheimnisse standardmäßig durch Umschlagverschlüsselung geschützt. Weitere Informationen finden Sie unter the section called “Standardmäßige Umschlagverschlüsselung für alle Kubernetes-API-Daten”.

Wenn Sie die Verschlüsselung von Geheimnissen aktivieren, werden die Kubernetes-Geheimnisse mit dem von Ihnen ausgewählten AWS KMS-Schlüssel verschlüsselt. Der KMS-Schlüssel muss die folgenden Bedingungen erfüllen:

- Symmetrisch
- Kann Daten verschlüsseln und entschlüsseln
- In derselben AWS Region wie der Cluster erstellt
- Wenn der KMS-Schlüssel in einem anderen Konto erstellt wurde, muss der IAM-Prinzipal Zugriff auf den KMS-Schlüssel haben.

Weitere Informationen finden Sie unter Zulassen, dass IAM-Prinzipale in anderen Konten einen KMS-Schlüssel verwenden können im AWS Key Management Service Developer Guide.

> **Warning**
>
> Sie können die Verschlüsselung von Geheimnissen nicht deaktivieren, nachdem Sie sie aktiviert haben. Diese Aktion ist unumkehrbar.

eksctl

Dieses Verfahren gilt nur für EKS-Cluster, auf denen Kubernetes Version 1.27 oder niedriger ausgeführt wird. Weitere Informationen finden Sie unter the section called “Standardmäßige Umschlagverschlüsselung für alle Kubernetes-API-Daten”.

Sie können die Verschlüsselung auf zwei Arten aktivieren:

- Fügen Sie Ihrem Cluster mit einem einzigen Befehl Verschlüsselung hinzu.

  Um Ihre Secrets automatisch erneut zu verschlüsseln, führen Sie den folgenden Befehl aus.

  ```
  eksctl utils enable-secrets-encryption \
      --cluster my-cluster \
      --key-arn arn:aws: kms:region-code:account:key/key
  ```

  Um die automatische Neuverschlüsselung Ihrer Secrets zu deaktivieren, führen Sie den folgenden Befehl aus.

  ```
  eksctl utils enable-secrets-encryption
      --cluster my-cluster \
      --key-arn arn:aws: kms:region-code:account:key/key \
      --encrypt-existing-secrets=false
  ```

- Fügen Sie Ihrem Cluster eine Verschlüsselung mit einer `kms-cluster.yaml`-Datei hinzu.

  ```
  apiVersion: eksctl.io/v1alpha5
  kind: ClusterConfig

  metadata:
    name: my-cluster
    region: region-code

  secretsEncryption:
    keyARN: arn:aws: kms:region-code:account:key/key
  ```

  Um Ihre Secrets automatisch erneut zu verschlüsseln, führen Sie den folgenden Befehl aus.

  ```
  eksctl utils enable-secrets-encryption -f kms-cluster.yaml
  ```

Um die automatische Neuverschlüsselung Ihrer Secrets zu deaktivieren, führen Sie den folgenden Befehl aus.

```
eksctl utils enable-secrets-encryption -f kms-cluster.yaml --encrypt-existing-
secrets=false
```

AWS-Managementkonsole

a. Dieses Verfahren gilt nur für EKS-Cluster, auf denen Kubernetes Version 1.27 oder niedriger ausgeführt wird. Weitere Informationen finden Sie unter the section called “Standardmäßige Umschlagverschlüsselung für alle Kubernetes-API-Daten”.
b. Öffnen Sie die Amazon-EKS-Konsole.
c. Wählen Sie den Cluster aus, dem Sie die KMS-Verschlüsselung hinzufügen möchten.
d. Wählen Sie die Registerkarte Overview (Übersicht) aus (sie ist standardmäßig ausgewählt).
e. Scrollen Sie nach unten zum Abschnitt Secrets encryption (Secrets-Verschlüsselung) und wählen Sie Enable (Aktivieren) aus.
f. Wählen Sie einen Schlüssel in der Dropdown-Liste und dann die Schaltfläche Enable (Aktivieren) aus. Wenn keine Schlüssel aufgeführt sind, müssen Sie zuerst einen erstellen. Weitere Informationen finden Sie unter Erstellen von Schlüsseln
g. Wählen Sie die Schaltfläche Confirm (Bestätigen) aus, um den ausgewählten Schlüssel zu verwenden.

AWS CLI

a. Dieses Verfahren gilt nur für EKS-Cluster, auf denen Kubernetes Version 1.27 oder niedriger ausgeführt wird. Weitere Informationen finden Sie unter the section called “Standardmäßige Umschlagverschlüsselung für alle Kubernetes-API-Daten”.
b. Ordnen Sie die Secrets-Verschlüsselungskonfiguration mithilfe des folgenden AWS CLI-Befehls Ihrem Cluster zu. Ersetzen Sie die Beispielwerte durch Ihre eigenen Werte.

```
aws eks associate-encryption-config \
    --cluster-name my-cluster \
    --encryption-config '[{"resources":["secrets"],"provider":
{"keyArn":"arn:aws: kms:region-code:account:key/key"}}]'
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
```

```
  "update": {
    "id": "3141b835-8103-423a-8e68-12c2521ffa4d",
    "status": "InProgress",
    "type": "AssociateEncryptionConfig",
    "params": [
      {
        "type": "EncryptionConfig",
        "value": "[{\"resources\":[\"secrets\"],\"provider\":{\"keyArn\":
\"arn:aws: kms:region-code:account:key/key\"}}]"
      }
    ],
    "createdAt": 1613754188.734,
    "errors": []
  }
}
```

c. Sie können den Status Ihrer Verschlüsselungsaktualisierung mit dem folgenden Befehl überwachen. Verwenden Sie den `cluster name` und die `update ID`, die in der vorherigen Ausgabe zurückgegeben wurden. Wenn der Status `Successful` angezeigt wird, ist das Update abgeschlossen.

```
aws eks describe-update \
    --region region-code \
    --name my-cluster \
    --update-id 3141b835-8103-423a-8e68-12c2521ffa4d
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
  "update": {
    "id": "3141b835-8103-423a-8e68-12c2521ffa4d",
    "status": "Successful",
    "type": "AssociateEncryptionConfig",
    "params": [
      {
        "type": "EncryptionConfig",
        "value": "[{\"resources\":[\"secrets\"],\"provider\":{\"keyArn\":
\"arn:aws: kms:region-code:account:key/key\"}}]"
      }
    ],
    "createdAt": 1613754188.734>,
    "errors": []
```

```
  }
}
```

d. Führen Sie den Befehl `describe-cluster` aus, um zu überprüfen, ob die Verschlüsselung in Ihrem Cluster aktiviert ist. Die Antwort enthält eine `EncryptionConfig`-Zeichenfolge.

```
aws eks describe-cluster --region region-code --name my-cluster
```

Nachdem Sie die Verschlüsselung in Ihrem Cluster aktiviert haben, müssen Sie alle vorhandenen Secrets mit dem neuen Schlüssel verschlüsseln:

> **Note**
>
> Wenn Sie `eksctl` verwenden, ist die Ausführung des folgenden Befehls nur dann erforderlich, wenn Sie die automatische Neuverschlüsselung Ihrer Secrets deaktivieren.

```
kubectl get secrets --all-namespaces -o json | kubectl annotate --overwrite -f - kms-encryption-timestamp="time value"
```

> **Warning**
>
> Wenn Sie die Geheimnisverschlüsselung für einen vorhandenen Cluster aktivieren und der von Ihnen verwendete KMS-Schlüssel jemals gelöscht wird, können Sie den Cluster nicht wiederherstellen. Wenn Sie den KMS-Schlüssel löschen, wird der Cluster dauerhaft auf einen degradierten Zustand festgelegt. Weitere Informationen finden Sie unter Löschen von AWS KMS-Schlüsseln.

> **Note**
>
> Standardmäßig erstellt der `create-key` Befehl einen KMS-Schlüssel mit symmetrischer Verschlüsselung und einer Schlüsselrichtlinie, die dem Konto Root-Administratorzugriff auf AWS KMS-Aktionen und -Ressourcen gewährt. Wenn Sie die Berechtigungen begrenzen möchten, müssen Sie sicherstellen, dass die Aktionen `kms:DescribeKey` und `kms:CreateGrant` für die Richtlinie für den Prinzipal zulässig sind, der die API `create-cluster` aufruft.

Für Cluster, die KMS-Umschlagverschlüsselung verwenden, sind `kms:CreateGrant`-Berechtigungen erforderlich. Die Bedingung `kms:GrantIsForAWSResource` wird für die CreateCluster Aktion nicht unterstützt und sollte nicht in KMS-Richtlinien zur Steuerung der `kms:CreateGrant` Berechtigungen von Benutzern verwendet werden, die diese Aktion ausführen CreateCluster.

Verwendung von AWS-Secrets-Manager-Geheimnissen mit Amazon-EKS-Pods

Um Secrets aus Secrets Manager und Parameter aus Parameter Store als in Amazon-EKS-Pods bereitgestellte Dateien anzuzeigen, können Sie den AWS Secrets and Configuration Provider (ASCP) für die Kubernetes-Secrets-Store-CSI-Treiber verwenden.

Mit dem ASCP können Sie Ihre Secrets in Secrets Manager speichern und verwalten und diese dann über Ihre auf Amazon EKS ausgeführten Workloads abrufen. Sie können IAM-Rollen und -Richtlinien verwenden, um den Zugriff auf Ihre Secrets auf bestimmte Kubernetes-Pods in einem Cluster zu beschränken. Der ASCP ruft die Pod Identity ab und tauscht die Identität gegen eine IAM-Rolle. Der ASCP übernimmt die IAM-Rolle des Pods und kann dann Geheimnisse aus Secrets Manager abrufen, die für diese Rolle autorisiert sind.

Wenn Sie in Secrets Manager die automatische Rotation für Ihre Secrets verwenden, können Sie auch die Secrets-Store-CSI-Treiberrotation verwenden, um sicherzustellen, dass Sie das neueste Secret aus Secrets Manager abrufen.

> **Note**
>
> AWS Fargate (Fargate)-Knotengruppen werden nicht unterstützt.

Weitere Informationen finden Sie unter Verwenden von Secrets-Manager-Secrets in Amazon EKS im AWS-Secrets Manager-Benutzerhandbuch.

Standardmäßige Umschlagverschlüsselung für alle Kubernetes-API-Daten

Amazon Elastic Kubernetes Service (Amazon EKS) bietet standardmäßige Umschlagverschlüsselung für alle Kubernetes-API-Daten in EKS-Clustern, in denen Kubernetes Version 1.28 oder höher ausgeführt wird.

Die Umschlagverschlüsselung schützt die Daten, die Sie auf dem Kubernetes-API-Server speichern. Die Umschlagverschlüsselung gilt beispielsweise für die Konfiguration Ihres Kubernetes-Clusters, z. B. `ConfigMaps`. Die Umschlagverschlüsselung gilt nicht für Daten auf Knoten oder EBS-Volumes. EKS unterstützte zuvor die Verschlüsselung von Kubernetes-Geheimnissen, und nun erstreckt sich diese Umschlagverschlüsselung auf alle Kubernetes-API-Daten.

Dies bietet eine verwaltete Standarderfahrung, die defense-in-depth für Ihre Kubernetes-Anwendungen implementiert wird und keine Maßnahmen von Ihrer Seite erfordert.

Amazon EKS verwendet AWS Key Management Service (KMS) mit Kubernetes KMS Provider v2 für diese zusätzliche Sicherheitsebene mit einem Amazon Web Services Services-eigenen Schlüssel und der Option, dass Sie Ihren eigenen vom Kunden verwalteten Schlüssel (CMK) von KMS verwenden können. AWS

Grundlegendes zur Umschlagverschlüsselung

Bei der Umschlagverschlüsselung werden Klartextdaten mit einem Datenverschlüsselungsschlüssel (DEK) verschlüsselt, bevor sie an den Datenspeicher (etcd) gesendet werden. Anschließend werden die DEK mit einem KMS-Stammschlüssel verschlüsselt, der in einem zentral verwalteten Remote-KMS-System (AWS KMS) gespeichert ist. Dies ist eine defense-in-depth Strategie, da sie die Daten mit einem Verschlüsselungsschlüssel (DEK) schützt und dann eine weitere Sicherheitsebene hinzufügt, indem dieser DEK mit einem separaten, sicher gespeicherten Verschlüsselungsschlüssel geschützt wird, der als Key Encryption Key (KEK) bezeichnet wird.

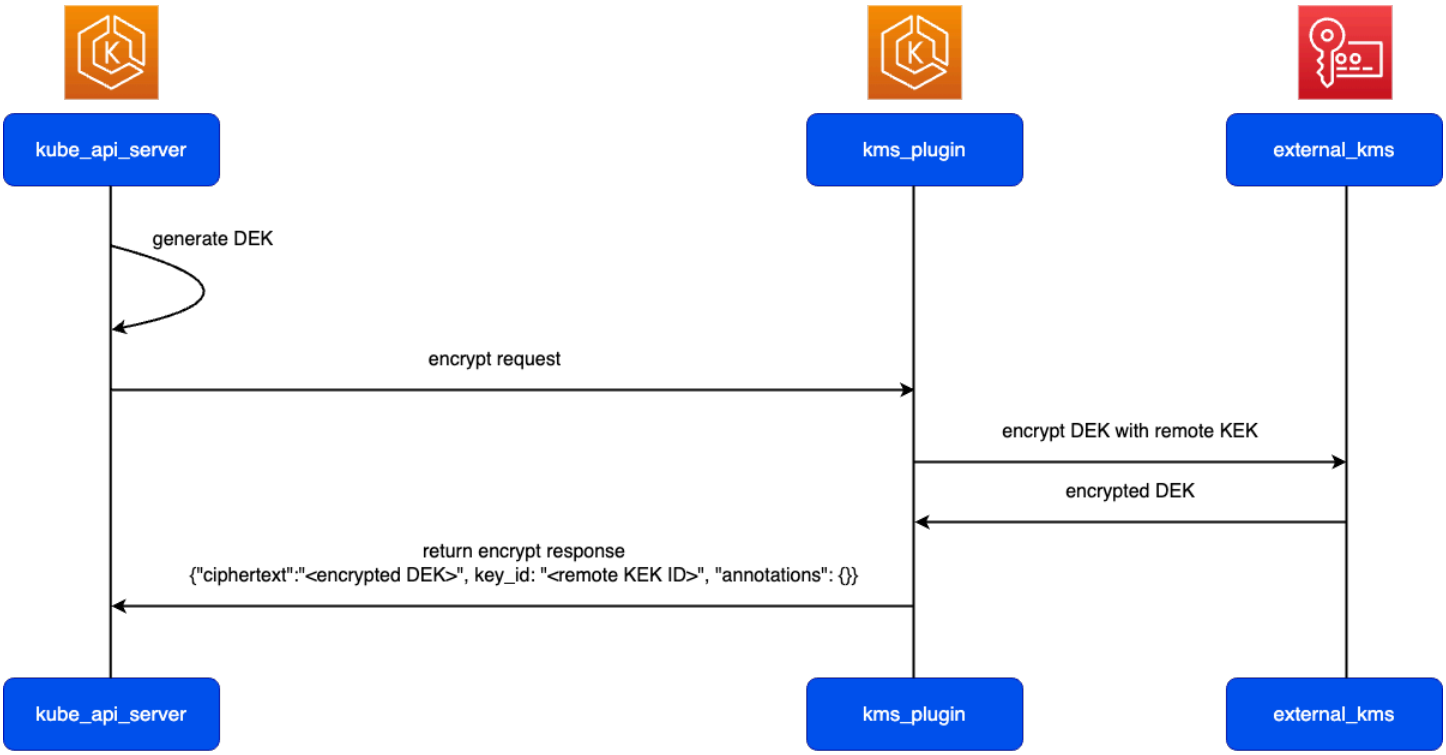
So ermöglicht Amazon EKS die standardmäßige Envelope-Verschlüsselung mit KMS v2 und AWS KMS

Amazon EKS nutzt KMS v2, um eine standardmäßige Umschlagverschlüsselung für alle API-Daten in der verwalteten Kubernetes-Steuerebene zu implementieren, bevor diese in der etcd-Datenbank gespeichert werden. Beim Startup generiert der Cluster-API-Server einen Datenverschlüsselungsschlüssel (DEK) aus einem geheimen Startup-Wert in Kombination mit zufällig generierten Daten. Außerdem ruft der API-Server beim Start das KMS-Plug-In auf, um den DEK-Seed mit einem Remote Key Encryption Key (KEK) von AWS KMS zu verschlüsseln. Dies ist ein einmaliger Aufruf, der beim Startup des API-Servers und bei der KEK-Rotation ausgeführt wird. Der API-Server speichert dann den verschlüsselten DEK-Seed im Cache. Danach verwendet der API-Server den zwischengespeicherten DEK-Seed, um auf der DEKs Grundlage einer Key Derivation Function (KDF) weitere einmalige Verwendungszwecke zu generieren. Jede dieser generierten Ressourcen DEKs wird dann nur einmal verwendet, um eine einzelne Kubernetes-Ressource zu

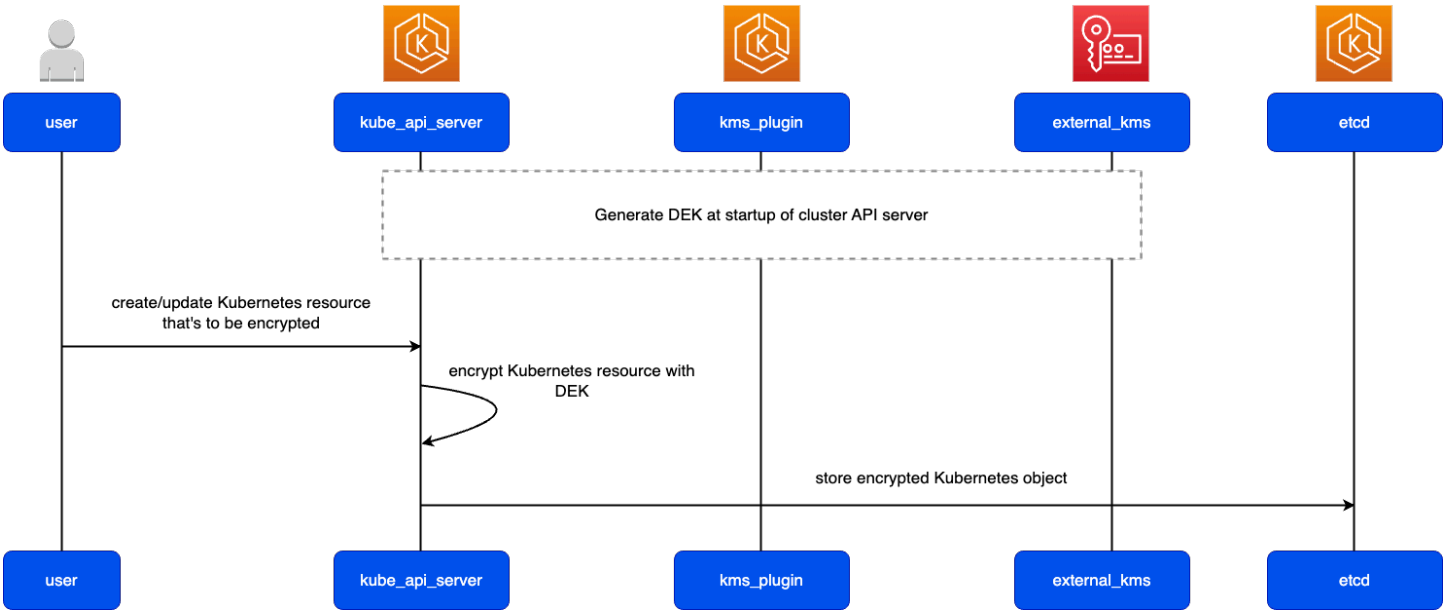
verschlüsseln, bevor sie in etcd gespeichert wird. Durch die Verwendung eines verschlüsselten zwischengespeicherten DEK-Seeds in KMS v2 ist der Prozess der Verschlüsselung von Kubernetes-Ressourcen im API-Server sowohl leistungsfähiger als auch kostengünstiger.

Standardmäßig gehört dieser KEK AWS, aber Sie können optional Ihren eigenen von KMS mitbringen. AWS

Das nachfolgende Diagramm veranschaulicht die Generierung und Verschlüsselung eines DEK beim Startup des API-Servers.



Das folgende allgemeine Diagramm veranschaulicht die Verschlüsselung einer Kubernetes-Ressource, bevor sie in etcd gespeichert wird.



Häufig gestellte Fragen

Inwiefern verbessert die standardmäßige Umschlagverschlüsselung die Sicherheitslage meines EKS-Clusters?

Dieses Feature reduziert die Oberfläche und den Zeitraum, in dem Metadaten und Kundeninhalte unverschlüsselt sind. Mit der standardmäßigen Umschlagverschlüsselung befinden sich Metadaten und Kundeninhalte nur vorübergehend in einem unverschlüsselten Zustand im Speicher des kube-apiserver, bevor sie in etcd gespeichert werden. Der Speicher des kube-apiserver wird durch das Nitro-System gesichert. Amazon EKS verwendet nur Nitro-basierte EC2 Instances für die verwaltete Kubernetes-Steuerebene. Diese Instances verfügen über Sicherheitskontrollmechanismen, die verhindern, dass Systeme oder Personen auf ihren Speicher zugreifen können.

Welche Version von Kubernetes muss ich ausführen, um dieses Feature nutzen zu können?

Damit die standardmäßige Umschlagverschlüsselung aktiviert werden kann, muss auf Ihrem Amazon-EKS-Cluster Kubernetes Version 1.28 oder höher ausgeführt werden.

Sind meine Daten noch sicher, wenn ich eine Kubernetes-Cluster-Version verwende, die dieses Feature nicht unterstützt?

Ja. Sicherheit hat AWS bei uns höchste Priorität. Unsere gesamte digitale Transformation und Innovation basiert auf den höchsten Sicherheitsstandards, und wir sind bestrebt, diese Standards kontinuierlich zu verbessern.

Alle in etcd gespeicherten Daten werden für jeden EKS-Cluster auf Festplattenebene verschlüsselt, unabhängig von der verwendeten Kubernetes-Version. EKS verwendet Root-Schlüssel, die Volume-Verschlüsselungsschlüssel generieren, die vom EKS- Service verwaltet werden. Darüber hinaus wird jeder Amazon-EKS-Cluster in einer isolierten VPC mit clusterspezifischen virtuellen Rechnern ausgeführt. Aufgrund dieser Architektur und unserer Praktiken im Bereich der Betriebssicherheit hat Amazon EKS mehrere Compliance-Bewertungen und -Standards erreicht, darunter SOC 1, 2, 3, PCI-DSS, ISO und HIPAA-Konformität. Diese Compliance-Bewertungen und -Standards gelten für alle EKS-Cluster, unabhängig davon, ob sie über eine standardmäßige Umschlagverschlüsselung verfügen oder nicht.

Wie funktioniert die Umschlagverschlüsselung in Amazon EKS?

Beim Startup generiert der Cluster-API-Server einen Datenverschlüsselungsschlüssel (DEK) aus einem geheimen Startup-Wert in Kombination mit zufällig generierten Daten. Außerdem ruft der API-Server beim Start das KMS-Plug-In auf, um den DEK mit einem Remote Key Encryption Key (KEK) von AWS KMS zu verschlüsseln. Dies ist ein einmaliger Aufruf, der beim Startup des API-Servers und bei der KEK-Rotation ausgeführt wird. Der API-Server speichert dann den verschlüsselten DEK-Seed im Cache. Danach verwendet der API-Server den zwischengespeicherten DEK-Seed, um auf der DEKs Grundlage einer Key Derivation Function (KDF) weitere einmalige Verwendungszwecke zu generieren. Jede dieser generierten Ressourcen DEKs wird dann nur einmal verwendet, um eine einzelne Kubernetes-Ressource zu verschlüsseln, bevor sie in etcd gespeichert wird.

Es ist wichtig zu beachten, dass vom API-Server zusätzliche Aufrufe getätigt werden, um den Zustand und die normale Funktionalität der KMS-Integration zu überprüfen. AWS Diese zusätzlichen Gesundheitschecks sind in Ihrem sichtbar AWS CloudTrail.

Muss ich irgendwelche Maßnahmen ergreifen oder Berechtigungen ändern, damit dieses Feature in meinem EKS-Cluster funktioniert?

Nein, Sie müssen keine Maßnahmen ergreifen. Umschlagverschlüsselung in Amazon EKS ist nun eine Standardkonfiguration, die in allen Clustern mit Kubernetes Version 1.28 oder höher aktiviert ist. Die AWS KMS-Integration wird durch den Kubernetes-API-Server eingerichtet, der von verwaltet wird. AWS Das bedeutet, dass Sie keine Berechtigungen konfigurieren müssen, um die KMS-Verschlüsselung für Ihren Cluster zu verwenden.

Wie kann ich feststellen, ob die standardmäßige Umschlagverschlüsselung in meinem Cluster aktiviert ist?

Wenn Sie auf die Verwendung Ihres eigenen CMK umstellen, wird Ihnen die ARN des mit Ihrem Cluster verknüpften KMS-Schlüssels angezeigt. Darüber hinaus können Sie die AWS CloudTrail Ereignisprotokolle anzeigen, die mit der Verwendung des CMK Ihres Clusters verknüpft sind.

Wenn Ihr Cluster einen AWS eigenen Schlüssel verwendet, wird dies in der EKS-Konsole detailliert beschrieben (mit Ausnahme des ARN des Schlüssels).

Kann ich AWS auf den AWS eigenen Schlüssel zugreifen, der für die Standard-Umschlagverschlüsselung in Amazon EKS verwendet wird?

Nein. AWS verfügt über strenge Sicherheitskontrollen in Amazon EKS, die verhindern, dass Personen auf Klartext-Verschlüsselungsschlüssel zugreifen, die zur Sicherung von Daten in der etcd-Datenbank verwendet werden. Diese Sicherheitsmaßnahmen werden auch auf den AWS eigenen KMS-Schlüssel angewendet.

Ist die standardmäßige Umschlagverschlüsselung in meinem vorhandenen EKS-Cluster aktiviert?

Wenn Sie einen Amazon-EKS-Cluster mit Kubernetes-Version 1.28 oder höher ausführen, ist die Umschlagverschlüsselung aller Kubernetes-API-Daten aktiviert. Für bestehende Cluster verwendet Amazon EKS den `eks:kms-storage-migrator` RBAC, ClusterRole um Daten, die zuvor nicht in etcd umhüllt waren, in diesen neuen Verschlüsselungsstatus zu migrieren.

Was bedeutet das, wenn ich die Umschlagverschlüsselung für Geheimnisse in meinem EKS-Cluster bereits aktiviert habe?

Wenn Sie über einen vorhandenen kundenseitig verwalteten Schlüssel (CMK) in KMS verfügen, der zur Umschlagverschlüsselung Ihrer Kubernetes-Geheimnisse verwendet wurde, wird derselbe Schlüssel als KEK für die Umschlagverschlüsselung aller Kubernetes-API-Datentypen in Ihrem Cluster verwendet.

Entstehen zusätzliche Kosten für den Betrieb eines EKS-Clusters mit standardmäßiger Umschlagverschlüsselung?

Es fallen keine zusätzlichen Kosten für die verwaltete Kubernetes-Steuerebene an, wenn Sie einen Schlüssel von Amazon Web Services für die standardmäßige Umschlagverschlüsselung verwenden. Standardmäßig verwendet jeder EKS-Cluster, auf dem Kubernetes Version 1.28 oder höher ausgeführt wird, einen Schlüssel von Amazon Web Services. Wenn Sie jedoch Ihren eigenen AWS KMS-Schlüssel verwenden, gelten die normalen KMS-Preise.

Wie viel kostet es, meinen eigenen AWS KMS-Schlüssel zur Verschlüsselung von Kubernetes-API-Daten in meinem Cluster zu verwenden?

Sie zahlen 1 USD pro Monat für die Speicherung aller benutzerdefinierten Schlüssel, die Sie erstellen oder in KMS importieren. KMS berechnet Gebühren für Verschlüsselungs- und Entschlüsselungsanfragen. Es gibt ein kostenloses Kontingent von 20 000 Anfragen pro Monat und Konto. Für 10 000 Anfragen über das kostenlose Kontingent hinaus zahlen Sie 0,03 USD pro Monat. Dies gilt für die gesamte KMS-Nutzung für ein Konto, sodass die Kosten für die Verwendung Ihres eigenen AWS KMS-Schlüssels in Ihrem Cluster durch die Verwendung dieses Schlüssels in anderen Clustern oder AWS Ressourcen innerhalb Ihres Kontos beeinflusst werden.

Werden meine KMS-Gebühren jetzt höher sein, da mein kundenseitig verwalteter Schlüssel (CMK) zur Umschlagverschlüsselung aller Kubernetes-API-Daten und nicht nur von Geheimnissen verwendet wird?

Nein. Unsere Implementierung mit KMS v2 reduziert die Anzahl der Anrufe an AWS KMS erheblich. Dies wird wiederum die mit Ihrem CMK verbundenen Kosten senken, unabhängig davon, ob zusätzliche Kubernetes-Daten in Ihrem EKS-Cluster verschlüsselt oder entschlüsselt werden.

Wie oben beschrieben, wird der generierte DEK-Seed, der für die Verschlüsselung von Kubernetes-Ressourcen verwendet wird, lokal im Cache des Kubernetes-API-Servers gespeichert, nachdem er mit dem KEK aus der Ferne verschlüsselt wurde. Wenn sich der verschlüsselte DEK-Seed nicht im Cache des API-Servers befindet, ruft der API-Server AWS KMS auf, um den DEK-Seed zu verschlüsseln. Der API-Server speichert dann den verschlüsselten DEK-Seed für die zukünftige Verwendung im Cluster im Cache, ohne KMS aufzurufen. In ähnlicher Weise ruft der API-Server bei Entschlüsselungsanforderungen AWS KMS für die erste Entschlüsselungsanforderung auf. Danach wird der entschlüsselte DEK-Seed zwischengespeichert und für future Entschlüsselungsvorgänge verwendet.

Weitere Informationen finden Sie unter KEP-3299: KMS v2-Verbesserungen in den Kubernetes-Erweiterungen unter. GitHub

Kann ich denselben CMK-Schlüssel für mehrere Amazon-EKS-Cluster verwenden?

Ja. Um einen Schlüssel erneut zu verwenden, können Sie ihn einem Cluster in derselben Region zuordnen, indem Sie die ARN bei der Erstellung dem Cluster zuweisen. Wenn Sie jedoch denselben CMK für mehrere EKS-Cluster verwenden, sollten Sie die erforderlichen Maßnahmen ergreifen, um eine willkürliche Deaktivierung des CMK zu verhindern. Andernfalls hat ein deaktivierter CMK, der mehreren EKS-Clustern zugeordnet ist, je nach Schlüssel einen größeren Einflussbereich auf die Cluster.

Was geschieht mit meinem EKS-Cluster, wenn mein CMK nach der Aktivierung der standardmäßigen Umschlagverschlüsselung nicht mehr verfügbar ist?

Wenn Sie einen KMS-Schlüssel deaktivieren, kann dieser für keine kryptografischen Vorgänge mehr verwendet werden Ohne Zugriff auf einen vorhandenen CMK kann der API-Server keine neu erstellten Kubernetes-Objekte verschlüsseln und speichern sowie zuvor verschlüsselte Kubernetes-Objekte, die in etcd gespeichert sind, entschlüsseln Wenn der CMK deaktiviert ist, wird der Cluster sofort in einen unhealthy/degraded Zustand versetzt, in dem wir unsere Serviceverpflichtung erst erfüllen können, wenn Sie den zugehörigen CMK erneut aktivieren.

Wenn ein CMK deaktiviert wird, erhalten Sie Benachrichtigungen über den beeinträchtigten Zustand Ihres EKS-Clusters und die Notwendigkeit, Ihr CMK innerhalb von 30 Tagen nach der Deaktivierung wieder zu aktivieren. Damit können Sie die erfolgreiche Wiederherstellung Ihrer Kubernetes-Steuerebenen-Ressourcen sicherstellen.

Wie kann ich meinen EKS-Cluster vor den Auswirkungen eines CMK schützen? disabled/deleted

Um Ihre EKS-Cluster vor solchen Vorfällen zu schützen, sollten Ihre Schlüsseladministratoren den Zugriff auf KMS-Schlüsseloperationen mithilfe von IAM-Richtlinien nach dem Prinzip der geringsten Berechtigungen verwalten. Dadurch wird das Risiko einer willkürlichen Deaktivierung oder Löschung von Schlüsseln im Zusammenhang mit EKS-Clustern verringert. Darüber hinaus können Sie einen CloudWatch Alarm einrichten, um über den Status Ihres CMK informiert zu werden.

Wird mein EKS-Cluster wiederhergestellt, wenn ich den CMK erneut aktiviere?

Um eine erfolgreiche Wiederherstellung Ihres EKS-Clusters zu gewährleisten, empfehlen wir nachdrücklich, Ihren CMK innerhalb der ersten 30 Tage nach seiner Deaktivierung wieder zu aktivieren. Die erfolgreiche Wiederherstellung Ihres EKS-Clusters hängt jedoch auch davon ab, ob er aufgrund eines automatischen Kubernetes-Upgrades, das möglicherweise stattfindet, während sich der Cluster in einem Status befindet, Änderungen erfährt, die die API beeinträchtigen. unhealthy/degraded

Warum wird mein EKS-Cluster nach der Deaktivierung des CMK in einen unhealthy/degraded Zustand versetzt?

Der API-Server der EKS-Steuerebene verwendet einen DEK-Schlüssel, der verschlüsselt und im Speicher des API-Servers zwischengespeichert wird, um alle Objekte während des create/update Betriebs zu verschlüsseln, bevor sie in etcd gespeichert werden. Wenn ein vorhandenes Objekt aus etcd abgerufen wird, verwendet der API-Server denselben zwischengespeicherten DEK-

Schlüssel und entschlüsselt das Kubernetes-Ressourcenobjekt. Wenn Sie den CMK deaktivieren, hat dies keine unmittelbaren Auswirkungen auf den API-Server, da der DEK-Schlüssel im Speicher des API-Servers zwischengespeichert ist. Wenn die API-Serverinstanz jedoch neu gestartet wird, hat sie kein zwischengespeichertes DEK und muss AWS KMS für Verschlüsselungs- und Entschlüsselungsvorgänge aufrufen. Ohne CMK schlägt dieser Vorgang mit dem Fehlercode KMS_KEY_DISABLED fehl, wodurch der erfolgreiche Start des API-Servers verhindert wird.

Was passiert mit meinem EKS-Cluster, wenn ich meinen CMK lösche?

Das Löschen des zugehörigen CMK-Schlüssels Ihres EKS-Clusters beeinträchtigt dessen Integrität irreparabel. Ohne den CMK Ihres Clusters kann der API-Server keine neuen Kubernetes-Objekte mehr verschlüsseln und speichern sowie zuvor verschlüsselte Kubernetes-Objekte, die in der etcd-Datenbank gespeichert sind, nicht mehr entschlüsseln. Sie sollten mit dem Löschen eines CMK-Schlüssels für Ihren EKS-Cluster nur fortfahren, wenn Sie sicher sind, dass Sie den EKS-Cluster nicht mehr benötigen.

Beachten Sie, dass Ihr Cluster nicht wiederhergestellt werden kann, wenn der CMK nicht gefunden wird (KMS_KEY_NOT_FOUND) oder die Berechtigungen für den mit Ihrem Cluster verknüpften CMK widerrufen werden (KMS_GRANT_REVOKED). Weitere Informationen zum Clusterstatus und zu Fehlercodes finden Sie unter Clusterintegritäts FAQs - und Fehlercodes mit Lösungspfaden.

Fallen mir trotzdem Gebühren für einen degraded/unhealthy EKS-Cluster an, weil ich meinen CMK deaktiviert oder gelöscht habe?

Ja. Obwohl die EKS-Kontrollebene im Falle eines deaktivierten CMK nicht verwendet werden kann, AWS werden weiterhin dedizierte Infrastrukturressourcen ausgeführt, die dem EKS-Cluster zugewiesen sind, bis sie vom Kunden gelöscht werden. Darüber hinaus gilt unsere Serviceverpflichtung in einem solchen Fall nicht, da es sich um eine freiwillige Handlung oder Unterlassung des Kunden handelt, die den normalen Zustand und Betrieb Ihres EKS-Clusters beeinträchtigt.

Kann mein EKS-Cluster automatisch aktualisiert werden, wenn er sich aufgrund eines deaktivierten CMK in einem unhealthy/degraded Zustand befindet?

Ja. Wenn Ihr Cluster jedoch über einen deaktivierten CMK verfügt, haben Sie 30 Tage Zeit, ihn wieder zu aktivieren In diesem Zeitraum von 30 Tagen wird Ihr Kubernetes-Cluster nicht automatisch aktualisiert. Wenn dieser Zeitraum jedoch abläuft und Sie den CMK nicht wieder aktiviert haben, wird der Cluster gemäß dem Kubernetes-Versionslebenszyklus in EKS automatisch auf die nächste Version (n+1) aktualisiert, die im Standard-Support enthalten ist.

Wir empfehlen nachdrücklich, einen deaktivierten CMK umgehend wieder zu aktivieren, sobald Sie einen betroffenen Cluster bemerken. Es ist wichtig zu beachten, dass EKS diese betroffenen Cluster zwar automatisch aktualisiert, jedoch keine Garantie für eine erfolgreiche Wiederherstellung besteht. Dies trifft insbesondere dann zu, wenn der Cluster mehrere automatische Upgrades durchläuft, da dies Änderungen an der Kubernetes-API und unerwartetes Verhalten im Bootstrap-Prozess des API-Servers zur Folge haben kann.

Kann ich einen KMS-Schlüssel-Alias verwenden?

Ja. Amazon EKS unterstützt die Verwendung von KMS-Schlüssel-Aliasen. Ein Alias ist ein benutzerfreundlicher Name für einen KMS-Schlüssel für Amazon Web Service. Mit einem Alias können Sie beispielsweise auf einen KMS-Schlüssel als my-key anstelle von **`1234abcd-12ab-34cd-56ef-1234567890ab`** verweisen.

Kann ich meine Cluster-Ressourcen weiterhin mit meiner eigenen Kubernetes-Backup-Lösung sichern und wiederherstellen?

Ja. Sie können eine Kubernetes-Backup-Lösung (wie Velero) für die Notfallwiederherstellung, Datenmigration und Datensicherung von Kubernetes-Clustern verwenden. Wenn Sie eine Kubernetes-Backup-Lösung verwenden, die über den API-Server auf die Cluster-Ressourcen zugreift, werden alle Daten, die die Anwendung abruft, entschlüsselt, bevor sie den Client erreichen. Auf diese Weise können Sie die Cluster-Ressourcen in einem anderen Kubernetes-Cluster wiederherstellen.

Sicherheitsüberlegungen für Amazon EKS Auto Mode

In diesem Thema werden die Sicherheitsarchitektur, die Steuerungen und die bewährten Verfahren für Amazon EKS Auto Mode beschrieben. Da Unternehmen containerisierte Anwendungen in großem Umfang einsetzen, wird es immer komplexer, eine starke Sicherheitslage aufrechtzuerhalten. EKS Auto Mode implementiert automatisierte Sicherheitskontrollen und lässt sich in AWS-Sicherheitsservices integrieren, um Sie beim Schutz Ihrer Cluster-Infrastruktur, Workloads und Daten zu unterstützen. Durch integrierte Sicherheitsfeatures wie die erzwungene Verwaltung des Knotenlebenszyklus und die automatisierte Bereitstellung von Patches unterstützt Sie EKS Auto Mode dabei, bewährte Sicherheitsverfahren einzuhalten und gleichzeitig den Betriebsaufwand zu reduzieren.

Bevor Sie mit diesem Thema fortfahren, stellen Sie sicher, dass Sie mit den grundlegenden Konzepten von EKS Auto Mode vertraut sind und die Voraussetzungen für die Aktivierung von EKS

Auto Mode in Ihren Clustern überprüft haben. Allgemeine Informationen zur Sicherheit von Amazon EKS finden Sie unter Sicherheit.

Amazon EKS Auto Mode basiert auf den bestehenden Sicherheitsgrundlagen von Amazon EKS und führt zusätzliche automatisierte Sicherheitskontrollen für von EC2 verwaltete Instances ein.

API-Sicherheit und Authentifizierung

Amazon EKS Auto Mode nutzt AWS-Plattform-Sicherheitsmechanismen, um Aufrufe an die Amazon-EKS-API zu sichern und zu authentifizieren.

- Der Zugriff auf die Kubernetes-API wird durch EKS-Zugriffseinträge gesichert, die in AWS-IAM-Identitäten integriert sind.
  - Weitere Informationen finden Sie unter the section called “Zugriffseinträge”.
- Kunden können durch die Konfiguration von EKS-Zugriffseinträgen eine detaillierte Zugriffskontrolle für den Kubernetes-API-Endpunkt implementieren.

Netzwerksicherheit

Amazon EKS Auto Mode unterstützt mehrere Ebenen der Netzwerksicherheit:

- VPC-Integration
  - Funktioniert innerhalb Ihrer Amazon Virtual Private Cloud (VPC)
  - Unterstützt benutzerdefinierte VPC-Konfigurationen und Subnetz-Layouts
  - Ermöglicht private Netzwerke zwischen Cluster-Komponenten
  - Weitere Informationen finden Sie unter Verwaltung von Sicherheitsaufgaben für Amazon Virtual Private Cloud
- Netzwerkrichtlinien
  - Nativer Support für Kubernetes-Netzwerkrichtlinien
  - Möglichkeit zur Definition detaillierter Regeln für den Netzwerk-Datenverkehr
  - Weitere Informationen finden Sie unter the section called “Kubernetes-Richtlinien”.

Sicherheit für verwaltete EC2-Instances

Amazon EKS Auto Mode betreibt verwaltete EC2-Instances mit den folgenden Sicherheitskontrollen:

EC2-Sicherheit

- Verwaltete EC2-Instances bieten die gleichen Sicherheitsfeature wie Amazon EC2.
- Weitere Informationen zu verwalteten EC2-Instances finden Sie unter Sicherheit in Amazon EC2.

Lebenszyklusmanagement von Instances

Von EKS Auto Mode betriebene verwaltete EC2-Instances haben eine maximale Lebensdauer von 21 Tagen. Amazon EKS Auto Mode beendet automatisch Instances, die diese Lebensdauer überschreiten. Diese Lebenszyklusbeschränkung trägt dazu bei, Konfigurationsabweichungen zu verhindern und die Sicherheitslage aufrechtzuerhalten.

Datenschutz

- Der Speicher der Amazon-EC2-Instance ist verschlüsselt. Es handelt sich hierbei um Speicher, der direkt mit der Instance verbunden ist. Weitere Informationen finden Sie unter Datenschutz in Amazon EC2.
- EKS Auto Mode verwaltet die bei der Erstellung an EC2-Instances angefügten Volumes, einschließlich Root- und Daten-Volumes. EKS Auto Mode verwaltet EBS-Volumes, die mit den persistenten Speicherfunktionen von Kubernetes erstellt wurden, nicht vollständig.

Patch-Management

- Amazon EKS Auto Mode wendet automatisch Patches auf verwaltete Instances an.
- Patches beinhalten:
  - Betriebssystemupdates
  - Sicherheits-Patches
  - Komponenten von Amazon EKS Auto Mode

> **Note**
>
> Kunden sind weiterhin für die Sicherung und Aktualisierung der in diesen Instances ausgeführten Workloads verantwortlich.

Zugriffskontrollen

- Der direkte Instance-Zugriff ist eingeschränkt:
  - SSH-Zugriff ist nicht verfügbar.
  - Der Zugriff auf AWS Systems Manager Session Manager (SSM) ist nicht verfügbar.
- Managementoperationen werden über die Amazon-EKS-API und die Kubernetes-API durchgeführt.

Automatisiertes Ressourcenmanagement

Amazon EKS Auto Mode verwaltet Amazon Elastic Block Store (Amazon EBS)-Volumes, die mit den persistenten Speicherfunktionen von Kubernetes erstellt wurden, nicht vollständig. EKS Auto Mode verwaltet auch keine Elastic Load Balancers (ELB). Amazon EKS Auto Mode automatisiert Routineaufgaben für diese Ressourcen.

Speichersicherheit

- AWS empfiehlt, die Verschlüsselung für EBS-Volumes zu aktivieren, die durch persistente Speicher-Features von Kubernetes bereitgestellt werden. Weitere Informationen finden Sie unter the section called “Erstellen StorageClass”.
- Verschlüsselung im Ruhezustand mit AWS KMS
- Sie können Ihr AWS-Konto so konfigurieren, dass es die Verschlüsselung neuer EBS-Volumes und Snapshot-Kopien erzwingt, die Sie erstellen. Weitere Informationen finden Sie unter Amazon-EBS-Verschlüsselung standardmäßig aktivieren im Amazon-EBS-Benutzerhandbuch.
- Weitere Informationen finden Sie unter Sicherheit in Amazon EBS.

Load-Balancer-Sicherheit

- Automatisierte Konfiguration von Elastic Load Balancers
- Verwaltung von SSL/TLS-Zertifikaten durch Integration des AWS-Zertifikatsmanagers
- Automatisierung von Sicherheitsgruppen für die Zugriffskontrolle des Load Balancers
- Weitere Informationen finden Sie unter Sicherheit in Elastic Load Balancing.

Bewährte Methoden für die Gewährleistung der Sicherheit

Der folgende Abschnitt beschreibt bewährte Sicherheitsmethoden für Amazon EKS Auto Mode.

- Überprüfen Sie regelmäßig die AWS-IAM-Richtlinien und EKS-Zugriffseinträge.
- Implementieren Sie Zugriffsmuster mit geringsten Berechtigungen für Workloads.
- Überwachen Sie die Cluster-Aktivität über AWS CloudTrail und Amazon CloudWatch. Weitere Informationen erhalten Sie unter the section called “ AWS CloudTrail” und the section called “Amazon CloudWatch”.
- Verwenden Sie AWS Security Hub zur Bewertung der Sicherheitslage.
- Implementieren Sie Pod-Sicherheitsstandards, die für Ihre Workloads geeignet sind.

Sicherheitsüberlegungen für EKS-Funktionen

In diesem Thema werden wichtige Sicherheitsaspekte für EKS-Funktionen behandelt, darunter die IAM-Rollenkonfiguration, Kubernetes-Berechtigungen und Architekturmuster für Bereitstellungen mit mehreren Clustern und kontenübergreifendes Ressourcenmanagement. AWS

EKS-Funktionen verwenden eine Kombination aus IAM-Rollen, EKS-Zugriffseinträgen und Kubernetes RBAC, um sicheren Zugriff auf AWS Dienste, Cluster-Kubernetes-Ressourcen und Integrationen mit Secrets Manager und anderen Diensten zu ermöglichen. AWS CodeConnections AWS AWS

Fähigkeit: IAM-Rolle

Wenn Sie eine Funktion erstellen, geben Sie eine IAM-Fähigkeitsrolle an, die EKS verwendet, um Aktionen in Ihrem Namen auszuführen. Diese Rolle muss:

- Sie müssen sich im selben AWS Konto befinden wie der Cluster und die Capability-Ressource
- Verfügen Sie über eine Vertrauensrichtlinie, die es dem `capabilities.eks.amazonaws.com` Dienstprinzipal ermöglicht, die Rolle zu übernehmen
- Verfügen Sie je nach Ihren Anforderungen über IAM-Berechtigungen, die für den Funktionstyp und den Anwendungsfall geeignet sind. Ausführliche Informationen zu den erforderlichen IAM-Berechtigungen finden Sie unterthe section called “ AWS CodeConnections”, und the section called “ AWS Secrets Manager” the section called “Konfigurieren von -Berechtigungen”

Es hat sich bewährt, den Umfang der für Ihren speziellen Anwendungsfall erforderlichen Rechte zu berücksichtigen und nur die Berechtigungen zu gewähren, die für Ihre Anforderungen erforderlich sind. Wenn Sie beispielsweise die EKS-Funktion für Kube Resource Orchestrator verwenden, sind

möglicherweise keine IAM-Berechtigungen erforderlich, während Sie bei Verwendung der EKS-Funktion für AWS Controller für Kubernetes Vollzugriff auf einen oder mehrere Dienste gewähren können. AWS

> **Important**
>
> In einigen Anwendungsfällen kann die Verwendung umfassender Administratorrechte gerechtfertigt sein. Folgen Sie jedoch dem Prinzip der geringsten Rechte, indem Sie nur die für Ihren speziellen Anwendungsfall erforderlichen IAM-Berechtigungen gewähren ARNs und den Zugriff auf bestimmte Ressourcen mithilfe von Bedingungsschlüsseln einschränken, anstatt Platzhalterberechtigungen zu verwenden.

Ausführliche Informationen zum Erstellen und Konfigurieren von Capability-IAM-Rollen finden Sie unter. the section called “Funktion: IAM-Rolle”

EKS-Zugriffseinträge

Wenn Sie eine Funktion mit einer IAM-Rolle erstellen, erstellt Amazon EKS automatisch einen Zugriffseintrag für diese Rolle in Ihrem Cluster. Dieser Zugriffseintrag gewährt der Fähigkeit grundlegende Funktionsberechtigungen für Kubernetes.

> **Note**
>
> Zugriffseinträge werden für den Cluster erstellt, in dem die Funktion erstellt wurde. Für Argo-CD-Bereitstellungen auf Remote-Clustern müssen Sie Zugriffseinträge auf diesen Clustern mit den entsprechenden Berechtigungen für die Argo-CD-Funktion erstellen, um Anwendungen bereitzustellen und zu verwalten.

Der Zugriffseintrag umfasst:

- Die IAM-Rolle ARN als Principal
- Funktionsspezifische Zugriffsrichtlinien, die grundlegende Kubernetes-Berechtigungen gewähren
- Angemessener Geltungsbereich (clusterweit oder namespace-bezogen) je nach Funktionstyp

> **Note**
>
> Für Argo CD werden namespace-bezogene Berechtigungen für den in der Capability-Konfiguration angegebenen Namespace gewährt (standardmäßig). `argocd`

Standardrichtlinien für den Zugriff nach Fähigkeit

Jeder Funktionstyp gewährt der Funktionsrolle die erforderlichen Berechtigungen, wobei unterschiedliche Standardzugriffsrichtlinien wie folgt festgelegt werden:

kro

- `arn:aws:eks::aws:cluster-access-policy/AmazonEKSKROPolicy`(im Clusterbereich)

  Erteilt Berechtigungen zum Überwachen, Verwalten ResourceGraphDefinitions und Erstellen von Instanzen benutzerdefinierter Ressourcen, die von definiert sind. RGDs

ZURÜCK

- `arn:aws:eks::aws:cluster-access-policy/AmazonEKSACKPolicy`(im Clusterbereich)

  Erteilt Berechtigungen zum Erstellen, Lesen, Aktualisieren und Löschen von benutzerdefinierten ACK-Ressourcen in allen Namespaces.

Argo-CD

- `arn:aws:eks::aws:cluster-access-policy/AmazonEKSArgoCDClusterPolicy`(im Clusterbereich)

  Gewährt Argo CD Berechtigungen auf Clusterebene, um Ressourcen zu ermitteln und Objekte im Clusterbereich zu verwalten.
- `arn:aws:eks::aws:cluster-access-policy/AmazonEKSArgoCDPolicy`(mit Namespace-Bereich)

  Gewährt Argo CD Berechtigungen auf Namespace-Ebene zur Bereitstellung und Verwaltung von Anwendungen. Gültig für den in der Capability-Konfiguration angegebenen Namespace (standardmäßig). `argocd`

Weitere Informationen finden Sie unterthe section called “Überprüfung von Zugriffsrichtlinien”.

Zusätzliche Kubernetes-Berechtigungen

Für einige Funktionen sind möglicherweise zusätzliche Kubernetes-Berechtigungen erforderlich, die über die standardmäßigen Zugriffsrichtlinien hinausgehen. Sie können diese Berechtigungen auf folgende Weise gewähren:

- Richtlinien für den Zugriff: Ordnen Sie dem Zugriffseintrag zusätzliche verwaltete Richtlinien zu
- Kubernetes RBAC: Erstellen `Role` oder `ClusterRole` Bindungen für den Kubernetes-Benutzer der Funktion

Geheime ACK-Leserberechtigungen

Einige ACK-Controller müssen Kubernetes-Geheimnisse lesen, um sensible Daten wie Datenbankpasswörter abzurufen. Die folgenden ACK-Controller benötigen geheimen Lesezugriff:

- `acm`, `acmpca`, `documentdb`, `memorydb`, `mq`, `rds`, `secretsmanager`

So gewähren Sie geheime Leseberechtigungen:

1. Ordnen Sie die `arn:aws:eks::aws:cluster-access-policy/AmazonEKSSecretReaderPolicy` Zugriffsrichtlinie dem Zugriffseintrag der Funktion zu
2. Richten Sie die Richtlinie auf bestimmte Namespaces ein, in denen ACK-Ressourcen auf geheime Informationen verweisen oder clusterweiten Zugriff gewähren

> **Important**
>
> Geheime Leseberechtigungen sind auf die Namespaces beschränkt, die Sie bei der Zuordnung der Zugriffsrichtlinie angeben. Auf diese Weise können Sie einschränken, auf welche Geheimnisse die Fähigkeit zugreifen kann.

kro, beliebige Ressourcenberechtigungen.

kro benötigt Berechtigungen zum Erstellen und Verwalten der in Ihrem definierten Ressourcen. ResourceGraphDefinitions Standardmäßig kann Kro nur sich selbst überwachen und verwalten RGDs .

So gewähren Sie Kro-Berechtigungen zum Erstellen von Ressourcen:

Option 1: Zugriffsrichtlinien

Ordnen Sie vordefinierte Zugriffsrichtlinien wie `AmazonEKSAdminPolicy` oder `AmazonEKSEditPolicy` dem Zugriffseintrag der Funktion zu.

Option 2: Kubernetes RBAC

Erstellen Sie eine`ClusterRoleBinding`, die dem Kubernetes-Benutzer der Funktion die erforderlichen Berechtigungen gewährt:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: kro-cluster-admin
subjects:
- kind: User
  name: arn:aws:sts::111122223333:assumed-role/my-kro-role/KRO
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: cluster-admin
  apiGroup: rbac.authorization.k8s.io
```

> **Note**
>
> Der Kubernetes-Benutzername für kro folgt dem Muster:
> `arn:aws:sts::ACCOUNT_ID:assumed-role/ROLE_NAME/KRO`
> Der Sitzungsname `/KRO` (in Großbuchstaben) wird automatisch von der EKS-Kro-Fähigkeit festgelegt.

Je nach Funktion sind IAM-Berechtigungen erforderlich

kro (Kube Resource Orchestrator)

Keine IAM-Berechtigungen erforderlich. Sie können eine Funktionsrolle ohne angehängte Richtlinien erstellen. kro benötigt nur Kubernetes-RBAC-Berechtigungen.

ACK (Controller für Kubernetes)AWS

Erfordert Berechtigungen zur Verwaltung der AWS Ressourcen, die ACK erstellt und verwaltet. Sie sollten die Berechtigungen auf bestimmte Dienste, Aktionen und Ressourcen auf der

Grundlage Ihrer Anforderungen einschränken. Ausführliche Informationen zur Konfiguration von ACK-Berechtigungen, einschließlich bewährter Produktionsmethoden mit IAM-Rollenselektoren, finden Sie unter. the section called “Konfigurieren von -Berechtigungen”

Argo CD

Standardmäßig sind keine IAM-Berechtigungen erforderlich. Optionale Berechtigungen können erforderlich sein für:

- AWS Secrets Manager: Wenn die Anmeldeinformationen für das Git-Repository in Secrets Manager gespeichert werden
- AWS CodeConnections: Bei Verwendung CodeConnections für die Git-Repository-Authentifizierung
- Amazon ECR: Wenn Sie Helm-Diagramme verwenden, die im OCI-Format in Amazon ECR gespeichert sind

Bewährte Methoden für die Gewährleistung der Sicherheit

IAM hat das geringste Privileg

Gewähren Sie Ihren Capability-Ressourcen nur die Berechtigungen, die für Ihren Anwendungsfall erforderlich sind. Dies bedeutet nicht, dass Sie Ihren Fähigkeiten bei Bedarf keine umfassenden Administratorberechtigungen gewähren können. In solchen Fällen sollten Sie den Zugriff auf diese Ressourcen angemessen regeln.

Fähigkeitsrollen:

- ACK: Beschränken Sie die IAM-Berechtigungen nach Möglichkeit auf bestimmte AWS Dienste und Ressourcen, die Ihre Teams benötigen, je nach Anwendungsfall und Anforderungen
- Argo CD: Beschränken Sie den Zugriff auf bestimmte Git-Repositorys und Kubernetes-Namespaces
- kro: Erfordert eine Capability-Rolle für die Vertrauensrichtlinie, es sind jedoch keine IAM-Berechtigungen erforderlich (verwendet nur Cluster-RBAC)

Beispiel: Geben Sie statt `"Resource": "*"` dessen Muster für bestimmte Ressourcen oder Ressourcengruppen an.

```
"Resource": [
```

```
  "arn:aws:s3:::my-app-*",
  "arn:aws:rds:us-west-2:111122223333:db:prod-*"
]
```

Verwenden Sie IAM-Bedingungsschlüssel, um den Zugriff weiter einzuschränken:

```
"Condition": {
  "StringEquals": {
    "aws:ResourceTag/Environment": "production"
  }
}
```

Weitere Informationen zur IAM-Konfiguration finden Sie im Abschnitt mit den Überlegungen zu den einzelnen Funktionen.

Namespace-Isolierung für Argo-CD-Geheimnisse

Die verwaltete Argo-CD-Funktion hat Zugriff auf alle Kubernetes-Secrets innerhalb des konfigurierten Namespace (Standard:). `argocd` Um ein optimales Sicherheitsniveau aufrechtzuerhalten, befolgen Sie die folgenden Methoden zur Namespace-Isolierung:

- Bewahren Sie nur Argo-CD-relevante Geheimnisse im Argo-CD-Namespace auf
- Vermeiden Sie es, Anwendungsgeheimnisse, die nichts miteinander zu tun haben, im selben Namespace wie Argo-CD zu speichern
- Verwenden Sie separate Namespaces für Anwendungsgeheimnisse, die für Argo-CD-Operationen nicht erforderlich sind

Diese Isolierung stellt sicher, dass der geheime Zugriff von Argo CD nur auf die Anmeldeinformationen beschränkt ist, die es für die Git-Repository-Authentifizierung und andere Argo-CD-spezifische Operationen benötigt.

Kubernetes RBAC

Steuern Sie, welche Benutzer und Dienstkonten Funktionsressourcen erstellen und verwalten können. Es hat sich bewährt, Capability-Ressourcen in dedizierten Namespaces mit entsprechenden RBAC-Richtlinien bereitzustellen.

Beispiel: Die RBAC-Rolle funktioniert mit ACK und ermöglicht die S3-Bucket-Ressourcenverwaltung im Namespace: `app-team`

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: ack-s3-manager
  namespace: app-team
rules:
- apiGroups: ["s3.services.k8s.aws"]
  resources: ["buckets"]
  verbs: ["get", "list", "create", "update", "delete"]
```

Audit-Protokollierung

CloudTrail: Alle EKS-Capability-API-Operationen (Erstellen, Aktualisieren, Löschen) werden protokolliert. AWS CloudTrail

Aktivieren Sie die CloudTrail Protokollierung, um Folgendes zu verfolgen:

- Wer hat Funktionen erstellt oder geändert
- Wann sich die Funktionskonfigurationen geändert haben
- Welche Funktionsrollen werden verwendet

Netzwerkzugriff und VPC-Endpunkte

Privater Argo CD-API-Zugriff

Sie können den Zugriff auf den Argo CD API-Server einschränken, indem Sie einen oder mehrere VPC-Endpunkte mit dem gehosteten Argo CD-Endpunkt verknüpfen. Dies ermöglicht private Konnektivität innerhalb Ihrer VPC, ohne das öffentliche Internet zu durchqueren. Der VPC-Endpunkt bietet Zugriff sowohl auf die Argo CD-Weboberfläche als auch auf die Argo-CD-API (einschließlich CLI-Zugriff).

> **Note**
>
> VPC-Endpunkte, die mit gehosteten Argo CD-API-Endpunkten verbunden sind (unter Verwendung von eks-Capabilities). *region*.amazonaws.com) unterstützen keine VPC-Endpunktrichtlinien.

Bereitstellung auf privaten Clustern

Mit der Argo-CD-Funktion können Anwendungen auf vollständig privaten EKS-Clustern bereitgestellt werden, was einen erheblichen betrieblichen Vorteil bietet, da kein VPC-Peering oder komplexe Netzwerkkonfigurationen erforderlich sind. Beachten Sie beim Entwerfen dieser Architektur jedoch, dass Argo CD die Konfiguration aus Git-Repositorys (die möglicherweise öffentlich sein können) abruft und auf Ihre privaten Cluster anwendet.

Stellen Sie sicher, dass Sie:

- Verwenden Sie private Git-Repositorys für sensible Workloads
- Implementieren Sie die richtigen Zugriffskontrollen und Authentifizierung für das Git-Repository
- Überprüfe und genehmige Änderungen mithilfe von Pull-Requests, bevor du sie zusammenführst
- Erwägen Sie, die Synchronisierungsfenster von Argo CD zu verwenden, um zu kontrollieren, wann Bereitstellungen erfolgen können
- Überwachen Sie die Audit-Logs von Argo CD auf unbefugte Konfigurationsänderungen

Compliance

Die EKS-Funktionen werden vollständig verwaltet und verfügen über die Compliance-Zertifizierungen von Amazon EKS.

Aktuelle Compliance-Informationen finden Sie unter AWS Services in Scope by Compliance Program.

Nächste Schritte

- the section called “Konfigurieren von -Berechtigungen”- Konfigurieren Sie IAM-Berechtigungen für ACK
- the section called “Konfigurieren von -Berechtigungen”- Konfigurieren Sie Kubernetes RBAC für Kro
- the section called “Konfigurieren von -Berechtigungen”- Konfigurieren Sie die Identity Center-Integration für Argo CD
- the section called “Problembehebung bei Funktionen”- Beheben Sie Sicherheits- und Berechtigungsprobleme

Identitäts- und Zugriffsverwaltung für Amazon EKS

AWS Identity and Access Management (IAM) ist ein AWS Dienst, der einem Administrator hilft, den Zugriff auf AWS Ressourcen sicher zu kontrollieren. IAM-Administratoren kontrollieren, wer authentifiziert (angemeldet) und autorisiert (mit Berechtigungen ausgestattet) werden kann, um Amazon-EKS-Ressourcen zu nutzen. IAM ist ein AWS Dienst, den Sie ohne zusätzliche Kosten nutzen können.

Zielgruppe

Wie Sie AWS Identity and Access Management (IAM) verwenden, hängt davon ab, welche Arbeit Sie in Amazon EKS ausführen.

Service-Benutzer – Wenn Sie den Amazon–EKS-Service zur Ausführung von Aufgaben verwenden, stellt Ihnen Ihr Administrator die Anmeldeinformationen und Berechtigungen bereit, die Sie benötigen. Je mehr Amazon-EKS-Funktionen Sie für Ihre Arbeit nutzen, desto mehr Berechtigungen benötigen Sie möglicherweise. Wenn Sie die Funktionsweise der Zugriffskontrolle nachvollziehen, wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anzufordern müssen. Unter the section called “Fehlerbehebung” finden Sie nützliche Informationen für den Fall, dass Sie keinen Zugriff auf eine Funktion in Amazon EKS haben.

Service-Administrator – Wenn Sie in Ihrem Unternehmen für die Amazon-EKS-Ressourcen zuständig sind, haben Sie wahrscheinlich vollständigen Zugriff auf Amazon EKS. Ihre Aufgabe besteht darin, zu bestimmen, auf welche Amazon-EKS-Features und -Ressourcen Ihre Servicebenutzer zugreifen sollen. Anschließend müssen Sie Anforderungen an Ihren IAM-Administrator senden, um die Berechtigungen der Servicebenutzer zu ändern. Lesen Sie die Informationen auf dieser Seite, um die Grundkonzepte von IAM nachzuvollziehen. Weitere Informationen dazu, wie Ihr Unternehmen IAM mit Amazon EKS verwenden kann, finden Sie unter the section called “Amazon EKS und IAM”.

IAM-Administrator – Wenn Sie ein IAM-Administrator sind, möchten Sie vielleicht Details darüber erfahren, wie Sie Richtlinien zur Verwaltung des Zugriffs auf Amazon EKS erstellen können. Beispiele für identitätsbasierte Amazon-EKS-Richtlinien, die Sie in IAM verwenden können, finden Sie unter the section called “Identitätsbasierte Richtlinien”.

Authentifizierung mit Identitäten

Authentifizierung ist die Art und Weise, wie Sie sich AWS mit Ihren Identitätsdaten anmelden. Sie müssen als Root-Benutzer des AWS Kontos, als IAM-Benutzer oder durch Übernahme einer IAM-Rolle authentifiziert (angemeldet AWS) sein.

Sie können sich AWS als föderierte Identität anmelden, indem Sie Anmeldeinformationen verwenden, die über eine Identitätsquelle bereitgestellt wurden. AWS IAM Identity Center (IAM Identity Center) -Benutzer, die Single Sign-On-Authentifizierung Ihres Unternehmens und Ihre Google- oder Facebook-Anmeldeinformationen sind Beispiele für föderierte Identitäten. Wenn Sie sich als Verbundidentität anmelden, hat der Administrator vorher mithilfe von IAM-Rollen einen Identitätsverbund eingerichtet. Wenn Sie über den Verbund darauf zugreifen AWS , übernehmen Sie indirekt eine Rolle.

Je nachdem, welcher Benutzertyp Sie sind, können Sie sich beim AWS-Managementkonsole oder beim AWS Zugangsportal anmelden. Weitere Informationen zur Anmeldung finden Sie unter So melden Sie sich bei Ihrem AWS Konto an im AWS Anmelde-Benutzerhandbuch. AWS

Wenn Sie AWS programmgesteuert darauf zugreifen, AWS stellt es ein Software Development Kit (SDK) und eine Befehlszeilenschnittstelle (CLI) bereit, mit denen Sie Ihre Anfragen mithilfe Ihrer Anmeldeinformationen kryptografisch signieren können. Wenn Sie keine AWS Tools verwenden, müssen Sie Anfragen selbst signieren. Weitere Informationen zur Verwendung der empfohlenen Methode, um Anfragen selbst zu signieren, finden Sie im IAM-Benutzerhandbuch unter AWS API-Anfragen signieren.

Unabhängig von der verwendeten Authentifizierungsmethode müssen Sie möglicherweise zusätzliche Sicherheitsinformationen bereitstellen. AWS Empfiehlt beispielsweise, die Multi-Faktor-Authentifizierung (MFA) zu verwenden, um die Sicherheit Ihres Kontos zu erhöhen. Weitere Informationen finden Sie unter Multi-Factor Authentication im AWS IAM Identity Center User Guide und Using Multi-Factor Authentication (MFA) AWS im IAM-Benutzerhandbuch.

AWS Konto (Root-Benutzer)

Wenn Sie ein AWS Konto erstellen, beginnen Sie mit einer einzigen Anmeldeidentität, die vollständigen Zugriff auf alle AWS Dienste und Ressourcen im Konto hat. Diese Identität wird als Root-Benutzer des AWS Kontos bezeichnet. Der Zugriff erfolgt, indem Sie sich mit der E-Mail-Adresse und dem Passwort anmelden, mit denen Sie das Konto erstellt haben. Wir raten ausdrücklich davon ab, den Root-Benutzer für alltägliche Aufgaben zu verwenden. Schützen Sie Ihre Root-Benutzer-Anmeldeinformationen. Verwenden Sie diese nur, um die Aufgaben auszuführen, die nur der Root-Benutzer ausführen kann. Eine vollständige Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie unter Aufgaben, die Root-Benutzer-Anmeldeinformationen erfordern im IAM-Benutzerhandbuch.

IAM-Benutzer und -Gruppen

Ein IAM-Benutzer ist eine Identität innerhalb Ihres AWS Kontos, die über spezifische Berechtigungen für eine einzelne Person oder Anwendung verfügt. Wenn möglich, empfehlen wir, temporäre Anmeldeinformationen zu verwenden, anstatt IAM-Benutzer zu erstellen, die langfristige Anmeldeinformationen wie Passwörter und Zugriffsschlüssel haben. Bei speziellen Anwendungsfällen, die langfristige Anmeldeinformationen mit IAM-Benutzern erfordern, empfehlen wir jedoch, die Zugriffsschlüssel zu rotieren. Weitere Informationen finden Sie unter Regelmäßiges Rotieren von Zugriffsschlüsseln für Anwendungsfälle, die langfristige Anmeldeinformationen erfordern im IAM-Benutzerhandbuch.

Eine IAM-Gruppe ist eine Identität, die eine Sammlung von IAM-Benutzern angibt. Sie können sich nicht als Gruppe anmelden. Mithilfe von Gruppen können Sie Berechtigungen für mehrere Benutzer gleichzeitig angeben. Gruppen erleichtern die Verwaltung von Berechtigungen für große Benutzergruppen. Sie könnten beispielsweise einer Gruppe einen Namen geben IAMAdminsund dieser Gruppe Berechtigungen zur Verwaltung von IAM-Ressourcen erteilen.

Benutzer sind nicht dasselbe wie Rollen. Ein Benutzer ist einer einzigen Person oder Anwendung eindeutig zugeordnet. Eine Rolle kann von allen Personen angenommen werden, die sie benötigen. Benutzer besitzen dauerhafte Anmeldeinformationen. Rollen stellen temporäre Anmeldeinformationen bereit. Weitere Informationen finden Sie unter Erstellen eines IAM-Benutzers (anstatt einer Rolle) im IAM-Benutzerhandbuch.

IAM-Rollen

Eine IAM-Rolle ist eine Identität innerhalb Ihres AWS Kontos, die über bestimmte Berechtigungen verfügt. Sie ist einem IAM-Benutzer vergleichbar, jedoch nicht mit einer bestimmten Person verknüpft. Sie können vorübergehend eine IAM-Rolle in der übernehmen, AWS-Managementkonsole indem Sie die Rollen wechseln. Sie können eine Rolle übernehmen, indem Sie eine AWS CLI- oder AWS API-Operation aufrufen oder eine benutzerdefinierte URL verwenden. Weitere Informationen zu Methoden für die Verwendung von Rollen finden Sie unter Verwenden von IAM-Rollen im IAM-Benutzerhandbuch.

IAM-Rollen mit temporären Anmeldeinformationen sind in folgenden Situationen hilfreich:

- Verbundbenutzerzugriff – Um einer Verbundidentität Berechtigungen zuzuweisen, erstellen Sie eine Rolle und definieren Berechtigungen für die Rolle. Wird eine Verbundidentität authentifiziert, so wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter Erstellen von Rollen für externe

Identitätsanbieter im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM. Informationen zu Berechtigungssätzen finden Sie unter Berechtigungssätze im AWS IAM Identity Center-Benutzerhandbuch.

- Temporäre IAM-Benutzerberechtigungen – Ein IAM-Benutzer oder eine -Rolle kann eine IAM-Rolle übernehmen, um vorübergehend andere Berechtigungen für eine bestimmte Aufgabe zu erhalten.
- Kontoübergreifender Zugriff – Sie können eine IAM-Rolle verwenden, um einem vertrauenswürdigen Prinzipal in einem anderen Konto den Zugriff auf Ressourcen in Ihrem Konto zu ermöglichen. Rollen stellen die primäre Möglichkeit dar, um kontoübergreifendem Zugriff zu gewähren. Bei einigen AWS Diensten können Sie jedoch eine Richtlinie direkt an eine Ressource anhängen (anstatt eine Rolle als Proxy zu verwenden). Informationen zu den Unterschieden zwischen Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.
- Serviceübergreifender Zugriff — Einige AWS Dienste verwenden Funktionen in anderen AWS Diensten. Wenn Sie beispielsweise in einem Service einen Anruf tätigen, ist es üblich, dass dieser Service Anwendungen in Amazon ausführt EC2 oder Objekte in Amazon S3 speichert. Ein Service kann dies mithilfe der Berechtigungen des aufrufenden Prinzipals, einer Servicerolle oder einer serviceverknüpften Rolle tun.
  - Forward Access Sessions (FAS) — Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen AWS Dienst aufruft, in Kombination mit dem anfordernden AWS Dienst, um Anfragen an nachgelagerte Dienste zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS Diensten oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anforderungen finden Sie unter Zugriffssitzungen weiterleiten.
  - Servicerolle – Eine Servicerolle ist eine IAM-Rolle, die ein Service übernimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Erstellen einer Rolle zum Delegieren von Berechtigungen für einen AWS Dienst.
  - Dienstverknüpfte Rolle — Eine dienstverknüpfte Rolle ist eine Art von Servicerolle, die mit einem Dienst verknüpft ist. AWS Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Servicebezogene Rollen werden in Ihrem AWS Konto angezeigt und

gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.

- Auf Amazon ausgeführte Anwendungen EC2 — Sie können eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer EC2 Instance ausgeführt werden und AWS CLI- oder AWS API-Anfragen stellen. Dies ist dem Speichern von Zugriffsschlüsseln innerhalb der EC2 Instance vorzuziehen. Um einer EC2 Instanz eine AWS Rolle zuzuweisen und sie allen ihren Anwendungen zur Verfügung zu stellen, erstellen Sie ein Instanzprofil, das an die Instanz angehängt ist. Ein Instanzprofil enthält die Rolle und ermöglicht Programmen, die auf der EC2 Instanz ausgeführt werden, temporäre Anmeldeinformationen abzurufen. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf EC2 Amazon-Instances ausgeführt werden.

Informationen dazu, wann Sie IAM-Rollen oder IAM-Benutzer verwenden sollten, finden Sie unter Erstellen einer IAM-Rolle (anstatt eines Benutzers) im IAM-Benutzerhandbuch.

Verwalten des Zugriffs mit Richtlinien

Sie kontrollieren den Zugriff, AWS indem Sie Richtlinien erstellen und diese an AWS Identitäten oder Ressourcen anhängen. Eine Richtlinie ist ein Objekt, AWS das, wenn es einer Identität oder Ressource zugeordnet ist, deren Berechtigungen definiert. AWS wertet diese Richtlinien aus, wenn ein Prinzipal (Benutzer, Root-Benutzer oder Rollensitzung) eine Anfrage stellt. Die Berechtigungen in den Richtlinien legen fest, ob eine Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Weitere Informationen zu Struktur und Inhalten von JSON-Richtliniendokumenten finden Sie unter Übersicht über JSON-Richtlinien im IAM-Benutzerhandbuch.

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

IAM-Richtlinien definieren Berechtigungen für eine Aktion unabhängig von der Methode, die Sie zur Ausführung der Aktion verwenden. Angenommen, es gibt eine Richtlinie, die Berechtigungen für die

`iam:GetRole`-Aktion erteilt. Ein Benutzer mit dieser Richtlinie kann Rolleninformationen von der AWS-Managementkonsole, der AWS CLI oder der AWS API abrufen.

Identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.

Identitätsbasierte Richtlinien können weiter als Inline-Richtlinien oder verwaltete Richtlinien kategorisiert werden. Eingebundene Richtlinien sind direkt in einen einzelnen Benutzer, eine einzelne Gruppe oder eine einzelne Rolle eingebettet. Verwaltete Richtlinien sind eigenständige Richtlinien, die Sie mehreren Benutzern, Gruppen und Rollen in Ihrem AWS Konto zuordnen können. Zu den verwalteten Richtlinien gehören AWS verwaltete Richtlinien und vom Kunden verwaltete Richtlinien. Informationen dazu, wie Sie zwischen einer verwalteten Richtlinie und einer eingebundenen Richtlinie wählen, finden Sie unter Auswahl zwischen verwalteten und eingebundenen Richtlinien im IAM-Benutzerhandbuch.

Ressourcenbasierte Richtlinien

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben. Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder AWS Dienste gehören.

Ressourcenbasierte Richtlinien sind Richtlinien innerhalb dieses Diensts. Sie können AWS verwaltete Richtlinien von IAM nicht in einer ressourcenbasierten Richtlinie verwenden.

Zugriffskontrolllisten () ACLs

Zugriffskontrolllisten (ACLs) steuern, welche Principals (Kontomitglieder, Benutzer oder Rollen) über Zugriffsberechtigungen für eine Ressource verfügen. ACLs ähneln ressourcenbasierten Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Amazon S3, AWS WAF und Amazon VPC sind Beispiele für Dienste, die Unterstützung bieten. ACLs Weitere Informationen finden Sie unter Übersicht über ACLs die Zugriffskontrollliste (ACL) im Amazon Simple Storage Service Developer Guide.

Weitere Richtlinientypen

AWS unterstützt zusätzliche, weniger verbreitete Richtlinientypen. Mit diesen Richtlinientypen können Sie die maximalen Berechtigungen festlegen, die Ihnen durch die gängigeren Richtlinientypen gewährt werden.

- Berechtigungsgrenzen – Eine Berechtigungsgrenze ist ein erweitertes Feature, mit der Sie die maximalen Berechtigungen festlegen können, die eine identitätsbasierte Richtlinie einer IAM-Entität (IAM-Benutzer oder -Rolle) erteilen kann. Sie können eine Berechtigungsgrenze für eine Entität festlegen. Die resultierenden Berechtigungen sind eine Schnittmenge der identitätsbasierten Richtlinien einer Entität und ihrer Berechtigungsgrenzen. Ressourcenbasierte Richtlinien, die den Benutzer oder die Rolle im Feld `Principal` angeben, werden nicht durch Berechtigungsgrenzen eingeschränkt. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen über Berechtigungsgrenzen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten im IAM-Benutzerhandbuch.
- Dienststeuerungsrichtlinien (SCPs) — SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OU) in AWS Organizations festlegen. AWS Organizations ist ein Dienst zur Gruppierung und zentralen Verwaltung mehrerer AWS Konten, die Ihrem Unternehmen gehören. Wenn Sie alle Funktionen in einer Organisation aktivieren, können Sie Richtlinien zur Servicesteuerung (SCPs) auf einige oder alle Ihre Konten anwenden. Das SCP schränkt die Berechtigungen für Entitäten in Mitgliedskonten ein, einschließlich der einzelnen AWS Root-Benutzer. Weitere Informationen zu Organizations und SCPs finden Sie unter Richtlinien zur Servicesteuerung im AWS Organizations User Guide.
- Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen verbundenen Benutzer programmgesteuert erstellen. Die resultierenden Sitzungsberechtigungen sind die Schnittmenge der identitätsbasierten Richtlinien des Benutzers oder der Rolle und der Sitzungsrichtlinien. Berechtigungen können auch aus einer ressourcenbasierten Richtlinie stammen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen finden Sie unter Sitzungsrichtlinien im IAM-Benutzerhandbuch.

Mehrere Richtlinientypen

Wenn für eine Anfrage mehrere Arten von Richtlinien gelten, sind die sich daraus ergebenden Berechtigungen schwieriger zu verstehen. Informationen darüber, wie AWS bestimmt wird, ob eine Anfrage zulässig ist, wenn mehrere Richtlinientypen betroffen sind, finden Sie unter Bewertungslogik für Richtlinien im IAM-Benutzerhandbuch.

Funktionsweise von Amazon EKS mit IAM

Bevor Sie IAM verwenden, um den Zugriff auf Amazon EKS zu verwalten, sollten Sie verstehen, welche IAM-Funktionen für die Verwendung mit Amazon EKS verfügbar sind. Einen Überblick darüber, wie Amazon EKS und andere AWS-Services mit IAM arbeiten, finden Sie unter AWS-Services, die mit IAM arbeiten im IAM-Benutzerhandbuch.

Themen

- Identitätsbasierte Amazon-EKS-Richtlinien
- Ressourcenbasierte Amazon-EKS-Richtlinien
- Autorisierung auf der Basis von Amazon-EKS-Tags
- Amazon EKS IAM-Rollen

Identitätsbasierte Amazon-EKS-Richtlinien

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen zugelassen oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter denen Aktionen zugelassen oder abgelehnt werden. Amazon EKS unterstützt bestimmte Aktionen, Ressourcen und Zustandsschlüssel. Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der IAM-Referenz für JSON-Richtlinienelemente im IAM-Benutzerhandbuch.

Aktionen

Administratoren können mit AWS-JSON-Richtlinien festlegen, welche Personen zum Zugriff auf welche Ressourcen berechtigt sind. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das Element `Action` einer JSON-Richtlinie beschreibt die Aktionen, mit denen Sie den Zugriff in einer Richtlinie zulassen oder verweigern können. Richtlinienaktionen haben normalerweise

denselben Namen wie die zugehörige AWS-API-Operation. Es gibt einige Ausnahmen, z. B. Aktionen, die nur mit Genehmigung durchgeführt werden können und für die es keine entsprechende API-Operation gibt. Es gibt auch einige Operationen, die mehrere Aktionen in einer Richtlinie erfordern. Diese zusätzlichen Aktionen werden als abhängige Aktionen bezeichnet.

Schließen Sie Aktionen in eine Richtlinie ein, um Berechtigungen zur Durchführung der zugeordneten Operation zu erteilen.

Richtlinienaktionen in Amazon EKS verwenden das folgende Präfix vor der Aktion: `eks:`. Um jemandem beispielsweise die Berechtigung zum Abrufen von Informationen zu einem Amazon-EKS-Cluster zu erteilen, fügen Sie die Aktion `DescribeCluster` in seine Richtlinie ein. Richtlinienanweisungen müssen entweder ein `Action`- oder ein `NotAction`-Element enthalten.

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie wie folgt durch Kommata:

```
"Action": ["eks:action1", "eks:action2"]
```

Sie können auch Platzhalter verwenden, um mehrere Aktionen anzugeben. Beispielsweise können Sie alle Aktionen festlegen, die mit dem Wort `Describe` beginnen, einschließlich der folgenden Aktion:

```
"Action": "eks:Describe*"
```

Eine Liste der Amazon-EKS-Aktionen finden Sie unter Von Amazon Elastic Kubernetes Service definierte Aktionen in der Serviceautorisierungsreferenz.

Ressourcen

Administratoren können mit AWS-JSON-Richtlinien festlegen, welche Personen zum Zugriff auf welche Ressourcen berechtigt sind. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das JSON-Richtlinienelement `Resource` gibt die Objekte an, auf welche die Aktion angewendet wird. Anweisungen müssen entweder ein – `Resource`oder ein `NotResource`-Element enthalten. Als bewährte Methode geben Sie eine Ressource mit dem zugehörigen Amazon-Ressourcennamen (ARN) an. Sie können dies für Aktionen tun, die einen bestimmten Ressourcentyp unterstützen, der als Berechtigungen auf Ressourcenebene bezeichnet wird.

Verwenden Sie für Aktionen, die keine Berechtigungen auf Ressourcenebene unterstützen, z. B. Auflistungsoperationen, einen Platzhalter (*), um anzugeben, dass die Anweisung für alle Ressourcen gilt.

```
"Resource": "*"
```

Die Amazon-EKS-Cluster-Ressource verfügt über den folgenden ARN:

```
arn:aws:eks:region-code:account-id:cluster/cluster-name
```

Weitere Informationen zum Format von ARNs finden Sie unter Amazon-Ressourcennamen (ARNs) und AWS-Service-Namespaces.

Um beispielsweise den Cluster mit dem Namen *my-cluster* in Ihrer Anweisung anzugeben, verwenden Sie den folgenden ARN:

```
"Resource": "arn:aws:eks:region-code:111122223333:cluster/my-cluster"
```

Um alle Cluster anzugeben, die zu einem bestimmten Konto und einer bestimmten AWS-Region gehören, verwenden Sie den Platzhalter (*):

```
"Resource": "arn:aws:eks:region-code:111122223333:cluster/*"
```

Einige Amazon-EKS-Aktionen, z. B. das Erstellen von Ressourcen, können für bestimmte Ressourcen nicht ausgeführt werden. In diesen Fällen müssen Sie den Platzhalter (*) verwenden.

```
"Resource": "*"
```

Eine Liste der Amazon-EKS-Ressourcentypen und ihrer ARNs finden Sie unter Von Amazon Elastic Kubernetes Service definierte Ressourcen in der Service-Autorisierungsreferenz. Um zu erfahren, mit welchen Aktionen Sie den ARN jeder Ressource angeben können, lesen Sie von Amazon Elastic Kubernetes Service definierte Aktionen.

Bedingungsschlüssel

Amazon EKS definiert seinen eigenen Satz von Konditionsschlüsseln und unterstützt auch die Verwendung einiger globaler Konditionsschlüssel. Eine Liste aller globalen AWS-

Bedingungsschlüssel finden Sie unter Globale AWS-Bedingungskontextschlüssel im IAM-Benutzerhandbuch.

Sie können Bedingungsschlüssel festlegen, wenn Sie Ihrem Cluster einen OpenID Connect-Anbieter zuordnen. Weitere Informationen finden Sie unter the section called “Beispiel für eine IAM-Richtlinie”.

Alle Amazon EC2-Aktionen unterstützen die Bedingungsschlüssel `aws:RequestedRegion` und `ec2:Region`. Weitere Informationen finden Sie unter Beispiel: Einschränken des Zugriffs auf eine bestimmte AWS-Region.

Eine Liste der Amazon EKS-Bedingungsschlüssel finden Sie unter Bedingungsschlüssel für Amazon Elastic Kubernetes Service in der Service-Autorisierungsreferenz. Um zu erfahren, mit welchen Aktionen und Ressourcen Sie einen Bedingungsschlüssel verwenden können, lesen Sie von Amazon Elastic Kubernetes Service definierte Aktionen.

Beispiele

Beispiele für identitätsbasierte Amazon-EKS-Richtlinien finden Sie unter the section called “Identitätsbasierte Richtlinien”.

Wenn Sie einen Amazon-EKS-Cluster erstellen, werden dem IAM-Prinzipal, der den Cluster erstellt, automatisch `system:masters`-Berechtigungen in der rollenbasierten Konfiguration für Zugriffskontrolle (RBAC) des Clusters in der Amazon-EKS-Steuerebene gewährt. Dieser Prinzipal wird in keiner sichtbaren Konfiguration angezeigt. Achten Sie daher darauf, welcher Prinzipal den Cluster ursprünglich erstellt hat. Um zusätzlichen IAM-Prinzipalen die Möglichkeit zu geben, mit Ihrem Cluster zu interagieren, bearbeiten Sie den `aws-auth ConfigMap` in Kubernetes und erstellen Sie ein Kubernetes-`rolebinding` oder -`clusterrolebinding` mit dem Namen eines `group`, das Sie im `aws-auth ConfigMap` angeben.

Weitere Informationen zum Arbeiten mit der ConfigMap finden Sie unter the section called “Kubernetes-API-Zugriff”.

Ressourcenbasierte Amazon-EKS-Richtlinien

Amazon EKS unterstützt keine ressourcenbasierten Richtlinien.

Autorisierung auf der Basis von Amazon-EKS-Tags

Sie können Tags an Amazon-EKS-Ressourcen anhängen oder Tags in einer Anforderung an Amazon EKS übergeben. Um den Zugriff auf der Grundlage von Tags zu steuern, geben

Sie im Bedingungselement einer Richtlinie Tag-Informationen an, indem Sie die Schlüssel `aws:ResourceTag/key-name` , `aws:RequestTag/key-name` , oder `aws:TagKeys`Bedingung verwenden. Weitere Informationen über die Markierung von Amazon-EKS-Ressourcen finden Sie unter the section called “Markieren Ihrer -Ressourcen”. Weitere Informationen darüber, mit welchen Aktionen Sie Tags in Bedingungsschlüsseln verwenden können, finden Sie unter Von Amazon EKS definierte Aktionen in der Service-Authorization-Referenz.

Amazon EKS IAM-Rollen

Eine IAM-Rolle ist eine Entität in Ihrem AWS-Konto mit spezifischen Berechtigungen.

Verwenden temporärer Anmeldeinformationen mit Amazon EKS

Sie können temporäre Anmeldeinformationen verwenden, um sich über einen Verbund anzumelden, eine IAM-Rolle anzunehmen oder eine kontenübergreifende Rolle anzunehmen. Temporäre Sicherheitsanmeldeinformationen erhalten Sie durch Aufrufen von AWS-STS-API-Vorgängen wie AssumeRole oder GetFederationToken.

Amazon EKS unterstützt die Verwendung temporärer Anmeldeinformationen.

Serviceverknüpfte Rollen

Serviceverknüpfte Rollen erlauben AWS-Services den Zugriff auf Ressourcen in anderen Services, um eine Aktion in Ihrem Auftrag auszuführen. Serviceverknüpfte Rollen werden in Ihrem IAM-Konto angezeigt und gehören zum Service. Ein Administrator kann die Berechtigungen für serviceverknüpfte Rollen anzeigen, jedoch nicht bearbeiten.

Amazon EKS unterstützt Service-verknüpfte Rollen. Details zum Erstellen oder Verwalten von serviceverknüpften Amazon-EKS-Rollen finden Sie unter the section called “Service-verknüpfte Rollen”.

Servicerollen

Dieses Feature ermöglicht einem Service das Annehmen einer Servicerolle in Ihrem Namen. Diese Rolle gewährt dem Service Zugriff auf Ressourcen in anderen Diensten, um eine Aktion in Ihrem Namen auszuführen. Servicerollen werden in Ihrem IAM-Konto angezeigt und gehören zum Konto. Dies bedeutet, dass ein IAM-Administrator die Berechtigungen für diese Rolle ändern kann. Dies kann jedoch die Funktionalität des Services beeinträchtigen.

Amazon EKS unterstützt Servicerollen. Weitere Informationen erhalten Sie unter the section called “Cluster-IAM-Rolle” und the section called “Knoten-IAM-Rolle”.

Auswählen einer IAM-Rolle in Amazon EKS

Wenn Sie eine Clusterressource in Amazon EKS erstellen, müssen Sie eine Rolle auswählen, damit Amazon EKS in Ihrem Namen auf mehrere andere AWS-Ressourcen zugreifen kann. Wenn Sie zuvor eine Servicerolle erstellt haben, stellt Ihnen Amazon EKS eine Liste der Rollen bereit, aus denen Sie auswählen können. Es ist wichtig, eine Rolle auszuwählen, an welche die von Amazon EKS verwalteten Richtlinien angefügt sind. Weitere Informationen erhalten Sie unter the section called “Überprüfen, ob eine Clusterrolle vorhanden ist” und the section called “Nach einer vorhandenen Knotenrolle suchen”.

Beispiele für identitätsbasierte Amazon-EKS-Richtlinien

Standardmäßig haben IAM-Benutzer und -Rollen nicht die Berechtigung, Amazon-EKS-Ressourcen zu erstellen oder zu ändern. Sie können auch keine Aufgaben mit der AWS-Managementkonsole AWS CLI oder der AWS API ausführen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern und Rollen die Berechtigung zum Ausführen bestimmter API-Operationen für die angegebenen Ressourcen gewähren, die diese benötigen. Der Administrator muss diese Richtlinien anschließend den IAM-Benutzern oder -Gruppen anfügen, die diese Berechtigungen benötigen.

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter Erstellen von Richtlinien auf der JSON-Registerkarte im IAM-Benutzerhandbuch.

Wenn Sie einen Amazon-EKS-Cluster erstellen, werden dem IAM-Prinzipal, der den Cluster erstellt, automatisch `system:masters`-Berechtigungen in der rollenbasierten Konfiguration für Zugriffskontrolle (RBAC) des Clusters in der Amazon-EKS-Steuerebene gewährt. Dieser Prinzipal wird in keiner sichtbaren Konfiguration angezeigt. Achten Sie daher darauf, welcher Prinzipal den Cluster ursprünglich erstellt hat. Um zusätzlichen IAM-Prinzipalen die Möglichkeit zu geben, mit Ihrem Cluster zu interagieren, bearbeiten Sie den `aws-auth ConfigMap` in Kubernetes und erstellen Sie ein Kubernetes-`rolebinding` oder -`clusterrolebinding` mit dem Namen eines `group`, das Sie im `aws-auth ConfigMap` angeben.

Weitere Informationen zur Arbeit mit dem ConfigMap finden Sie unterthe section called “Kubernetes-API-Zugriff”.

Themen

- Best Practices für Richtlinien
- Verwenden der Amazon-EKS-Konsole

- Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für IAM-Benutzer
- Erstellen Sie einen Kubernetes-Cluster in der Cloud AWS
- Kubernetes-Cluster auf einem Outpost erstellen
- Aktualisieren eines Kubernetes-Clusters
- Auflisten oder Beschreiben aller Cluster

Best Practices für Richtlinien

Identitätsbasierte Richtlinien können festlegen, ob jemand Amazon-EKS-Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder daraus löschen kann. Diese Aktionen können mit Kosten für Ihr AWS Konto verbunden sein. Beachten Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Richtlinien und Empfehlungen:

- Erste Schritte mit AWS verwalteten Richtlinien und Umstellung auf Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um damit zu beginnen, Ihren Benutzern und Workloads Berechtigungen zu gewähren. Sie sind in Ihrem Konto verfügbar. AWS Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien oder Von AWS verwaltete Richtlinien für Auftragsfunktionen im IAM-Benutzerhandbuch.
- Anwendung von Berechtigungen mit den geringsten Rechten – Wenn Sie mit IAM-Richtlinien Berechtigungen festlegen, gewähren Sie nur die Berechtigungen, die für die Durchführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung von IAM zum Anwenden von Berechtigungen finden Sie unter Richtlinien und Berechtigungen in IAM im IAM-Benutzerhandbuch.
- Verwenden von Bedingungen in IAM-Richtlinien zur weiteren Einschränkung des Zugriffs – Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und Ressourcen zu beschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Serviceaktionen zu gewähren, wenn diese über einen bestimmten AWS Dienst verwendet werden, z. AWS CloudFormation B. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingung im IAM-Benutzerhandbuch.

- Verwenden von IAM Access Analyzer zur Validierung Ihrer IAM-Richtlinien, um sichere und funktionale Berechtigungen zu gewährleisten – IAM Access Analyzer validiert neue und vorhandene Richtlinien, damit die Richtlinien der IAM-Richtliniensprache (JSON) und den bewährten IAM-Methoden entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie unter Richtlinienvalidierung zum IAM Access Analyzer im IAM-Benutzerhandbuch.
- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem AWS Konto erfordert, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren Richtlinien MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter Konfigurieren eines MFA-geschützten API-Zugriffs im IAM-Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden in IAM finden Sie unter Bewährte Methoden für die Sicherheit in IAM im IAM-Benutzerhandbuch.

Verwenden der Amazon-EKS-Konsole

Um auf die Amazon EKS-Konsole zugreifen zu können, muss ein IAM-Prinzipal über einen Mindestsatz von Berechtigungen verfügen. Diese Berechtigungen ermöglichen es dem Principal, Details zu den Amazon EKS-Ressourcen in Ihrem AWS Konto aufzulisten und einzusehen. Wenn Sie eine identitätsbasierte Richtlinie erstellen, die restriktiver als die mindestens erforderlichen Berechtigungen ist, funktioniert die Konsole für Prinzipals, denen diese Richtlinie zugewiesen ist, nicht wie vorgesehen.

Um sicherzustellen, dass Ihre IAM-Prinzipals die Amazon EKS-Konsole weiterhin verwenden können, erstellen Sie eine Richtlinie mit Ihrem eigenen eindeutigen Namen, z. B. `AmazonEKSAdminPolicy`. Hängen Sie die Richtlinie an die Prinzipale an. Informationen finden Sie im Abschnitt Hinzufügen und Entfernen von IAM-Identitätsberechtigungen im IAM-Benutzerhandbuch.

> **Important**
>
> Mit der folgenden Beispielrichtlinie kann ein Prinzipal Informationen auf der Registerkarte Konfiguration in der Konsole anzeigen. Um Informationen auf den Registerkarten „Übersicht“ und „Ressourcen“ in anzeigen zu können AWS-Managementkonsole, benötigt der Principal außerdem Kubernetes-Berechtigungen. Weitere Informationen finden Sie unter the section called “Erforderliche Berechtigungen”.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "eks.amazonaws.com"
                }
            }
        }
    ]
}
```

Sie müssen keine Mindestkonsolenberechtigungen für Principals gewähren, die nur die AWS CLI oder die AWS API aufrufen. Stattdessen sollten Sie nur Zugriff auf die Aktionen zulassen, die der API-Operation entsprechen, die Sie ausführen möchten.

Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für IAM-Benutzer

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie umfasst Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der AWS CLI oder AWS API.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
```

```
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Erstellen Sie einen Kubernetes-Cluster in der Cloud AWS

Diese Beispielrichtlinie umfasst die Mindestberechtigungen, die erforderlich sind, um einen Amazon EKS-Cluster mit *`my-cluster`* dem Namen *`us-west-2`* AWS Region zu erstellen. Sie können die AWS Region durch die AWS Region ersetzen, in der Sie einen Cluster erstellen möchten. Wenn Sie eine Warnung sehen, die besagt, dass die Aktionen in Ihrer Richtlinie keine Berechtigungen auf Ressourcenebene unterstützen und Sie sich für die entscheiden **`All resources`** müssen AWS-Managementkonsole, können Sie diese problemlos ignorieren. Wenn Ihr Konto bereits über die *`AWSServiceRoleForAmazonEKS`*-Rolle verfügt, können Sie die `iam:CreateServiceLinkedRole`-Aktion aus der Richtlinie entfernen. Wenn Sie jemals einen Amazon-EKS-Cluster in Ihrem Konto erstellt haben, ist diese Rolle bereits vorhanden, sofern Sie sie nicht gelöscht haben.

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "eks:CreateCluster",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-cluster"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::111122223333:role/aws-service-role/
eks.amazonaws.com/AWSServiceRoleForAmazonEKS",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "iam:AWSServiceName": "eks"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/cluster-role-name"
        }
    ]
}
```

Kubernetes-Cluster auf einem Outpost erstellen

Diese Beispielrichtlinie umfasst die Mindestberechtigungen, die erforderlich sind, um einen lokalen Amazon EKS-Cluster mit *`my-cluster`* dem Namen eines Außenpostens in der *`us-west-2`* AWS Region zu erstellen. Sie können die AWS Region durch die AWS Region ersetzen, in der Sie einen Cluster erstellen möchten. Wenn Sie eine Warnung sehen, die besagt, dass die Aktionen in Ihrer Richtlinie keine Berechtigungen auf Ressourcenebene unterstützen und Sie sich für die entscheiden **`All resources`** müssen AWS-Managementkonsole, können Sie diese problemlos ignorieren. Wenn Ihr Konto bereits über die `AWSServiceRoleForAmazonEKSLocalOutpost`-Rolle verfügt, können Sie die `iam:CreateServiceLinkedRole`-Aktion aus der Richtlinie entfernen. Wenn Sie jemals einen lokalen Amazon-EKS-Cluster auf einem Outpost in Ihrem Konto erstellt haben, ist diese Rolle bereits vorhanden, sofern Sie sie nicht gelöscht haben.

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": "eks:CreateCluster",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-cluster"
        },
        {
            "Action": [
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "iam:GetRole"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::111122223333:role/aws-service-role/outposts.eks-
local.amazonaws.com/AWSServiceRoleForAmazonEKSLocalOutpost"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole",
                "iam:ListAttachedRolePolicies"
            ],
            "Resource": "arn:aws:iam::111122223333:role/cluster-role-name"
        },
        {
            "Action": [
                "iam:CreateInstanceProfile",
                "iam:TagInstanceProfile",
                "iam:AddRoleToInstanceProfile",
                "iam:GetInstanceProfile",
                "iam:DeleteInstanceProfile",
                "iam:RemoveRoleFromInstanceProfile"
            ],
            "Resource": "arn:aws:iam::*:instance-profile/eks-local-*",
            "Effect": "Allow"
        }
    ]
}
```

Aktualisieren eines Kubernetes-Clusters

Diese Beispielrichtlinie umfasst die Mindestberechtigungen, die zum Aktualisieren eines Clusters mit dem Namen der *my-cluster* Region us-west-2 AWS erforderlich sind.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "eks:UpdateClusterVersion",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-cluster"
        }
    ]
}
```

Auflisten oder Beschreiben aller Cluster

Diese Beispielrichtlinie enthält die benötigten Mindestberechtigungen zum Auflisten und Beschreiben aller Cluster in Ihrem Konto. Ein IAM-Prinzipal muss in der Lage sein, Cluster aufzulisten und zu beschreiben, um den `update-kubeconfig` AWS CLI-Befehl verwenden zu können.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks:DescribeCluster",
                "eks:ListClusters"
            ],
            "Resource": "*"
        }
    ]
}
```

Verwendung von serviceverknüpften Rollen für Amazon EKS

Amazon Elastic Kubernetes Service verwendet serviceverknüpfte Rollen für AWS Identity and Access Management (IAM). Eine servicegebundene Rolle ist eine einzigartige Art von IAM-Rolle, die direkt

mit Amazon EKS verknüpft ist. Serviceverknüpfte Rollen werden von Amazon EKS vordefiniert und schließen alle Berechtigungen ein, die der Service zum Aufrufen anderer AWS-Services in Ihrem Namen erfordert.

Themen

- Verwenden von Rollen für Amazon-EKS-Cluster
- Verwenden von Rollen für Amazon-EKS-Knotengruppen
- Verwenden von Rollen für Amazon-EKS-Fargate-Profile
- Verbinden eines Kubernetes-Clusters mithilfe von Rollen mit Amazon EKS
- Verwenden von Rollen für lokale Amazon-EKS-Cluster in Outpost
- Verwendung von Rollen für das Amazon-EKS-Dashboard

Verwenden von Rollen für Amazon-EKS-Cluster

Amazon Elastic Kubernetes Service verwendet serviceverknüpfte Rollen für AWS Identity and Access Management (IAM). Eine servicegebundene Rolle ist eine einzigartige Art von IAM-Rolle, die direkt mit Amazon EKS verknüpft ist. Serviceverknüpfte Rollen werden von Amazon EKS vordefiniert und schließen alle Berechtigungen ein, die der Service zum Aufrufen anderer AWS-Services in Ihrem Namen erfordert.

Eine serviceverknüpfte Rolle macht die Einrichtung von Amazon EKS einfacher, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. Amazon EKS definiert die Berechtigungen seiner mit dem Service verbundenen Rollen, und sofern nicht anders definiert, kann nur Amazon EKS seine Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keinen anderen IAM-Entitäten zugewiesen werden.

Sie können eine serviceverknüpfte Rolle erst löschen, nachdem ihre verwandten Ressourcen gelöscht wurden. Dies schützt Ihre Amazon-EKS-Ressourcen, da Sie nicht versehentlich die Zugriffsberechtigung für die Ressourcen entfernen können.

Informationen zu anderen Services, die serviceverknüpfte Rollen unterstützen, finden Sie unter AWS-Services, die mit IAM funktionieren. Suchen Sie nach den Services, für die Ja in der Spalte Serviceverknüpfte Rolle angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer servicegebundenen Rolle für diesen Service anzuzeigen.

Serviceverknüpfte Rollenberechtigungen für Amazon EKS

Amazon ESK verwendet die serviceverknüpfte Rolle mit dem Namen `AWSServiceRoleForAmazonEKS`. Diese Rolle ermöglicht es Amazon EKS, Cluster in Ihrem Konto zu verwalten. Die angehängten Richtlinien ermöglichen es der Rolle, die folgenden Ressourcen zu verwalten: Netzwerkschnittstellen, Sicherheitsgruppen, Protokolle und VPCs.

> **Note**
>
> Die serviceverknüpfte `AWSServiceRoleForAmazonEKS`-Rolle unterscheidet sich von der Rolle, die für die Clustererstellung erforderlich ist. Weitere Informationen finden Sie unter the section called “Cluster-IAM-Rolle”.

Die serviceverknüpfte Rolle `AWSServiceRoleForAmazonEKS` vertraut darauf, dass die folgenden Services die Rolle annehmen:

- `eks.amazonaws.com`

Die Richtlinie für Rollenberechtigungen erlaubt Amazon EKS, die folgenden Aktionen auf den angegebenen Ressourcen durchzuführen:

- AmazonEKSServiceRolePolicy

Sie müssen Berechtigungen konfigurieren, damit eine juristische Stelle von IAM (z. B. Benutzer, Gruppe oder Rolle) eine serviceverknüpfte Rolle erstellen, bearbeiten oder löschen kann. Weitere Informationen finden Sie unter Serviceverknüpfte Rollenberechtigung im IAM-Benutzerhandbuch.

Erstellen einer serviceverknüpften Rolle für Amazon EKS

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie ein Cluster in der AWS-Managementkonsole, der AWS-CLI oder AWS-API ausführen, erstellt Amazon EKS die serviceverknüpfte Rolle für Sie.

Wenn Sie diese serviceverknüpfte Rolle löschen und sie dann erneut erstellen müssen, können Sie dasselbe Verfahren anwenden, um die Rolle in Ihrem Konto neu anzulegen. Wenn Sie einen Cluster anlegen, legt Amazon EKS die servicegebundene Rolle wieder für Sie an.

Bearbeiten einer serviceverknüpften Rolle für Amazon EKS

Amazon EKS verhindert die Bearbeitung der serviceverknüpften Rolle `AWSServiceRoleForAmazonEKS`. Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollenname nach dem Erstellen einer servicegebundenen Rolle nicht mehr geändert werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Löschen einer serviceverknüpften Rolle für Amazon EKS

Wenn Sie ein Feature oder einen Dienst, die bzw. der eine serviceverknüpften Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle löschen. Auf diese Weise haben Sie keine ungenutzte juristische Stelle, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch Ihre serviceverknüpfte Rolle zunächst bereinigen, bevor Sie sie manuell löschen können.

Bereinigen einer serviceverknüpften Rolle

Bevor mit IAM eine serviceverknüpfte Rolle löschen können, müssen Sie zunächst alle von der Rolle verwendeten Ressourcen löschen.

> **Note**
>
> Wenn der Amazon-EKS-Service die Rolle verwendet, wenn Sie versuchen, die Ressourcen zu löschen, schlägt der Löschvorgang möglicherweise fehl. Wenn das passiert, warten Sie einige Minuten und versuchen Sie es erneut.

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich die Option Cluster aus.
3. Wenn Ihr Cluster über Knotengruppen oder Fargate-Profile verfügt, müssen Sie diese löschen, bevor Sie den Cluster löschen können. Weitere Informationen erhalten Sie unter the section called “Delete” und the section called “Profile löschen”.
4. Wählen Sie auf der Seite Cluster den Cluster aus, den Sie löschen möchten, und wählen Sie Löschen.
5. Geben Sie den Namen des Clusters im Bestätigungsfenster zum Löschen ein, und wählen Sie dann Bestätigen.
6. Wiederholen Sie diesen Vorgang für alle anderen Cluster in Ihrem Konto. Warten Sie, bis alle Löschoperationen abgeschlossen sind.

Manuelles Löschen der -serviceverknüpften Rolle

Sie können die serviceverknüpfte Rolle `AWSServiceRoleForAmazonEKS` über die IAM–Konsole, die AWS-CLI oder die API AWS löschen. Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Unterstützte Regionen für Amazon EKS serviceverknüpfte Rollen

Amazon EKS unterstützt die Verwendung von serviceverknüpften Rollen in allen Regionen, in denen der Service verfügbar ist. Weitere Informationen finden Sie unter Amazon-EKS-Endpunkte und -Kontingente.

Verwenden von Rollen für Amazon-EKS-Knotengruppen

Amazon EKS verwendet serviceverknüpfte Rollen von AWS AWS Identity and Access Management (IAM). Eine servicegebundene Rolle ist eine einzigartige Art von IAM-Rolle, die direkt mit Amazon EKS verknüpft ist. Serviceverknüpfte Rollen werden von Amazon EKS vordefiniert und schließen alle Berechtigungen ein, die der Service zum Aufrufen anderer AWS-Services in Ihrem Namen erfordert.

Eine serviceverknüpfte Rolle macht die Einrichtung von Amazon EKS einfacher, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. Amazon EKS definiert die Berechtigungen seiner mit dem Service verbundenen Rollen, und sofern nicht anders definiert, kann nur Amazon EKS seine Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keinen anderen IAM-Entitäten zugewiesen werden.

Sie können eine serviceverknüpfte Rolle erst löschen, nachdem ihre verwandten Ressourcen gelöscht wurden. Dies schützt Ihre Amazon-EKS-Ressourcen, da Sie nicht versehentlich die Zugriffsberechtigung für die Ressourcen entfernen können.

Informationen zu anderen Services, die serviceverknüpfte Rollen unterstützen, finden Sie unter AWS-Services, die mit IAM funktionieren. Suchen Sie nach den Services, für die Ja in der Spalte Serviceverknüpfte Rolle angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer servicegebundenen Rolle für diesen Service anzuzeigen.

Serviceverknüpfte Rollenberechtigungen für Amazon EKS

Amazon ESK verwendet die serviceverknüpfte Rolle mit dem Namen `AWSServiceRoleForAmazonEKSNodegroup`. Mit dieser Rolle kann Amazon EKS die Knotengruppen in Ihrem Konto verwalten. Die angefügte `AWSServiceRoleForAmazonEKSNodegroup`-Richtlinie ermöglichen es der Rolle, die folgenden

Ressourcen zu verwalten: Auto-Scaling-Gruppen, Sicherheitsgruppen, Startvorlagen und IAM-Instance-Profile. Weitere Informationen finden Sie unter the section called “AWS verwaltete Richtlinie: AWSService RoleForAmazon EKSNodegroup”.

Die serviceverknüpfte Rolle `AWSServiceRoleForAmazonEKSNodegroup` vertraut darauf, dass die folgenden Services die Rolle annehmen:

- `eks-nodegroup.amazonaws.com`

Die Richtlinie für Rollenberechtigungen erlaubt Amazon EKS, die folgenden Aktionen auf den angegebenen Ressourcen durchzuführen:

- AWSServiceRoleForAmazonEKSNodegroup

Sie müssen Berechtigungen konfigurieren, damit eine juristische Stelle von IAM (z. B. Benutzer, Gruppe oder Rolle) eine serviceverknüpfte Rolle erstellen, bearbeiten oder löschen kann. Weitere Informationen finden Sie unter Serviceverknüpfte Rollenberechtigung im IAM-Benutzerhandbuch.

Erstellen einer serviceverknüpften Rolle für Amazon EKS

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie CreateNodegroup in der AWS-Managementkonsole, der AWS-CLI oder AWS-API ausführen, erstellt Amazon EKS die serviceverknüpfte Rolle für Sie.

> **Important**
>
> Diese serviceverknüpfte Rolle kann in Ihrem Konto erscheinen, wenn Sie eine Aktion in einem anderen Service abgeschlossen haben, der die von dieser Rolle unterstützten Funktionen verwendet. Wenn Sie den Amazon EKS-Service vor dem 1. Januar 2017 verwendet haben, als dieser serviceverknüpfte Rollen unterstützt, hat Amazon EKS die AWSServiceRoleForAmazonEKSNodegroup-Rolle in Ihrem Konto erstellt. Weitere Informationen finden Sie unter In meinem IAM-Konto wird eine neue Rolle angezeigt.

Erstellen einer serviceverknüpften Rolle in Amazon EKS (AWS-API)

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie eine verwaltete Knotengruppe in der AWS-Managementkonsole, der AWS-CLI oder AWS-API erstellen, erstellt Amazon EKS die serviceverknüpfte Rolle für Sie.

Wenn Sie diese serviceverknüpfte Rolle löschen und sie dann erneut erstellen müssen, können Sie dasselbe Verfahren anwenden, um die Rolle in Ihrem Konto neu anzulegen. Wenn Sie eine andere verwaltete Knotengruppe erstellen, erstellt Amazon EKS die serviceverknüpfte Rolle erneut für Sie.

Bearbeiten einer serviceverknüpften Rolle für Amazon EKS

Amazon EKS verhindert die Bearbeitung der serviceverknüpften Rolle `AWSServiceRoleForAmazonEKSNodegroup`. Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollenname nach dem Erstellen einer servicegebundenen Rolle nicht mehr geändert werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Löschen einer serviceverknüpften Rolle für Amazon EKS

Wenn Sie ein Feature oder einen Dienst, die bzw. der eine serviceverknüpften Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle löschen. Auf diese Weise haben Sie keine ungenutzte juristische Stelle, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch Ihre serviceverknüpfte Rolle zunächst bereinigen, bevor Sie sie manuell löschen können.

Bereinigen einer serviceverknüpften Rolle

Bevor mit IAM eine serviceverknüpfte Rolle löschen können, müssen Sie zunächst alle von der Rolle verwendeten Ressourcen löschen.

> **Note**
>
> Wenn der Amazon-EKS-Service die Rolle verwendet, wenn Sie versuchen, die Ressourcen zu löschen, schlägt der Löschvorgang möglicherweise fehl. Wenn das passiert, warten Sie einige Minuten und versuchen Sie es erneut.

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich die Option Cluster aus.
3. Wählen Sie die Registerkarte Compute (Datenverarbeitung) aus.
4. Wählen Sie im Abschnitt Node groups (Knotengruppen) die zu löschende Knotengruppe aus.
5. Geben Sie den Namen der Knotengruppe im Bestätigungsfenster zum Löschen ein und wählen Sie dann Bestätigen.
6. Wiederholen Sie diesen Vorgang für alle anderen Knotengruppen im Cluster. Warten Sie, bis alle Löschoperationen abgeschlossen sind.

Manuelles Löschen der -serviceverknüpften Rolle

Sie können die serviceverknüpfte Rolle `AWSServiceRoleForAmazonEKSNodegroup` über die IAM–Konsole, die AWS-CLI oder die API AWS löschen. Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Unterstützte Regionen für Amazon EKS serviceverknüpfte Rollen

Amazon EKS unterstützt die Verwendung von serviceverknüpften Rollen in allen Regionen, in denen der Service verfügbar ist. Weitere Informationen finden Sie unter Amazon-EKS-Endpunkte und -Kontingente.

Verwenden von Rollen für Amazon-EKS-Fargate-Profile

Amazon EKS verwendet serviceverknüpfte Rollen von AWS AWS Identity and Access Management (IAM). Eine servicegebundene Rolle ist eine einzigartige Art von IAM-Rolle, die direkt mit Amazon EKS verknüpft ist. Serviceverknüpfte Rollen werden von Amazon EKS vordefiniert und schließen alle Berechtigungen ein, die der Service zum Aufrufen anderer AWS-Services in Ihrem Namen erfordert.

Eine serviceverknüpfte Rolle macht die Einrichtung von Amazon EKS einfacher, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. Amazon EKS definiert die Berechtigungen seiner mit dem Service verbundenen Rollen, und sofern nicht anders definiert, kann nur Amazon EKS seine Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keinen anderen IAM-Entitäten zugewiesen werden.

Sie können eine serviceverknüpfte Rolle erst löschen, nachdem ihre verwandten Ressourcen gelöscht wurden. Dies schützt Ihre Amazon-EKS-Ressourcen, da Sie nicht versehentlich die Zugriffsberechtigung für die Ressourcen entfernen können.

Informationen zu anderen Services, die serviceverknüpfte Rollen unterstützen, finden Sie unter AWS-Services, die mit IAM funktionieren. Suchen Sie nach den Services, für die Ja in der Spalte Serviceverknüpfte Rolle angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer servicegebundenen Rolle für diesen Service anzuzeigen.

Serviceverknüpfte Rollenberechtigungen für Amazon EKS

Amazon ESK verwendet die serviceverknüpfte Rolle mit dem Namen `AWSServiceRoleForAmazonEKSForFargate`. Die Rolle ermöglicht Amazon EKS Fargate, das für Fargate Pods erforderliche VPC-Netzwerk zu konfigurieren. Die angehängten Richtlinien ermöglichen

es der Rolle, Elastic Network Interfaces zu erstellen und zu löschen und Elastic Network Interfaces und Ressourcen zu beschreiben.

Die serviceverknüpfte Rolle `AWSServiceRoleForAmazonEKSForFargate` vertraut darauf, dass die folgenden Services die Rolle annehmen:

- `eks-fargate.amazonaws.com`

Die Richtlinie für Rollenberechtigungen erlaubt Amazon EKS, die folgenden Aktionen auf den angegebenen Ressourcen durchzuführen:

- AmazonEKSForFargateServiceRolePolicy

Sie müssen Berechtigungen konfigurieren, damit eine juristische Stelle von IAM (z. B. Benutzer, Gruppe oder Rolle) eine serviceverknüpfte Rolle erstellen, bearbeiten oder löschen kann. Weitere Informationen finden Sie unter Serviceverknüpfte Rollenberechtigung im IAM-Benutzerhandbuch.

Erstellen einer serviceverknüpften Rolle für Amazon EKS

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie ein Fargate-Profil in der AWS-Managementkonsole, der AWS-CLI oder der AWS-API erstellen, erstellt Amazon EKS die serviceverknüpfte Rolle für Sie.

> **Important**
>
> Diese serviceverknüpfte Rolle kann in Ihrem Konto erscheinen, wenn Sie eine Aktion in einem anderen Service abgeschlossen haben, der die von dieser Rolle unterstützten Features verwendet. Wenn Sie den Amazon-EKS-Service vor dem 13. Dezember 2019 verwendet haben, als dieser Service-verknüpfte Rollen unterstützt, hat Amazon EKS die AWSServiceRoleForAmazonEKSForFargate-Rolle in Ihrem Konto erstellt. Weitere Informationen finden Sie unter Eine neue Rolle ist in meinem IAM-Konto erschienen.

Erstellen einer serviceverknüpften Rolle in Amazon EKS (AWS-API)

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie ein Fargate-Profil in der AWS-Managementkonsole, der AWS-CLI oder der AWS-API erstellen, erstellt Amazon EKS die serviceverknüpfte Rolle für Sie.

Wenn Sie diese serviceverknüpfte Rolle löschen und sie dann erneut erstellen müssen, können Sie dasselbe Verfahren anwenden, um die Rolle in Ihrem Konto neu anzulegen. Wenn Sie eine andere verwaltete Knotengruppe erstellen, erstellt Amazon EKS die serviceverknüpfte Rolle erneut für Sie.

Bearbeiten einer serviceverknüpften Rolle für Amazon EKS

Amazon EKS verhindert die Bearbeitung der serviceverknüpften Rolle `AWSServiceRoleForAmazonEKSForFargate`. Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollenname nach dem Erstellen einer servicegebundenen Rolle nicht mehr geändert werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Löschen einer serviceverknüpften Rolle für Amazon EKS

Wenn Sie ein Feature oder einen Dienst, die bzw. der eine serviceverknüpften Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle löschen. Auf diese Weise haben Sie keine ungenutzte juristische Stelle, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch Ihre serviceverknüpfte Rolle zunächst bereinigen, bevor Sie sie manuell löschen können.

Bereinigen einer serviceverknüpften Rolle

Bevor mit IAM eine serviceverknüpfte Rolle löschen können, müssen Sie zunächst alle von der Rolle verwendeten Ressourcen löschen.

> **Note**
>
> Wenn der Amazon-EKS-Service die Rolle verwendet, wenn Sie versuchen, die Ressourcen zu löschen, schlägt der Löschvorgang möglicherweise fehl. Wenn das passiert, warten Sie einige Minuten und versuchen Sie es erneut.

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich Cluster aus.
3. Wählen Sie auf der Seite Cluster (Cluster) den Cluster aus.
4. Wählen Sie die Registerkarte Compute (Datenverarbeitung) aus.
5. Wenn im Abschnitt Fargate profiles (Fargate-Profile) Fargate-Profile vorhanden sind, wählen Sie jedes einzeln aus und wählen Sie dann Delete (Löschen).

6. Geben Sie den Namen des Profils im Bestätigungsfenster zum Löschen ein, und wählen Sie dann Bestätigen.
7. Wiederholen Sie diesen Vorgang für alle anderen Fargate-Profile im Cluster und für alle anderen Cluster in Ihrem Konto.

Manuelles Löschen der -serviceverknüpften Rolle

Verwenden Sie die IAM-Konsole, die AWS CLI oder die AWS-API, um die serviceverknüpfte Rolle AWSServiceRoleForAmazonEKSForFargate zu löschen. Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Unterstützte Regionen für Amazon EKS serviceverknüpfte Rollen

Amazon EKS unterstützt die Verwendung von serviceverknüpften Rollen in allen Regionen, in denen der Service verfügbar ist. Weitere Informationen finden Sie unter Amazon-EKS-Endpunkte und -Kontingente.

Verbinden eines Kubernetes-Clusters mithilfe von Rollen mit Amazon EKS

Amazon EKS verwendet dienstbezogene Rollen für AWS Identity and Access Management (IAM). Eine servicegebundene Rolle ist eine einzigartige Art von IAM-Rolle, die direkt mit Amazon EKS verknüpft ist. Servicebezogene Rollen sind von Amazon EKS vordefiniert und beinhalten alle Berechtigungen, die der Service benötigt, um andere AWS Services in Ihrem Namen aufzurufen.

Eine serviceverknüpfte Rolle macht die Einrichtung von Amazon EKS einfacher, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. Amazon EKS definiert die Berechtigungen seiner mit dem Service verbundenen Rollen, und sofern nicht anders definiert, kann nur Amazon EKS seine Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keinen anderen IAM-Entitäten zugewiesen werden.

Sie können eine serviceverknüpfte Rolle erst löschen, nachdem ihre verwandten Ressourcen gelöscht wurden. Dies schützt Ihre Amazon-EKS-Ressourcen, da Sie nicht versehentlich die Zugriffsberechtigung für die Ressourcen entfernen können.

Informationen zu anderen Services, die serviceverknüpfte Rollen unterstützen, finden Sie unter AWS -Services, die mit IAM funktionieren. Suchen Sie nach den Services, für die Ja in der Spalte Serviceverknüpfte Rolle angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer servicegebundenen Rolle für diesen Service anzuzeigen.

Serviceverknüpfte Rollenberechtigungen für Amazon EKS

Amazon ESK verwendet die serviceverknüpfte Rolle mit dem Namen `AWSServiceRoleForAmazonEKSConnector`. Diese Rolle ermöglicht es Amazon EKS, Kubernetes-Cluster zu verbinden. Die angefügten Richtlinien ermöglichen es der Rolle, die erforderlichen Ressourcen für die Verbindung mit Ihrem registrierten Kubernetes-Cluster zu verwalten.

Die serviceverknüpfte Rolle `AWSServiceRoleForAmazonEKSConnector` vertraut darauf, dass die folgenden Services die Rolle annehmen:

- `eks-connector.amazonaws.com`

Die Richtlinie für Rollenberechtigungen erlaubt Amazon EKS, die folgenden Aktionen auf den angegebenen Ressourcen durchzuführen:

- AmazonEKSConnectorServiceRolePolicy

Sie müssen Berechtigungen konfigurieren, damit eine juristische Stelle von IAM (z. B. Benutzer, Gruppe oder Rolle) eine serviceverknüpfte Rolle erstellen, bearbeiten oder löschen kann. Weitere Informationen finden Sie unter Serviceverknüpfte Rollenberechtigung im IAM-Benutzerhandbuch.

Diese Rolle verwendet SSM-Berechtigungen (Systems Manager), um sichere Verbindungen herzustellen und verbundene Kubernetes-Cluster zu verwalten.

Erstellen einer serviceverknüpften Rolle für Amazon EKS

Sie müssen keine serviceverknüpfte Rolle manuell erstellen, um eine Verbindung mit einem Cluster herzustellen. Wenn Sie einen Cluster in der AWS-Managementkonsole, der AWS CLI oder der AWS API verbinden`eksctl`, erstellt Amazon EKS die serviceverknüpfte Rolle für Sie.

Wenn Sie diese serviceverknüpfte Rolle löschen und sie dann erneut erstellen müssen, können Sie dasselbe Verfahren anwenden, um die Rolle in Ihrem Konto neu anzulegen. Wenn Sie einen Cluster verbinden, legt Amazon EKS die servicegebundene Rolle wieder für Sie an.

Bearbeiten einer serviceverknüpften Rolle für Amazon EKS

Amazon EKS verhindert die Bearbeitung der serviceverknüpften Rolle `AWSServiceRoleForAmazonEKSConnector`. Da möglicherweise verschiedene Entitäten auf die

Rolle verweisen, kann der Rollenname nach dem Erstellen einer servicegebundenen Rolle nicht mehr geändert werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Löschen einer serviceverknüpften Rolle für Amazon EKS

Wenn Sie ein Feature oder einen Dienst, die bzw. der eine serviceverknüpften Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle löschen. Auf diese Weise haben Sie keine ungenutzte juristische Stelle, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch Ihre serviceverknüpfte Rolle zunächst bereinigen, bevor Sie sie manuell löschen können.

Bereinigen einer serviceverknüpften Rolle

Bevor mit IAM eine serviceverknüpfte Rolle löschen können, müssen Sie zunächst alle von der Rolle verwendeten Ressourcen löschen.

> **Note**
>
> Wenn der Amazon-EKS-Service die Rolle verwendet, wenn Sie versuchen, die Ressourcen zu löschen, schlägt der Löschvorgang möglicherweise fehl. Wenn das passiert, warten Sie einige Minuten und versuchen Sie es erneut.

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich Cluster aus.
3. Wählen Sie auf der Seite Cluster (Cluster) den Cluster aus.
4. Wählen Sie die Registerkarte Abmelden und dann die Registerkarte OK.

Manuelles Löschen der serviceverknüpften Rolle

Verwenden Sie die IAM-Konsole, die AWS CLI oder die AWS API, um die AWSService RoleForAmazon EKSConnector serviceverknüpfte Rolle zu löschen. Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Verwenden von Rollen für lokale Amazon-EKS-Cluster in Outpost

Amazon Elastic Kubernetes Service verwendet serviceverknüpfte Rollen für AWS Identity and Access Management (IAM). Eine servicegebundene Rolle ist eine einzigartige Art von IAM-Rolle, die direkt

mit Amazon EKS verknüpft ist. Serviceverknüpfte Rollen werden von Amazon EKS vordefiniert und schließen alle Berechtigungen ein, die der Service zum Aufrufen anderer AWS-Services in Ihrem Namen erfordert.

Eine serviceverknüpfte Rolle macht die Einrichtung von Amazon EKS einfacher, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. Amazon EKS definiert die Berechtigungen seiner mit dem Service verbundenen Rollen, und sofern nicht anders definiert, kann nur Amazon EKS seine Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keinen anderen IAM-Entitäten zugewiesen werden.

Sie können eine serviceverknüpfte Rolle erst löschen, nachdem ihre verwandten Ressourcen gelöscht wurden. Dies schützt Ihre Amazon-EKS-Ressourcen, da Sie nicht versehentlich die Zugriffsberechtigung für die Ressourcen entfernen können.

Informationen zu anderen Services, die serviceverknüpfte Rollen unterstützen, finden Sie unter AWS-Services, die mit IAM funktionieren. Suchen Sie nach den Services, für die Ja in der Spalte Serviceverknüpfte Rolle angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer servicegebundenen Rolle für diesen Service anzuzeigen.

Serviceverknüpfte Rollenberechtigungen für Amazon EKS

Amazon ESK verwendet die serviceverknüpfte Rolle mit dem Namen `AWSServiceRoleForAmazonEKSLocalOutpost`. Mit dieser Rolle können Sie Amazon EKS dazu konfigurieren, lokale Cluster in Ihrem Konto zu verwalten. Die angehängten Richtlinien ermöglichen es der Rolle, die folgenden Ressourcen zu verwalten: Netzwerkschnittstellen, Sicherheitsgruppen, Protokolle und Amazon-EC2-Instances.

> **Note**
>
> Die serviceverknüpfte `AWSServiceRoleForAmazonEKSLocalOutpost`-Rolle unterscheidet sich von der Rolle, die für die Clustererstellung erforderlich ist. Weitere Informationen finden Sie unter the section called “Cluster-IAM-Rolle”.

Die serviceverknüpfte Rolle `AWSServiceRoleForAmazonEKSLocalOutpost` vertraut darauf, dass die folgenden Services die Rolle annehmen:

- `outposts.eks-local.amazonaws.com`

Die Richtlinie für Rollenberechtigungen erlaubt Amazon EKS, die folgenden Aktionen auf den angegebenen Ressourcen durchzuführen:

- AmazonEKSServiceRolePolicy

Sie müssen Berechtigungen konfigurieren, damit eine juristische Stelle von IAM (z. B. Benutzer, Gruppe oder Rolle) eine serviceverknüpfte Rolle erstellen, bearbeiten oder löschen kann. Weitere Informationen finden Sie unter Serviceverknüpfte Rollenberechtigung im IAM-Benutzerhandbuch.

Erstellen einer serviceverknüpften Rolle für Amazon EKS

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie ein Cluster in der AWS-Managementkonsole, der AWS-CLI oder AWS-API ausführen, erstellt Amazon EKS die serviceverknüpfte Rolle für Sie.

Wenn Sie diese serviceverknüpfte Rolle löschen und sie dann erneut erstellen müssen, können Sie dasselbe Verfahren anwenden, um die Rolle in Ihrem Konto neu anzulegen. Wenn Sie einen Cluster anlegen, legt Amazon EKS die servicegebundene Rolle wieder für Sie an.

Bearbeiten einer serviceverknüpften Rolle für Amazon EKS

Amazon EKS verhindert die Bearbeitung der serviceverknüpften Rolle `AWSServiceRoleForAmazonEKSLocalOutpost`. Nachdem Sie eine serviceverknüpfte Rolle erstellt haben, können Sie den Namen der Rolle nicht mehr ändern, da verschiedene Entitäten auf die Rolle verweisen könnten. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Löschen einer serviceverknüpften Rolle für Amazon EKS

Wenn Sie ein Feature oder einen Dienst, die bzw. der eine serviceverknüpften Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle löschen. Auf diese Weise haben Sie keine ungenutzte juristische Stelle, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch Ihre serviceverknüpfte Rolle zunächst bereinigen, bevor Sie sie manuell löschen können.

Bereinigen einer serviceverknüpften Rolle

Bevor mit IAM eine serviceverknüpfte Rolle löschen können, müssen Sie zunächst alle von der Rolle verwendeten Ressourcen löschen.

> **Note**
>
> Wenn der Amazon-EKS-Service die Rolle verwendet, wenn Sie versuchen, die Ressourcen zu löschen, schlägt der Löschvorgang möglicherweise fehl. Wenn das passiert, warten Sie einige Minuten und versuchen Sie es erneut.

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie im linken Navigationsbereich Clusters (Cluster) für Amazon EKS aus.
3. Wenn Ihr Cluster über Knotengruppen oder Fargate-Profile verfügt, müssen Sie diese löschen, bevor Sie den Cluster löschen können. Weitere Informationen erhalten Sie unter the section called “Delete” und the section called “Profile löschen”.
4. Wählen Sie auf der Seite Cluster den Cluster aus, den Sie löschen möchten, und wählen Sie Löschen.
5. Geben Sie den Namen des Clusters im Bestätigungsfenster zum Löschen ein, und wählen Sie dann Bestätigen.
6. Wiederholen Sie diesen Vorgang für alle anderen Cluster in Ihrem Konto. Warten Sie, bis alle Löschoperationen abgeschlossen sind.

Manuelles Löschen der -serviceverknüpften Rolle

Sie können die serviceverknüpfte Rolle `AWSServiceRoleForAmazonEKSLocalOutpost` über die IAM–Konsole, die AWS-CLI oder die API AWS löschen. Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Unterstützte Regionen für Amazon EKS serviceverknüpfte Rollen

Amazon EKS unterstützt die Verwendung von serviceverknüpften Rollen in allen Regionen, in denen der Service verfügbar ist. Weitere Informationen finden Sie unter Amazon-EKS-Endpunkte und -Kontingente.

Verwendung von Rollen für das Amazon-EKS-Dashboard

Das Amazon-EKS-Dashboard verwendet diese serviceverknüpfte Rolle, um Informationen über Cluster-Ressourcen aus mehreren Regionen und Konten zu aggregieren. Das Dashboard lässt sich in AWS Organizations integrieren, um Informationen zu mehreren Konten in Ihrem Unternehmen sicher zu lesen.

Weitere Informationen zum Amazon-EKS-Dashboard finden Sie unter the section called “Amazon-EKS-Dashboard”.

Hintergrund

Amazon EKS verwendet serviceverknüpfte Rollen von AWS AWS Identity and Access Management (IAM). Eine servicegebundene Rolle ist eine einzigartige Art von IAM-Rolle, die direkt mit Amazon EKS verknüpft ist. Serviceverknüpfte Rollen werden von Amazon EKS vordefiniert und schließen alle Berechtigungen ein, die der Service zum Aufrufen anderer AWS-Services in Ihrem Namen erfordert.

Eine serviceverknüpfte Rolle macht die Einrichtung von Amazon EKS einfacher, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. Amazon EKS definiert die Berechtigungen seiner mit dem Service verbundenen Rollen, und sofern nicht anders definiert, kann nur Amazon EKS seine Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keinen anderen IAM-Entitäten zugewiesen werden.

Sie können eine serviceverknüpfte Rolle erst löschen, nachdem ihre verwandten Ressourcen gelöscht wurden. Dies schützt Ihre Amazon-EKS-Ressourcen, da Sie nicht versehentlich die Zugriffsberechtigung für die Ressourcen entfernen können.

Informationen zu anderen Services, die serviceverknüpfte Rollen unterstützen, finden Sie unter AWS-Services, die mit IAM funktionieren. Suchen Sie nach den Services, für die Ja in der Spalte Serviceverknüpfte Rolle angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer servicegebundenen Rolle für diesen Service anzuzeigen.

Serviceverknüpfte Rollenberechtigungen für Amazon EKS

Amazon ESK verwendet die serviceverknüpfte Rolle mit dem Namen `AWSServiceRoleForAmazonEKSDashboard`. Mit dieser Rolle kann Amazon EKS das EKS-Dashboard generieren und anzeigen, einschließlich aggregierter Information zu Cluster-Ressourcen. Die angefügte `AmazonEKSDashboardServiceRolePolicy`-Richtlinie ermöglichen es der Rolle, die folgenden Ressourcen zu verwalten: Auto-Scaling-Gruppen, Sicherheitsgruppen, Startvorlagen und IAM-Instance-Profile. Weitere Informationen finden Sie unter the section called “AWS verwaltete Richtlinie: Amazon EKSDashboard ServiceRolePolicy”.

Die serviceverknüpfte Rolle `AWSServiceRoleForAmazonEKSDashboard` vertraut darauf, dass die folgenden Services die Rolle annehmen:

- `dashboard.eks.amazonaws.com`

Die neueste Version des JSON-Richtliniendokuments finden Sie unter AmazonEKSDashboardServiceRolePolicy im Referenzhandbuch zu von AWSverwalteten Richtlinien.

Sie müssen Berechtigungen konfigurieren, damit eine juristische Stelle von IAM (z. B. Benutzer, Gruppe oder Rolle) eine serviceverknüpfte Rolle erstellen, bearbeiten oder löschen kann. Weitere Informationen finden Sie unter Serviceverknüpfte Rollenberechtigung im IAM-Benutzerhandbuch.

Erstellen einer serviceverknüpften Rolle für Amazon EKS

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie das Dashboard in der AWS-Konsole aktivieren, erstellt Amazon EKS die serviceverknüpfte Rolle für Sie.

Weitere Informationen zum Aktivieren des EKS-Dashboards finden Sie unter the section called “Organizations konfigurieren”.

> **Important**
>
> Diese serviceverknüpfte Rolle kann in Ihrem Konto erscheinen, wenn Sie eine Aktion in einem anderen Service abgeschlossen haben, der die von dieser Rolle unterstützten Features verwendet.

Bearbeiten einer serviceverknüpften Rolle für Amazon EKS

Amazon EKS verhindert die Bearbeitung der serviceverknüpften Rolle `AWSServiceRoleForAmazonEKSDashboard`. Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollenname nach dem Erstellen einer servicegebundenen Rolle nicht mehr geändert werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Löschen einer serviceverknüpften Rolle für Amazon EKS

Wenn Sie ein Feature oder einen Dienst, die bzw. der eine serviceverknüpften Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle löschen. Auf diese Weise haben Sie keine ungenutzte juristische Stelle, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch Ihre serviceverknüpfte Rolle zunächst bereinigen, bevor Sie sie manuell löschen können.

Bereinigen einer serviceverknüpften Rolle

Bevor mit IAM eine serviceverknüpfte Rolle löschen können, müssen Sie zunächst alle von der Rolle verwendeten Ressourcen löschen.

> **Note**
>
> Wenn der Amazon-EKS-Service die Rolle verwendet, wenn Sie versuchen, die Ressourcen zu löschen, schlägt der Löschvorgang möglicherweise fehl. Wenn das passiert, warten Sie einige Minuten und versuchen Sie es erneut.

Weitere Informationen zum Deaktivieren des EKS-Dashboards finden Sie unter the section called “Organizations konfigurieren”.

Manuelles Löschen der -serviceverknüpften Rolle

Sie können die serviceverknüpfte Rolle `AWSServiceRoleForAmazonEKSDashboard` über die IAM–Konsole, die AWS-CLI oder die API AWS löschen. Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Unterstützte Regionen für Amazon EKS serviceverknüpfte Rollen

Amazon EKS unterstützt die Verwendung von serviceverknüpften Rollen in allen Regionen, in denen der Service verfügbar ist. Weitere Informationen finden Sie unter Amazon-EKS-Endpunkte und -Kontingente.

IAM-Rolle für die Ausführung von Amazon-EKS-Pods

Die Amazon EKS Pod-Ausführungsrolle ist erforderlich, um Pods auf der AWS Fargate-Infrastruktur auszuführen.

Wenn Ihr Cluster Pods auf der AWS Fargate-Infrastruktur erstellt, müssen die Komponenten, die auf der Fargate-Infrastruktur ausgeführt werden, in AWS APIs Ihrem Namen Aufrufe tätigen. Auf diese Weise können sie Aktionen wie das Abrufen von Container-Images aus Amazon ECR oder das Weiterleiten von Protokollen an andere AWS Dienste ausführen. Die Amazon-EKS-Pod-Ausführungsrolle stellt die entsprechenden IAM-Berechtigungen bereit.

Wenn Sie ein Fargate-Profil erstellen, müssen Sie eine Pod-Ausführungsrolle für die Amazon-EKS-Komponenten angeben, die in der Fargate-Infrastruktur mit dem Profil ausgeführt werden. Diese Rolle wird zur Autorisierung der Rollenbasierten Zugriffskontrolle (RBAC) von Kubernetes des Clusters hinzugefügt. Auf diese Weise kann sich das `kubelet`, das in der Fargate-Infrastruktur ausgeführt wird, bei Ihrem Amazon-EKS-Cluster registrieren, sodass es als Knoten in Ihrem Cluster angezeigt werden kann.

> **Note**
>
> Das Fargate-Profil muss eine andere IAM-Rolle haben als EC2 Amazon-Knotengruppen.

> **Important**
>
> Die im Fargate-Pod ausgeführten Container können nicht die IAM-Berechtigungen annehmen, die einer Pod-Ausführungsrolle zugeordnet sind. Um den Containern in Ihrem Fargate-Pod Zugriff auf andere AWS Dienste zu gewähren, müssen Sie IAM-Rollen für Dienstkonten verwenden.

Bevor Sie ein Fargate-Profil erstellen, müssen Sie eine IAM-Rolle bei Amazon erstellen. EKSFargate PodExecutionRolePolicy

Auf eine korrekt konfigurierte Pod-Ausführungsrolle prüfen

Mit dem folgenden Verfahren können Sie feststellen, ob Ihr Konto bereits über eine korrekt konfigurierte Amazon-EKS-Pod-Ausführungsrolle verfügt. Um Sicherheitsprobleme durch verwirrte Stellvertreter zu vermeiden, ist es wichtig, dass die Rolle den Zugriff basierend auf `SourceArn` beschränkt. Sie können die Ausführungsrolle nach Bedarf ändern, um Fargate-Profile auch auf anderen Clustern zu unterstützen.

1. Öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im linken Navigationsbereich Roles aus.
3. Suchen Sie auf der Seite Rollen in der Liste der Rollen für Amazon EKSFargate PodExecutionRole. Wenn die Rolle nicht vorhanden ist, finden Sie unter the section called “Erstellen der Amazon-EKS-Pod-Ausführungsrolle” Informationen zum Erstellen der Rolle. Wenn die Rolle vorhanden ist, wählen Sie sie aus.
4. Gehen Sie auf der EKSFargatePodExecutionRoleAmazon-Seite wie folgt vor:
   a. Wählen Sie Berechtigungen.
   b. Stellen Sie sicher, dass die von EKSFargatePodExecutionRolePolicyAmazon verwaltete Richtlinie mit der Rolle verknüpft ist.
   c. Wählen Sie Trust Relationships (Vertrauensbeziehungen) aus.
   d. Wählen Sie Edit trust policy (Vertrauensrichtlinie bearbeiten) aus.

5. Prüfen Sie auf der Seite Edit trust policy (Vertrauensrichtlinie bearbeiten), dass die Vertrauensbeziehung die folgende Richtlinie und eine Zeile für Fargate-Profile in Ihrem Cluster enthält. Wenn ja, wählen Sie Cancel (Abbrechen) aus.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Condition": {
         "ArnLike": {
            "aws:SourceArn": "arn:aws:eks:us-east-1:111122223333:fargateprofile/my-
cluster/*"
         }
      },
      "Principal": {
        "Service": "eks-fargate-pods.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Wenn die Richtlinie übereinstimmt, aber keine Zeile mit den Fargate-Profilen in Ihrem Cluster enthält, können Sie die folgende Zeile oben im `ArnLike`-Objekt hinzufügen. *`region-code`*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet, *`111122223333`* durch Ihre Konto-ID und *`my-cluster`* durch den Namen Ihres Clusters.

```
"aws:SourceArn": "arn:aws: eks:region-code:111122223333:fargateprofile/my-cluster/*",
```

Wenn die Richtlinie nicht übereinstimmt, kopieren Sie die vollständige vorherige Richtlinie in das Formular und wählen Sie Richtlinie aktualisieren aus. *`region-code`*Ersetzen Sie durch die AWS Region, in der sich Ihr Cluster befindet. Wenn Sie dieselbe Rolle in allen AWS Regionen Ihres Kontos verwenden möchten, *`region-code`* ersetzen Sie es durch*. Ersetzen Sie *`111122223333`* durch Ihre Konto-ID und *`my-cluster`* durch den Namen Ihres Clusters. Wenn Sie dieselbe Rolle für alle Cluster in Ihrem Konto verwenden möchten, ersetzen Sie *`my-cluster`* durch *.

Erstellen der Amazon-EKS-Pod-Ausführungsrolle

Wenn Sie noch nicht über die Amazon EKS-Pod-Ausführungsrolle für Ihren Cluster verfügen, können Sie die AWS-Managementkonsole oder die AWS CLI verwenden, um sie zu erstellen.

AWS-Managementkonsole

a. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
b. Wählen Sie im linken Navigationsbereich Roles aus.
c. Klicken Sie auf der Seite Roles (Rollen) auf Create role (Rolle erstellen).
d. Gehen Sie auf der Seite Select trusted entity (Vertrauenswürdige Entität auswählen) wie folgt vor:
   i. Wählen Sie im Abschnitt Vertrauenswürdiger Entitätstyp die Option AWS Service aus.
   ii. Wählen Sie aus der Dropdownliste Anwendungsfälle für andere AWS Dienste die Option EKS aus.
   iii. Wählen Sie EKS – Fargate-Pod aus.
   iv. Wählen Sie Weiter aus.
e. Wählen Sie auf der Seite Add permissions (Berechtigungen hinzufügen) die Option Next (Weiter) aus.
f. Gehen Sie auf der Seite Name, review, and create (Benennen, überprüfen und erstellen) wie folgt vor:
   i. Geben Sie unter Role name (Rollenname) einen eindeutigen Namen für die Rolle ein, z. B. `AmazonEKSFargatePodExecutionRole`.
   ii. Fügen Sie der Rolle unter Tags hinzufügen (optional) Metadaten hinzu, indem Sie Tags als Schlüssel-Wert-Paare anfügen. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen im IAM-Benutzerhandbuch.
   iii. Wählen Sie Rolle erstellen aus.
g. Suchen Sie auf der Seite Rollen in der Liste der Rollen für Amazon EKSFargate PodExecutionRole. Wählen Sie die Rolle aus.
h. Gehen Sie auf der EKSFargatePodExecutionRoleAmazon-Seite wie folgt vor:
   i. Wählen Sie Trust Relationships (Vertrauensbeziehungen) aus.
   ii. Wählen Sie Edit trust policy (Vertrauensrichtlinie bearbeiten) aus.
i. Führen Sie auf der Seite Edit trust policy (Vertrauensrichtlinie bearbeiten) die folgenden Schritte aus:

i. Kopieren Sie die folgenden Inhalte in das Formular unter Edit trust policy (Vertrauensrichtlinie bearbeiten). *region-code*Ersetzen Sie es durch die AWS Region, in der sich Ihr Cluster befindet. Wenn Sie dieselbe Rolle in allen AWS Regionen Ihres Kontos verwenden möchten, *region-code* ersetzen Sie es durch*. Ersetzen Sie *111122223333* durch Ihre Konto-ID und *my-cluster* durch den Namen Ihres Clusters. Wenn Sie dieselbe Rolle für alle Cluster in Ihrem Konto verwenden möchten, ersetzen Sie *my-cluster* durch *.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Condition": {
         "ArnLike": {
            "aws:SourceArn": "arn:aws:eks:us-
east-1:111122223333:fargateprofile/my-cluster/*"
         }
      },
      "Principal": {
        "Service": "eks-fargate-pods.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

ii. Wählen Sie Update policy.

AWS CLI

a. Kopieren Sie den folgenden Inhalt in eine Datei namens `pod-execution-role-trust-policy.json`. *region-code*Ersetzen Sie durch die AWS Region, in der sich Ihr Cluster befindet. Wenn Sie dieselbe Rolle in allen AWS Regionen Ihres Kontos verwenden möchten, *region-code* ersetzen Sie es durch*. Ersetzen Sie *111122223333* durch Ihre Konto-ID und *my-cluster* durch den Namen Ihres Clusters. Wenn Sie dieselbe Rolle für alle Cluster in Ihrem Konto verwenden möchten, ersetzen Sie *my-cluster* durch *.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Condition": {
         "ArnLike": {
            "aws:SourceArn": "arn:aws:eks:us-east-1:111122223333:fargateprofile/
my-cluster/*"
         }
      },
      "Principal": {
        "Service": "eks-fargate-pods.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

b. Erstellen Sie eine IAM-Rolle für die Pod-Ausführung.

```
aws iam create-role \
  --role-name AmazonEKSFargatePodExecutionRole \
  --assume-role-policy-document file://"pod-execution-role-trust-policy.json"
```

c. Hängen Sie die erforderliche von Amazon EKS verwaltete IAM-Richtlinie an die Rolle an.

```
aws iam attach-role-policy \
  --policy-arn arn:aws: iam::aws:policy/AmazonEKSFargatePodExecutionRolePolicy \
  --role-name AmazonEKSFargatePodExecutionRole
```

Amazon-EKS-Konnektor-IAM-Rolle

Sie können Kubernetes-Cluster verbinden, um sie in Ihrem anzuzeigen. AWS-Managementkonsole Um eine Verbindung zu einem Kubernetes-Cluster herzustellen, erstellen Sie eine IAM-Rolle.

Überprüfen auf eine vorhandene EKS-Connector-Rolle

Mit dem folgenden Verfahren können Sie feststellen, ob Ihr Konto bereits über die Amazon-EKS-Konnektorrolle verfügt.

1. Öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im linken Navigationsbereich Roles aus.

3. Suchen Sie in der Liste der Rollen nach `AmazonEKSConnectorAgentRole`. Wenn keine Rolle mit `AmazonEKSConnectorAgentRole` vorhanden ist, informieren Sie sich unter the section called “Erstellen der Rolle des Amazon-EKS-Connector-Agenten”, wie Sie die Rolle erstellen können. Wenn eine Rolle mit `AmazonEKSConnectorAgentRole` vorhanden ist, wählen Sie die Rolle aus, um die angefügten Richtlinien anzuzeigen.
4. Wählen Sie Berechtigungen.
5. Stellen Sie sicher, dass die von Amazon EKSConnector AgentPolicy verwaltete Richtlinie der Rolle zugeordnet ist. Wenn die Richtlinie angefügt ist, ist Ihre Amazon-EKS-Connector-Rolle ordnungsgemäß konfiguriert.
6. Wählen Sie Vertrauensstellungen und dann Vertrauensrichtlinie bearbeiten aus.
7. Überprüfen Sie, dass die Vertrauensstellung die folgende Richtlinie enthält. Wenn die Vertrauensstellung mit der folgenden Richtlinie übereinstimmt, wählen Sie Abbrechen aus. Andernfalls kopieren Sie die Richtlinie in das Fenster Vertrauensrichtlinie bearbeiten und wählen Sie Richtlinie aktualisieren aus.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Erstellen der Rolle des Amazon-EKS-Connector-Agenten

Sie können das AWS-Managementkonsole oder verwenden AWS CloudFormation , um die Connector-Agent-Rolle zu erstellen.

AWS CLI

a. Erstellen Sie eine Datei mit dem Namen `eks-connector-agent-trust-policy.json`, die die folgende JSON enthält, die für die IAM-Rolle verwendet werden soll.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

b. Erstellen Sie eine Datei mit dem Namen `eks-connector-agent-policy.json`, die die folgende JSON enthält, die für die IAM-Rolle verwendet werden soll.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "SsmControlChannel",
            "Effect": "Allow",
            "Action": [
                "ssmmessages:CreateControlChannel"
            ],
            "Resource": "arn:aws:eks:*:*:cluster/*"
        },
        {
            "Sid": "ssmDataplaneOperations",
            "Effect": "Allow",
            "Action": [
                "ssmmessages:CreateDataChannel",
                "ssmmessages:OpenDataChannel",
                "ssmmessages:OpenControlChannel"
            ],
            "Resource": "*"
        }
    ]
}
```

c. Erstellen Sie die Agentenrolle von Amazon EKS Connector mithilfe der Vertrauensrichtlinie und Richtlinie, die Sie in den vorherigen Listenelementen erstellt haben.

```
aws iam create-role \
     --role-name AmazonEKSConnectorAgentRole \
     --assume-role-policy-document file://eks-connector-agent-trust-policy.json
```

d. Hängen Sie die Richtlinie an Ihre Amazon-EKS-Connector-Agentenrolle an.

```
aws iam put-role-policy \
     --role-name AmazonEKSConnectorAgentRole \
     --policy-name AmazonEKSConnectorAgentPolicy \
     --policy-document file://eks-connector-agent-policy.json
```

AWS CloudFormation

a. Speichern Sie die folgende AWS CloudFormation Vorlage in einer Textdatei auf Ihrem lokalen System.

> **Note**
>
> Diese Vorlage erstellt auch die mit dem Service verknüpfte Rolle, die andernfalls beim Aufruf der `registerCluster`-API erstellt würde. Details dazu finden Sie unter the section called “Rolle des Cluster-Konnektors”.

```
---
AWSTemplateFormatVersion: '2010-09-09'
Description: 'Provisions necessary resources needed to register clusters in EKS'
Parameters: {}
Resources:
  EKSConnectorSLR:
    Type: AWS::IAM::ServiceLinkedRole
    Properties:
      AWSServiceName: eks-connector.amazonaws.com

  EKSConnectorAgentRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
```

```
          - Effect: Allow
            Action: [ 'sts:AssumeRole' ]
            Principal:
              Service: 'ssm.amazonaws.com'

  EKSConnectorAgentPolicy:
    Type: AWS::IAM::Policy
    Properties:
      PolicyName: EKSConnectorAgentPolicy
      Roles:
        - {Ref: 'EKSConnectorAgentRole'}
      PolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: 'Allow'
            Action: [ 'ssmmessages:CreateControlChannel' ]
            Resource:
            - Fn::Sub: 'arn:${AWS::Partition}:eks:*:*:cluster/*'
          - Effect: 'Allow'
            Action: [ 'ssmmessages:CreateDataChannel',
 'ssmmessages:OpenDataChannel', 'ssmmessages:OpenControlChannel' ]
            Resource: "*"
Outputs:
  EKSConnectorAgentRoleArn:
    Description: The agent role that EKS connector uses to communicate with AWS
 services.
    Value: !GetAtt EKSConnectorAgentRole.Arn
```

b. Öffnen Sie die AWS CloudFormation -Konsole.

c. Wählen Sie Stack erstellen mit neuen Ressourcen (Standard).

d. Wählen Sie für Specify template (Vorlage festlegen) Upload a template file (Vorlagendatei hochladen) aus und wählen Sie dann Choose file (Datei wählen).

e. Wählen Sie die zuvor erstellte Datei und klicken Sie dann auf Next (Weiter).

f. Geben Sie für Stack name (Stack-Name) einen Namen für Ihre Rolle ein, wie z. B. `eksConnectorAgentRole`. Klicken Sie dann auf Next (Weiter).

g. Wählen Sie auf der Seite Configure stack options (Stack-Optionen konfigurieren) Next (Weiter) aus.

h. Überprüfen Sie auf der Seite Überprüfen Ihre Informationen, bestätigen Sie, dass der Stack IAM-Ressourcen erstellen kann, und wählen Sie dann Erstellen aus.

AWS verwaltete Richtlinien für Amazon Elastic Kubernetes Service

Eine AWS verwaltete Richtlinie ist eine eigenständige Richtlinie, die von erstellt und verwaltet wird AWS. AWS Verwaltete Richtlinien dienen dazu, Berechtigungen für viele gängige Anwendungsfälle bereitzustellen, sodass Sie damit beginnen können, Benutzern, Gruppen und Rollen Berechtigungen zuzuweisen.

Beachten Sie, dass AWS verwaltete Richtlinien für Ihre speziellen Anwendungsfälle möglicherweise keine Berechtigungen mit den geringsten Rechten gewähren, da sie allen AWS Kunden zur Verfügung stehen. Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind.

Sie können die in AWS verwalteten Richtlinien definierten Berechtigungen nicht ändern. Wenn die in einer AWS verwalteten Richtlinie definierten Berechtigungen AWS aktualisiert werden, wirkt sich das Update auf alle Prinzidentitäten (Benutzer, Gruppen und Rollen) aus, denen die Richtlinie zugeordnet ist. AWS aktualisiert eine AWS verwaltete Richtlinie höchstwahrscheinlich, wenn ein neuer AWS Dienst gestartet wird oder neue API-Operationen für bestehende Dienste verfügbar werden.

Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien im IAM-Benutzerhandbuch.

AWS verwaltete Richtlinie: Amazoneks_CNI_Policy

Sie können die `AmazonEKS_CNI_Policy` an Ihre IAM-Entitäten anhängen. Bevor Sie eine EC2 Amazon-Knotengruppe erstellen, muss diese Richtlinie entweder an die Knoten-IAM-Rolle oder an eine IAM-Rolle angehängt werden, die speziell vom Amazon VPC CNI-Plugin für Kubernetes verwendet wird. Dies dient dazu, Aktionen in Ihrem Namen auszuführen. Wir empfehlen, dass Sie die Richtlinie einer Rolle anfügen, die nur vom Plugin verwendet wird. Weitere Informationen erhalten Sie unter the section called “Amazon VPC CNI” und the section called “Konfiguration für IRSA”.

Details zu Berechtigungen

Diese Richtlinie enthält die folgenden Berechtigungen, die es Amazon EKS ermöglichen, die folgenden Aufgaben auszuführen:

- **`ec2:*NetworkInterface`** und **`ec2:*PrivateIpAddresses`** – Ermöglicht dem Amazon-VPC-CNI-Plugin, Aktionen wie die Bereitstellung von Elastic Network Interfaces und IP-Adressen für Pods auszuführen, um Netzwerke für Anwendungen bereitzustellen, die in Amazon EKS ausgeführt werden.
- **`ec2`**-Leseaktionen – Ermöglicht dem Amazon-VPC-CNI-Plugin, Aktionen wie das Beschreiben von Instances und Subnetzen auszuführen, um die Anzahl der freien IP-Adressen in Ihren

Amazon-VPC-Subnetzen anzuzeigen. Das VPC CNI kann die freien IP-Adressen in jedem Subnetz verwenden, um die Subnetze mit den meisten freien IP-Adressen für die Erstellung einer elastischen Netzwerkschnittstelle auszuwählen.

Die neueste Version des JSON-Richtliniendokuments finden Sie unter AmazonEKS_CNI_Policy im Managed Policy Reference Guide. AWS

AWS verwaltete Richtlinie: EKSCluster Amazon-Richtlinie

Sie können `AmazonEKSClusterPolicy` an Ihre IAM-Entitäten anhängen. Bevor Sie einen Cluster erstellen, müssen Sie über eine Cluster-IAM-Rolle mit dieser angehängten Richtlinie verfügen. Kubernetes-Cluster, die von Amazon EKS verwaltet werden, rufen in Ihrem Namen andere AWS Services auf. Sie tun dies, um die Ressourcen zu verwalten, die Sie mit dem Service verwenden.

Diese Richtlinie enthält die folgenden Berechtigungen, die es Amazon EKS ermöglichen, die folgenden Aufgaben auszuführen:

- **`autoscaling`** – Lesen und aktualisieren Sie die Konfiguration einer Auto-Scaling-Gruppe. Diese Berechtigungen werden von Amazon EKS nicht verwendet, verbleiben jedoch aus Gründen der Abwärtskompatibilität in der Richtlinie.
- **`ec2`**— Arbeiten Sie mit Volumes und Netzwerkressourcen, die EC2 Amazon-Knoten zugeordnet sind. Dies ist erforderlich, damit die Kubernetes-Steuerebene Instances zu einem Cluster verbinden und von persistenten Kubernetes-Volumes angeforderte Amazon-EBS-Volumes dynamisch bereitstellen und verwalten kann.
- **`ec2`** – Löschen Sie elastische Netzwerkschnittstellen, die vom VPC CNI erstellt wurden. Dies ist erforderlich, damit EKS elastische Netzwerkschnittstellen bereinigen kann, die zurückbleiben, wenn das VPC CNI unerwartet beendet wird.
- **`elasticloadbalancing`** – Arbeiten Sie mit Elastic Load Balancern und fügen Sie ihnen Knoten als Ziele hinzu. Dies ist erforderlich, damit die Kubernetes-Steuerebene von Kubernetes-Diensten angeforderte Elastic Load Balancer dynamisch bereitstellen kann.
- **`iam`** – Erstellen Sie eine serviceverknüpfte Rolle. Dies ist erforderlich, damit die Kubernetes-Steuerebene von Kubernetes-Services angeforderte Elastic Load Balancer dynamisch bereitstellen kann.
- **`kms`**— Liest einen Schlüssel von AWS KMS. Dies ist erforderlich, damit die Kubernetes-Steuerebene die Secrets-Verschlüsselung von Kubernetes-Secrets unterstützt, die in `etcd` gespeichert sind.

Die neueste Version des JSON-Richtliniendokuments finden Sie unter Amazon EKSCluster Policy im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: Amazon EKSDashboard ConsoleReadOnly

Sie können `AmazonEKSDashboardConsoleReadOnly` an Ihre IAM-Entitäten anhängen.

Diese Richtlinie enthält die folgenden Berechtigungen, die es Amazon EKS ermöglichen, die folgenden Aufgaben auszuführen:

- **`eks`** – Nur-Lese-Zugriff auf EKS-Dashboard-Daten, Ressourcen und Informationen zu Cluster-Versionen. Dies ermöglicht die Anzeige von EKS-bezogenen Metriken und Details zur Cluster-Konfiguration.
- **`organizations`**- Schreibgeschützter Zugriff auf Unternehmensinformationen AWS , einschließlich:
  - Anzeigen von Organisationsdetails und Servicezugriff
  - Auflisten von Organisations-Roots, Konten und Organisationseinheiten
  - Anzeigen der Organisationsstruktur

Die neueste Version des JSON-Richtliniendokuments finden Sie bei Amazon EKSDashboard ConsoleReadOnly im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: Amazon EKSFargate PodExecutionRolePolicy

Sie können `AmazonEKSFargatePodExecutionRolePolicy` an Ihre IAM-Entitäten anhängen. Bevor Sie ein Fargate-Profil erstellen können, müssen Sie eine Fargate-Pod-Ausführungsrolle erstellen und diese Richtlinie an diese anfügen. Weitere Informationen erhalten Sie unter the section called “Schritt 2: Fargate-Pod-Ausführungsrolle erstellen” und the section called “Profile definieren”.

Diese Richtlinie gewährt der Rolle die Berechtigungen, die den Zugriff auf andere AWS Serviceressourcen ermöglichen, die für die Ausführung von Amazon EKS-Pods auf Fargate erforderlich sind.

Details zu Berechtigungen

Diese Richtlinie enthält die folgenden Berechtigungen, die es Amazon EKS ermöglichen, die folgenden Aufgaben auszuführen:

- **`ecr`** – Ermöglicht Pods, die auf Fargate ausgeführt werden, Container-Images abzurufen, die in Amazon ECR gespeichert sind.

Die neueste Version des JSON-Richtliniendokuments finden Sie bei Amazon EKSFargate PodExecutionRolePolicy im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: Amazon EKSConnector ServiceRolePolicy

Sie können `AmazonEKSConnectorServiceRolePolicy` nicht an Ihre IAM-Entitäten anfügen. Diese Richtlinie ist mit einer servicegebundenen Rolle verknüpft, die es Amazon EKS ermöglicht, Aktionen in Ihrem Namen durchzuführen. Weitere Informationen finden Sie unter the section called “Rolle des Cluster-Konnektors”.

Diese Rolle ermöglicht es Amazon EKS, Kubernetes-Cluster zu verbinden. Die angefügten Richtlinien ermöglichen es der Rolle, die erforderlichen Ressourcen für die Verbindung mit Ihrem registrierten Kubernetes-Cluster zu verwalten.

Details zu Berechtigungen

Diese Richtlinie enthält die folgenden Berechtigungen, die es Amazon EKS ermöglichen, die folgenden Aufgaben auszuführen.

- **`SSM Management`**— SSM-Aktivierungen erstellen, beschreiben und löschen und verwaltete Instances deregistrieren. Dies ermöglicht grundlegende Systems Manager Manager-Operationen.
- **`Session Management`**— Starten Sie SSM-Sitzungen speziell für EKS-Cluster und führen Sie nicht interaktive Befehle mithilfe des AmazonEKS-Dokuments aus.
- **`IAM Role Passing`**— Übergeben Sie IAM-Rollen speziell an den SSM-Dienst, gesteuert durch eine Bedingung, die die übergebenen Rollen auf beschränkt. `ssm.amazonaws.com`
- **`EventBridge Rules`**— Erstellen Sie EventBridge Regeln und Ziele, aber nur, wenn diese von verwaltet werden. `eks-connector.amazonaws.com` Regeln sind speziell auf AWS SSM als Ereignisquelle beschränkt.

Die neueste Version des JSON-Richtliniendokuments finden Sie bei Amazon EKSConnector ServiceRolePolicy im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: Amazon EKSFor FargateServiceRolePolicy

Sie können `AmazonEKSForFargateServiceRolePolicy` nicht an Ihre IAM-Entitäten anfügen. Diese Richtlinie ist mit einer servicegebundenen Rolle verknüpft, die es Amazon EKS ermöglicht, Aktionen in Ihrem Namen durchzuführen. Weitere Informationen finden Sie unter `AWSServiceRoleforAmazonEKSForFargate`.

Diese Richtlinie erteilt Amazon EKS die erforderlichen Berechtigungen zum Ausführen von Fargate-Aufgaben. Die Richtlinie wird nur verwendet, wenn Sie über Fargate-Knoten verfügen.

Details zu Berechtigungen

Diese Richtlinie enthält die folgenden Berechtigungen, die es Amazon EKS ermöglichen, die folgenden Aufgaben auszuführen.

- **`ec2`** – Erstellen und löschen Sie Elastic Network Interfaces und beschreiben Sie Elastic Network Interfaces und Ressourcen. Dies ist erforderlich, damit der Amazon-EKS-Fargate-Service das für Fargate-Pods erforderliche VPC-Netzwerk konfigurieren kann.

Die neueste Version des JSON-Richtliniendokuments finden Sie bei Amazon EKSFor FargateServiceRolePolicy im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: EKSCompute Amazon-Richtlinie

Sie können `AmazonEKSComputePolicy` an Ihre IAM-Entitäten anhängen. Sie können diese Richtlinie an Ihre Cluster-IAM-Rolle anfügen, um die Ressourcen zu erweitern, die EKS in Ihrem Konto verwalten kann.

Diese Richtlinie gewährt die Berechtigungen, die Amazon EKS benötigt, um EC2 Instances für den EKS-Cluster zu erstellen und zu verwalten, sowie die erforderlichen IAM-Berechtigungen für die Konfiguration EC2. Außerdem gewährt diese Richtlinie Amazon EKS die Erlaubnis, die mit dem EC2 Spot-Dienst verknüpfte Rolle in Ihrem Namen zu erstellen.

Details zu Berechtigungen

Diese Richtlinie enthält die folgenden Berechtigungen, die es Amazon EKS ermöglichen, die folgenden Aufgaben auszuführen:

- **`ec2`**-Berechtigungen:

- `ec2:CreateFleet`und `ec2:RunInstances` — Ermöglicht die Erstellung von EC2 Instances und die Verwendung bestimmter EC2 Ressourcen (Bilder, Sicherheitsgruppen, Subnetze) für EKS-Clusterknoten.
- `ec2:CreateLaunchTemplate`- Ermöglicht das Erstellen von EC2 Startvorlagen für EKS-Clusterknoten.
- Die Richtlinie enthält auch Bedingungen, um die Verwendung dieser EC2 Berechtigungen auf Ressourcen zu beschränken, die mit dem EKS-Clusternamen und anderen relevanten Tags gekennzeichnet sind.
- `ec2:CreateTags`- Ermöglicht das Hinzufügen von Tags zu EC2 Ressourcen`CreateFleet`, die durch die `CreateLaunchTemplate` Aktionen`RunInstances`, und erstellt wurden.

- **`iam`**-Berechtigungen:
  - `iam:AddRoleToInstanceProfile` – Ermöglicht das Hinzufügen einer IAM-Rolle zum EKS-Rechen-Instance-Profil.
  - `iam:PassRole`- Ermöglicht die Übergabe der erforderlichen IAM-Rollen an den EC2 Dienst.

Die neueste Version des JSON-Richtliniendokuments finden Sie unter Amazon EKSCompute Policy im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: EKSNetworking Amazon-Richtlinie

Sie können `AmazonEKSNetworkingPolicy` an Ihre IAM-Entitäten anhängen. Sie können diese Richtlinie an Ihre Cluster-IAM-Rolle anfügen, um die Ressourcen zu erweitern, die EKS in Ihrem Konto verwalten kann.

Diese Richtlinie dient dazu, Amazon EKS die erforderlichen Berechtigungen zum Erstellen und Verwalten von Netzwerkschnittstellen für den EKS-Cluster zu gewähren, sodass die Steuerebene und die Worker-Knoten ordnungsgemäß kommunizieren und funktionieren können.

Details zu Berechtigungen

Diese Richtlinie gewährt die folgenden Berechtigungen, damit Amazon EKS Netzwerkschnittstellen für den Cluster verwalten kann:

- **`ec2`**-Netzwerkschnittstellen-Berechtigungen:
  - `ec2:CreateNetworkInterface`- Ermöglicht das Erstellen von EC2 Netzwerkschnittstellen.

  - Die Richtlinie enthält Bedingungen, um die Verwendung dieser Berechtigung auf Netzwerkschnittstellen zu beschränken, die mit dem EKS-Cluster-Namen und dem Kubernetes CNI-Knotennamen gekennzeichnet sind.
  - `ec2:CreateTags` – Ermöglicht das Hinzufügen von Tags zu den durch die `CreateNetworkInterface`-Aktion erstellten Netzwerkschnittstellen.
- Berechtigungen zur **`ec2`**-Netzwerkschnittstellenverwaltung::
  - `ec2:AttachNetworkInterface`, `ec2:DetachNetworkInterface` - Ermöglicht das Anhängen und Trennen von Netzwerkschnittstellen an EC2 Instanzen.
  - `ec2:UnassignPrivateIpAddresses`, `ec2:UnassignIpv6Addresses`, `ec2:AssignPrivateIpAddresses`, `ec2:AssignIpv6Addresses` – Ermöglicht die Verwaltung der IP-Adresszuweisungen der Netzwerkschnittstellen.
  - Diese Berechtigungen sind auf Netzwerkschnittstellen beschränkt, die mit dem EKS-Cluster-Namen gekennzeichnet sind.

Die neueste Version des JSON-Richtliniendokuments finden Sie unter Amazon EKSNetworking Policy im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: Amazon EKSBlock StoragePolicy

Sie können `AmazonEKSBlockStoragePolicy` an Ihre IAM-Entitäten anhängen. Sie können diese Richtlinie an Ihre Cluster-IAM-Rolle anfügen, um die Ressourcen zu erweitern, die EKS in Ihrem Konto verwalten kann.

Diese Richtlinie gewährt Amazon EKS die erforderlichen Berechtigungen, um EC2 Volumes und Snapshots für den EKS-Cluster zu erstellen, zu verwalten und zu verwalten, sodass die Steuerungsebene und die Worker-Knoten persistenten Speicher bereitstellen und verwenden können, wie es für Kubernetes-Workloads erforderlich ist.

Details zu Berechtigungen

Diese IAM-Richtlinie gewährt die folgenden Berechtigungen, damit Amazon EKS EC2 Volumes und Snapshots verwalten kann:

- Berechtigung für die Verwaltung von **`ec2`**-Volumes:
  - `ec2:AttachVolume`,, `ec2:DetachVolumeec2:ModifyVolume`, `ec2:EnableFastSnapshotRestores` — Ermöglicht das Anhängen, Trennen, Ändern und Aktivieren schneller Snapshot-Wiederherstellungen für Volumes. EC2

  - Diese Berechtigungen sind auf Volumes beschränkt, die mit dem EKS-Cluster-Namen gekennzeichnet sind.
  - `ec2:CreateTags`- Ermöglicht das Hinzufügen von Tags zu den EC2 Volumes und Snapshots, die mit den `CreateVolume` Aktionen und erstellt wurden. `CreateSnapshot`
- Berechtigungen zur Erstellung von **`ec2`**-Volumes:
  - `ec2:CreateVolume`- Ermöglicht das Erstellen neuer EC2 Volumes.
  - Die Richtlinie enthält Bedingungen, welche die Verwendung dieser Berechtigung auf Volumes beschränken, die mit dem EKS-Cluster-Namen und anderen relevanten Tags gekennzeichnet sind.
  - `ec2:CreateSnapshot`- Ermöglicht das Erstellen neuer EC2 Volume-Snapshots.
  - Die Richtlinie enthält Bedingungen, welche die Verwendung dieser Berechtigung auf Snapshots beschränken, die mit dem EKS-Cluster-Namen und anderen relevanten Tags gekennzeichnet sind.

Die neueste Version des JSON-Richtliniendokuments finden Sie bei Amazon EKSBlock StoragePolicy im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: Amazon EKSLoad BalancingPolicy

Sie können `AmazonEKSLoadBalancingPolicy` an Ihre IAM-Entitäten anhängen. Sie können diese Richtlinie an Ihre Cluster-IAM-Rolle anfügen, um die Ressourcen zu erweitern, die EKS in Ihrem Konto verwalten kann.

Diese IAM-Richtlinie gewährt Amazon EKS die erforderlichen Berechtigungen für die Zusammenarbeit mit verschiedenen AWS Diensten zur Verwaltung von Elastic Load Balancers (ELBs) und verwandten Ressourcen.

Details zu Berechtigungen

Die wichtigsten durch diese Richtlinie gewährten Berechtigungen sind:

- **`elasticloadbalancing`** : Ermöglicht das Erstellen, Ändern und Verwalten von Elastic Load Balancern und Zielgruppen. Dies umfasst Berechtigungen zum Erstellen, Aktualisieren und Löschen von Load Balancern, Zielgruppen, Listenern und Regeln.
- **`ec2`** : Ermöglicht das Erstellen und Verwalten von Sicherheitsgruppen, die für die Kubernetes-Steuerebene erforderlich sind, um Instances einem Cluster hinzuzufügen und Amazon-EBS-

Volumes zu verwalten. Ermöglicht auch die Beschreibung und Auflistung von EC2 Ressourcen wie Instances VPCs, Subnetzen, Sicherheitsgruppen und anderen Netzwerkressourcen.

- **`iam`**: Ermöglicht die Erstellung einer serviceverknüpften Rolle für Elastic Load Balancing, die für die dynamische Bereitstellung durch die Kubernetes-Steuerebene erforderlich ist. ELBs
- **`kms`**: Ermöglicht das Lesen eines Schlüssels aus AWS KMS, der erforderlich ist, damit die Kubernetes-Steuerebene die Verschlüsselung von in etcd gespeicherten Kubernetes-Geheimnissen unterstützt.
- **`wafv2`**und **`shield`**: Ermöglicht das Zuordnen und Trennen von ACLs Webanwendungen und das Erstellen/Löschen von AWS Shield-Schutzmaßnahmen für die Elastic Load Balancer.
- **`cognito-idp`** , **`acm`** und **`elasticloadbalancing`** : Gewährt Berechtigungen zum Beschreiben von Benutzerpool-Clients, zum Auflisten und Beschreiben von Zertifikaten sowie zum Beschreiben von Zielgruppen, die für die Kubernetes-Steuerebene zur Verwaltung der Elastic Load Balancers erforderlich sind.

Die Richtlinie umfasst auch mehrere Bedingungsprüfungen, um sicherzustellen, dass die Berechtigungen mithilfe des Tags `eks:eks-cluster-name` auf den spezifischen verwalteten EKS-Cluster beschränkt sind.

Die neueste Version des JSON-Richtliniendokuments finden Sie bei Amazon EKSLoad BalancingPolicy im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: Amazon EKSMCPRead OnlyAccess

Sie können `AmazonEKSMCPReadOnlyAccess` an Ihre IAM-Entitäten anhängen. Diese Richtlinie bietet Lesezugriff auf Amazon EKS-Ressourcen und zugehörige AWS Dienste, sodass der Amazon EKS Model Context Protocol (MCP) -Server Beobachtungs- und Fehlerbehebungsvorgänge durchführen kann, ohne Änderungen an Ihrer Infrastruktur vorzunehmen.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen, mit denen Principals die folgenden Aufgaben ausführen können:

- **`eks`**Ermöglicht Prinzipalen, EKS-Cluster, Knotengruppen, Add-Ons, Zugriffseinträge und Einblicke zu beschreiben und aufzulisten und auf die Kubernetes-API für schreibgeschützte Operationen zuzugreifen.
- **`iam`**Ermöglicht Prinzipalen das Abrufen von Informationen über IAM-Rollen, -Richtlinien und deren Anlagen, um die mit EKS-Ressourcen verknüpften Berechtigungen zu verstehen.

- **`ec2`**Ermöglicht es Prinzipalen VPCs, Subnetze und Routing-Tabellen zu beschreiben, um die Netzwerkkonfiguration von EKS-Clustern zu verstehen.
- **`sts`**Ermöglicht Prinzipalen das Abrufen von Anruferidentitätsinformationen zu Authentifizierungs- und Autorisierungszwecken.
- **`logs`**Ermöglicht es den Prinzipalen, Abfragen zu starten und Abfrageergebnisse zur Problembehandlung und Überwachung aus CloudWatch Protokollen abzurufen.
- **`cloudwatch`**Ermöglicht Prinzipalen das Abrufen von Metrikdaten zur Überwachung der Cluster- und Workload-Leistung.
- **`eks-mcp`**Ermöglicht Principals, MCP-Operationen aufzurufen und schreibgeschützte Tools innerhalb des Amazon EKS MCP-Servers aufzurufen.

Die neueste Version des JSON-Richtliniendokuments finden Sie bei Amazon EKSMCPRead OnlyAccess im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: EKSService Amazon-Richtlinie

Sie können `AmazonEKSServicePolicy` an Ihre IAM-Entitäten anhängen. Für Cluster, die vor dem 16. April 2020 erstellt wurden, mussten Sie eine IAM-Rolle erstellen und diese Richtlinie anhängen. Cluster, die am oder nach dem 16. April 2020 erstellt wurden, erfordern weder die Erstellung einer Rolle noch die Zuweisung dieser Richtlinie. Wenn Sie einen Cluster mithilfe eines IAM-Prinzipals erstellen, der über die `iam:CreateServiceLinkedRole` entsprechende Berechtigung verfügt, wird die mit dem Dienst verknüpfte AWSServiceRoleforAmazonEKS-Rolle automatisch für Sie erstellt. Der serviceverknüpften Rolle ist die verwaltete Richtlinie: Amazon EKSService RolePolicy zugeordnet.

Diese Richtlinie ermöglicht Amazon EKS, die erforderlichen Ressourcen für den Betrieb von Amazon-EKS-Clustern zu erstellen und zu verwalten.

Details zu Berechtigungen

Diese Richtlinie enthält die folgenden Berechtigungen, die es Amazon EKS ermöglichen, die folgenden Aufgaben auszuführen.

- **`eks`** – Aktualisieren Sie die Kubernetes-Version Ihres Clusters, nachdem Sie ein Update initiiert haben. Diese Berechtigung wird von Amazon EKS nicht verwendet, verbleibt jedoch aus Gründen der Abwärtskompatibilität in der Richtlinie.
- **`ec2`** – Arbeiten Sie mit Elastic Network Interfaces und anderen Netzwerkressourcen und Tags. Dies wird von Amazon EKS benötigt, um ein Netzwerk zu konfigurieren, das die Kommunikation

zwischen Knoten und der Kubernetes-Steuerebene erleichtert. Lesen Sie die Informationen zu Sicherheitsgruppen. Aktualisieren Sie Tags für Sicherheitsgruppen.

- **`route53`** – Verknüpfen Sie eine VPC mit einer gehosteten Zone. Dies wird von Amazon EKS benötigt, um ein privates Endpunktnetzwerk für Ihren Kubernetes-Cluster-API-Server zu aktivieren.
- **`logs`** – Protokollereignisse Dies ist erforderlich, damit Amazon EKS die Protokolle der Kubernetes-Steuerebene an senden kann. CloudWatch
- **`iam`** – Erstellen Sie eine serviceverknüpfte Rolle. Dies ist erforderlich, damit Amazon EKS die serviceverknüpfte the section called “Serviceverknüpfte Rollenberechtigungen für Amazon EKS”-Rolle in Ihrem Namen erstellt.

Die neueste Version des JSON-Richtliniendokuments finden Sie unter Amazon EKSService Policy im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: Amazon EKSService RolePolicy

Sie können `AmazonEKSServiceRolePolicy` nicht an Ihre IAM-Entitäten anfügen. Diese Richtlinie ist mit einer servicegebundenen Rolle verknüpft, die es Amazon EKS ermöglicht, Aktionen in Ihrem Namen durchzuführen. Weitere Informationen finden Sie unter the section called “Serviceverknüpfte Rollenberechtigungen für Amazon EKS”. Wenn Sie einen Cluster mithilfe eines IAM-Prinzipals erstellen, der über die `iam:CreateServiceLinkedRole` entsprechende Berechtigung verfügt, wird die mit dem Dienst verknüpfte AWSServiceRoleforAmazonEKS-Rolle automatisch für Sie erstellt, und diese Richtlinie wird ihr angehängt.

Diese Richtlinie ermöglicht es der serviceverknüpften Rolle, AWS Dienste in Ihrem Namen aufzurufen.

Details zu Berechtigungen

Diese Richtlinie enthält die folgenden Berechtigungen, die es Amazon EKS ermöglichen, die folgenden Aufgaben auszuführen.

- **`ec2`**— Erstellen und beschreiben Sie Elastic Network Interfaces und EC2 Amazon-Instances, die Cluster-Sicherheitsgruppe und die VPC, die für die Erstellung eines Clusters erforderlich sind. Weitere Informationen finden Sie unter the section called “Anforderungen an Sicherheitsgruppen”. Lesen Sie die Informationen zu Sicherheitsgruppen. Aktualisieren Sie Tags für Sicherheitsgruppen. Weitere Informationen zu On-Demand-Kapazitätsreservierungen. Lesen Sie die VPC-Konfiguration einschließlich Routentabellen und Netzwerk ACLs , um Konfigurationsprobleme im Rahmen von Cluster-Insights zu erkennen.

- **`ec2`**Automatischer Modus — Beendet EC2 Instances, die im automatischen Modus von EKS erstellt wurden. Weitere Informationen finden Sie unter EKS Auto Mode.
- **`iam`** – Listen Sie alle verwalteten Richtlinien auf, die einer IAM-Rolle zugeordnet sind. Dies ist erforderlich, damit Amazon EKS alle verwalteten Richtlinien und Berechtigungen auflisten und validieren kann, die zum Erstellen von Clustern erforderlich sind.
- Eine VPC mit einer gehosteten Zone verknüpfen – Dies wird von Amazon EKS benötigt, um ein privates Endpunktnetzwerk für Ihren Kubernetes-Cluster-API-Server zu aktivieren.
- Ereignis protokollieren — Dies ist erforderlich, damit Amazon EKS die Protokolle der Kubernetes-Steuerebene an senden kann. CloudWatch
- Metrik angeben — Dies ist erforderlich, damit Amazon EKS die Protokolle der Kubernetes-Steuerebene an senden kann. CloudWatch
- **`eks`** – Verwalten Sie Cluster-Zugriffseinträge und -Richtlinien, um detailliert zu steuern, wer auf EKS-Ressourcen zugreifen und welche Aktionen ausführen darf. Dazu gehört auch die Zuordnung von Standardzugriffsrichtlinien für Rechen-, Netzwerk-, Load Balancing- und Speichervorgänge.
- **`elasticloadbalancing`** – Erstellen, verwalten und löschen Sie Load Balancer und deren Komponenten (Listener, Zielgruppen, Zertifikate), die EKS-Clustern zugeordnet sind. Zeigen Sie die Attribute und den Integritätsstatus des Load Balancers an.
- **`events`**- Erstellen und verwalten Sie EventBridge Regeln für Überwachungs EC2 - und AWS Integritätsereignisse im Zusammenhang mit EKS-Clustern und ermöglichen Sie so automatisierte Reaktionen auf Infrastrukturänderungen und Integritätswarnungen.
- **`iam`**- Verwalten Sie EC2 Instanzprofile mit dem Präfix „eks“, einschließlich Erstellung, Löschung und Rollenzuweisung, was für die EKS-Knotenverwaltung erforderlich ist.
- **`pricingshield`**- Greifen Sie auf AWS Preisinformationen und den Shield-Schutzstatus zu, was Kostenmanagement und erweiterte Sicherheitsfunktionen für EKS-Ressourcen ermöglicht.
- Ressourcen-Bereinigung – Löschen Sie während Cluster-Bereinigungsvorgängen sicher EKS-gekennzeichnete Ressourcen, einschließlich Volumes, Snapshots, Startvorlagen und Netzwerkschnittstellen.

Die neueste Version des JSON-Richtliniendokuments finden Sie bei Amazon EKSService RolePolicy im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: Amazon EKSVPCResource Controller

Sie können die `AmazonEKSVPCResourceController`-Richtlinie an Ihre IAM-Identitäten anfügen. Wenn Sie Sicherheitsgruppen für Pods verwenden, müssen Sie diese Richtlinie an Ihre Amazon-EKS-Cluster-IAM-Rolle anfügen, um Aktionen in Ihrem Namen auszuführen.

Diese Richtlinie gewährt der Clusterrolle Berechtigungen zum Verwalten von Elastic Network Interfaces und IP-Adressen für Knoten.

Details zu Berechtigungen

Diese Richtlinie enthält die folgenden Berechtigungen, die es Amazon EKS ermöglichen, die folgenden Aufgaben auszuführen:

- **`ec2`** – Verwalten Sie Elastic Network Interfaces und IP-Adressen, um Pod-Sicherheitsgruppen und Windows-Knoten zu unterstützen.

Die neueste Version des JSON-Richtliniendokuments finden Sie unter Amazon EKSVPCResource Controller im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: Amazon EKSWorker NodePolicy

Sie können die `AmazonEKSWorkerNodePolicy` an Ihre IAM-Entitäten anhängen. Sie müssen diese Richtlinie an eine Node-IAM-Rolle anhängen, die Sie angeben, wenn Sie EC2 Amazon-Knoten erstellen, die es Amazon EKS ermöglichen, Aktionen in Ihrem Namen durchzuführen. Wenn Sie eine Knotengruppe mit `eksctl` erstellen, wird die Knoten-IAM-Rolle erstellt und diese Richtlinie automatisch an die Rolle angehängt.

Diese Richtlinie gewährt Amazon EKS EC2 Amazon-Knoten die Erlaubnis, sich mit Amazon EKS-Clustern zu verbinden.

Details zu Berechtigungen

Diese Richtlinie enthält die folgenden Berechtigungen, die es Amazon EKS ermöglichen, die folgenden Aufgaben auszuführen:

- **`ec2`** – Lesen Sie Informationen zum Instance-Volumen und zum Netzwerk. Dies ist erforderlich, damit Kubernetes-Knoten Informationen über EC2 Amazon-Ressourcen beschreiben können, die erforderlich sind, damit der Knoten dem Amazon EKS-Cluster beitritt.

- **`eks`** – Beschreiben Sie optional den Cluster als Teil des Knoten-Bootstrappings.
- **`eks-auth:AssumeRoleForPodIdentity`** – Erlauben Sie das Abrufen von Anmeldeinformationen für EKS-Workloads auf dem Knoten. Dies ist erforderlich, damit EKS Pod Identity ordnungsgemäß funktioniert.

Die neueste Version des JSON-Richtliniendokuments finden Sie bei Amazon EKSWorker NodePolicy im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: Amazon EKSWorker NodeMinimalPolicy

Sie können Amazon EKSWorker NodeMinimalPolicy an Ihre IAM-Entitäten anhängen. Sie können diese Richtlinie an eine Node-IAM-Rolle anhängen, die Sie angeben, wenn Sie EC2 Amazon-Knoten erstellen, die es Amazon EKS ermöglichen, Aktionen in Ihrem Namen durchzuführen.

Diese Richtlinie gewährt Amazon EKS EC2 Amazon-Knoten die Erlaubnis, sich mit Amazon EKS-Clustern zu verbinden. Diese Richtlinie hat im Vergleich zu Amazon weniger Berechtigungen EKSWorkerNodePolicy.

Details zu Berechtigungen

Diese Richtlinie enthält die folgenden Berechtigungen, die es Amazon EKS ermöglichen, die folgenden Aufgaben auszuführen:

- `eks-auth:AssumeRoleForPodIdentity` – Erlauben Sie das Abrufen von Anmeldeinformationen für EKS-Workloads auf dem Knoten. Dies ist erforderlich, damit EKS Pod Identity ordnungsgemäß funktioniert.

Die neueste Version des JSON-Richtliniendokuments finden Sie bei Amazon EKSWorker NodePolicy im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: AWSService RoleForAmazon EKSNodegroup

Sie können `AWSServiceRoleForAmazonEKSNodegroup` nicht an Ihre IAM-Entitäten anfügen. Diese Richtlinie ist mit einer servicegebundenen Rolle verknüpft, die es Amazon EKS ermöglicht, Aktionen in Ihrem Namen durchzuführen. Weitere Informationen finden Sie unter the section called “Serviceverknüpfte Rollenberechtigungen für Amazon EKS”.

Diese Richtlinie gewährt der `AWSServiceRoleForAmazonEKSNodegroup` Rolle Berechtigungen, die es ihr ermöglichen, EC2 Amazon-Knotengruppen in Ihrem Konto zu erstellen und zu verwalten.

Details zu Berechtigungen

Diese Richtlinie enthält die folgenden Berechtigungen, die es Amazon EKS ermöglichen, die folgenden Aufgaben auszuführen:

- **`ec2`** – Arbeiten Sie mit Sicherheitsgruppen, Tags, Kapazitätsreservierungen und Startvorlagen. Dies ist für von Amazon EKS verwaltete Knotengruppen erforderlich, um die Zugriffskonfiguration auf Distanz zu ermöglichen und Kapazitätsreservierungen zu beschreiben, die in verwalteten Knotengruppen verwendet werden können. Darüber hinaus erstellen von Amazon EKS verwaltete Knotengruppen in Ihrem Namen eine Startvorlage. Dies dient dazu, die Amazon EC2 Auto Scaling Scaling-Gruppe zu konfigurieren, die jede verwaltete Knotengruppe unterstützt.
- **`iam`** – Erstellen einer serviceverknüpften Rolle und Übergeben einer Rolle. Dies ist für von Amazon EKS verwaltete Knotengruppen erforderlich, um Instance-Profile für die Rolle zu verwalten, die beim Erstellen einer verwalteten Knotengruppe übergeben wird. Dieses Instance-Profil wird von EC2 Amazon-Instances verwendet, die als Teil einer verwalteten Knotengruppe gestartet wurden. Amazon EKS muss serviceverknüpfte Rollen für andere Services wie Amazon EC2 Auto Scaling Scaling-Gruppen erstellen. Diese Berechtigungen werden bei der Erstellung einer verwalteten Knotengruppe verwendet.
- **`autoscaling`** – Arbeiten Sie mit Sicherheitsgruppen für Auto Scaling. Dies ist für von Amazon EKS verwaltete Knotengruppen erforderlich, um die Amazon EC2 Auto Scaling Scaling-Gruppe zu verwalten, die jede verwaltete Knotengruppe unterstützt. Es wird auch verwendet, um Funktionen wie das Entfernen von Pods zu unterstützen, wenn Knoten während Aktualisierungen der Knotengruppe beendet oder recycelt werden.

Die neueste Version des JSON-Richtliniendokuments finden Sie AWSServiceRoleForAmazonEKSNodegroupim AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: Amazon EKSDashboard ServiceRolePolicy

Sie können `AmazonEKSDashboardServiceRolePolicy` nicht an Ihre IAM-Entitäten anfügen. Diese Richtlinie ist mit einer servicegebundenen Rolle verknüpft, die es Amazon EKS ermöglicht, Aktionen in Ihrem Namen durchzuführen. Weitere Informationen finden Sie unter the section called “Serviceverknüpfte Rollenberechtigungen für Amazon EKS”.

Diese Richtlinie gewährt der `AWSServiceRoleForAmazonEKSDashboard` Rolle Berechtigungen, die es ihr ermöglichen, EC2 Amazon-Knotengruppen in Ihrem Konto zu erstellen und zu verwalten.

Details zu Berechtigungen

Diese Richtlinie enthält die folgenden Berechtigungen, die den Zugriff zur Ausführung dieser Aufgaben ermöglichen:

- **`organizations`**— Informationen zur Struktur und zu den Konten Ihrer AWS Organizations anzeigen. Dies enthält Berechtigungen zum Auflisten von Konten in Ihrer Organisation, zum Anzeigen von Organisationseinheiten und Stammverzeichnissen, zum Auflisten von delegierten Administratoren, zum Anzeigen von Services, die Zugriff auf Ihre Organisation haben, sowie zum Abrufen detaillierter Informationen über Ihre Organisation und Konten.

Die neueste Version des JSON-Richtliniendokuments finden Sie bei Amazon EKSDashboard ServiceRolePolicy im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: EBSCSIDriver Amazon-Richtlinie

Die `AmazonEBSCSIDriverPolicy` Richtlinie ermöglicht es dem Amazon EBS Container Storage Interface (CSI) -Treiber, Volumes in Ihrem Namen zu erstellen, zu ändern, zu kopieren, anzuhängen, zu trennen und zu löschen. Dazu gehören auch das Ändern von Tags in vorhandenen Volumes und das Aktivieren von Fast Snapshot Restore (FSR) in EBS-Volumes. Sie gewährt dem EBS CSI-Treiber auch die Berechtigung, Snapshots zu erstellen, zu sperren, wiederherzustellen und zu löschen und Ihre Instances, Volumes und Snapshots aufzulisten.

Die neueste Version des JSON-Richtliniendokuments finden Sie bei Amazon EBSCSIDriver ServiceRolePolicy im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: EFSCSIDriver Amazon-Richtlinie

Die `AmazonEFSCSIDriverPolicy`-Richtlinie ermöglicht es dem Amazon EFS Container Storage Interface (CSI), Zugriffspunkte in Ihrem Namen zu erstellen und zu löschen. Es gewährt dem Amazon EFS CSI-Treiber auch Berechtigungen, Ihre Access Points, Dateisysteme, Mount-Ziele und EC2 Amazon-Verfügbarkeitszonen aufzulisten.

Die neueste Version des JSON-Richtliniendokuments finden Sie bei Amazon EFSCSIDriver ServiceRolePolicy im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: Amazon EKSLocal OutpostClusterPolicy

Sie können diese Richtlinie mit IAM-Entitäten verknüpfen. Bevor Sie einen lokalen Cluster erstellen, müssen Sie diese Richtlinie Ihrer Cluster-Rolle anfügen. Kubernetes-Cluster, die von Amazon EKS verwaltet werden, rufen in Ihrem Namen andere AWS Services auf. Sie tun dies, um die Ressourcen zu verwalten, die Sie mit dem Service verwenden.

Die `AmazonEKSLocalOutpostClusterPolicy` umfasst folgende Berechtigungen.

- **`ec2`**-Leseaktionen – Ermöglicht es Steuerebenen-Instances, Eigenschaften von Availability Zones, Routing-Tabellen, Instances und Netzwerkschnittstellen zu beschreiben. Erforderliche Berechtigungen für EC2 Amazon-Instances, um dem Cluster erfolgreich als Kontrollebene-Instances beizutreten.
- **`ssm`**— Ermöglicht Amazon EC2 Systems Manager Manager-Verbindung zur Kontrollebene-Instance, die von Amazon EKS zur Kommunikation und Verwaltung des lokalen Clusters in Ihrem Konto verwendet wird.
- **`logs`**— Ermöglicht Instances, Logs an Amazon zu übertragen CloudWatch.
- **`secretsmanager`**— Ermöglicht Instances das sichere Abrufen und Löschen von Bootstrap-Daten für die Kontrollebeneninstanzen aus AWS Secrets Manager.
- **`ecr`** – Ermöglicht Pods und Containern, die auf Instances auf der Steuerebene ausgeführt werden, das Abrufen von Container-Images, die in Amazon Elastic Container Registry gespeichert sind.

Die neueste Version des JSON-Richtliniendokuments finden Sie bei Amazon EKSLocal OutpostClusterPolicy im AWS Managed Policy Reference Guide.

AWS verwaltete Richtlinie: Amazon EKSLocal OutpostServiceRolePolicy

Sie können diese Richtlinie Ihren IAM-Entitäten nicht anfügen. Wenn Sie einen Cluster mit einem IAM-Prinzipal erstellen, der über die `iam:CreateServiceLinkedRole` entsprechende Berechtigung verfügt, erstellt Amazon EKS automatisch die serviceverknüpfte AWSServiceRoleforAmazonEKSLocalOutpost-Rolle für Sie und fügt ihr diese Richtlinie hinzu. Diese Richtlinie ermöglicht es der serviceverknüpften Rolle, in Ihrem Namen AWS Dienste für lokale Cluster aufzurufen.

Die `AmazonEKSLocalOutpostServiceRolePolicy` umfasst folgende Berechtigungen.

- **`ec2`** – Ermöglicht Amazon EKS die Zusammenarbeit mit Sicherheits-, Netzwerk- und anderen Ressourcen, um Instances der Steuerebene in Ihrem Konto erfolgreich zu starten und zu verwalten.
- **`ssm`**, **`ssmmessages`** — Ermöglicht Amazon EC2 Systems Manager eine Verbindung zu den Instances der Kontrollebene, die von Amazon EKS zur Kommunikation und Verwaltung des lokalen Clusters in Ihrem Konto verwendet wird.
- **`iam`** – Ermöglicht Amazon EKS die Verwaltung des Instance-Profils, das den Instances auf Steuerebene zugeordnet ist.

- **`secretsmanager`**- Ermöglicht Amazon EKS, Bootstrap-Daten für die Instances der Kontrollebene in AWS Secrets Manager zu übertragen, sodass sie beim Instance-Bootstrapping sicher referenziert werden können.
- **`outposts`** – Ermöglicht Amazon EKS, Outpost-Informationen von Ihrem Konto abzurufen, um einen lokalen Cluster in einem Outpost erfolgreich zu starten.

Die neueste Version des JSON-Richtliniendokuments finden Sie bei Amazon EKSLocal OutpostServiceRolePolicy im AWS Managed Policy Reference Guide.

Amazon EKS-Updates für AWS verwaltete Richtlinien

Sehen Sie sich Details zu Aktualisierungen der AWS verwalteten Richtlinien für Amazon EKS an, seit dieser Service begonnen hat, diese Änderungen zu verfolgen.

Um Benachrichtigungen über alle Änderungen an den Quelldateien dieser spezifischen Dokumentationsseite zu erhalten, können Sie die folgende URL mit einem RSS-Reader abonnieren:

```
https://github.com/awsdocs/amazon-eks-user-guide/commits/mainline/latest/ug/security/iam-reference/security-iam-awsmanpol.adoc.atom
```

Änderungen	Beschreibung	Date
Berechtigungen zur EBSCSIDriverAmazon-Richtlinie hinzugefügt.	Es wurde die `ec2:LockSnapshot` Berechtigung hinzugefügt, dem EBS-CSI-Treiber das direkte Sperren von EBS-Snapshots zu ermöglichen.	15. Januar 2026
Einführung von the section called “AWS verwaltete Richtlinie: Amazon EKSMCPRead OnlyAccess”.	Amazon EKS führte eine neue verwaltete Richtlinie ein`AmazonEKSMCPReadOnlyAccess` , um schreibgeschützte Tools auf dem Amazon EKS MCP-Server für Beobachtbarkeit und Fehlerbehebung zu aktivieren.	21. November 2025

Änderungen	Beschreibung	Date
Berechtigungen zur EBSCSIDriverAmazon-Richtlinie hinzugefügt.	Es wurde die `ec2:CopyVolumes` Berechtigung hinzugefügt, dem EBS-CSI-Treiber das direkte Kopieren von EBS-Volumes zu ermöglichen.	17. November 2025
Berechtigung zu the section called “AWS verwaltete Richtlinie: Amazon EKSServiceRolePolicy” hinzugefügt.	Hinzugefügt `ec2:DescribeRouteTables` und `ec2:DescribeNetworkAcls` Berechtigungen für`AmazonEKSServiceRolePolicy` . Auf diese Weise kann Amazon EKS im Rahmen von Cluster-Erkenntnissen Konfigurationsprobleme mit VPC-Routing-Tabellen und Netzwerken ACLs für Hybridknoten erkennen.	22. Oktober 2025
Berechtigung zu AWSServiceRoleForAmazonEKSConnector hinzugefügt	`ssmmessages:OpenDataChannel` Erlaubnis hinzugefügt zu `AmazonEKSConnectorServiceRolePolicy`	15. Oktober 2025
Berechtigung zu the section called “AWS verwaltete Richtlinie: Amazon EKSServiceRolePolicy” hinzugefügt	Diese Rolle kann die neue Zugriffsrichtlinie `AmazonEKSEventPolicy` anfügen. Eingeschränkte Berechtigungen für `ec2:DeleteLaunchTemplate` und `ec2:TerminateInstances` .	26. August 2025

Änderungen	Beschreibung	Date
Berechtigung zu the section called “AWS verwaltete Richtlinie: Amazon EKSLocal OutpostServiceRolePolicy” hinzugefügt	`ssmmessages:OpenDataChannel` -Berechtigung zu `AmazonEKSLocalOutpostServiceRolePolicy` hinzugefügt.	26. Juni 2025
Berechtigung zu the section called “AWS verwaltete Richtlinie: EKSCompute Amazon-Richtlinie” hinzugefügt.	Aktualisierte Ressourcenberechtigungen für die Aktionen `ec2:RunInstances` und `ec2:CreateFleet` , um Kapazitätsreservierungen `arn:aws:ec2:*:*:capacity-reservation/*` einzuschließen. Dadurch kann Amazon EKS Auto Mode Instances mithilfe der EC2 On-Demand-Kapazitätsreservierungen in Ihrem Konto starten. Hinzugefügt`iam:CreateServiceLinkedRole` , damit Amazon EKS Auto Mode die mit dem EC2 Spot-Dienst verknüpfte Rolle in `AWSServiceRoleForEC2Spot` Ihrem Namen erstellen kann.	20. Juni 2025

Änderungen	Beschreibung	Date
Die Erlaubnis wurde Amazon hinzugefügt EKSServic eRolePolicy.	Es wurde die `ec2:Descr ibeCapacityReserva tions` Erlaubnis hinzugefü gt, Amazon EKS Auto Mode zu erlauben, Instances mithilfe der EC2 On-Demand-Kapazitä tsreservierungen in Ihrem Konto zu starten.	20. Juni 2025
Einführung von the section called “AWS verwaltet e Richtlinie: Amazon EKSDashboard ConsoleRe adOnly”.	Neue `AmazonEKS DashboardConsoleRe adOnly` -Richtlinie eingeführt.	19. Juni 2025
Einführung von the section called “AWS verwaltet e Richtlinie: Amazon EKSDashboard ServiceRo lePolicy”.	Neue `AmazonEKS DashboardServiceRo lePolicy` -Richtlinie eingeführt.	21. Mai 2025
Berechtigungen zur EKSClusterAmazon-Richtlinie hinzugefügt.	`ec2:DeleteNetworkI nterfaces` -Berechtigung hinzugefügt, um Amazon EKS das Löschen von Elastic Network Interfaces zu ermöglichen, die zurückble iben, wenn das VPC CNI unerwartet beendet wird.	16. April 2025

Änderungen	Beschreibung	Date
Die Erlaubnis wurde Amazon hinzugefügt EKSServic eRolePolicy.	Es wurden `ec2:Autho rizeSecurityGroupE gress` Berechtigungen hinzugefügt`ec2:Revok eSecurityGroupEgre ss` , die es AI/ML EKS-Kunde n ermöglichen, als Teil der Version EKS 1.33 Sicherhei tsgruppenausgangsregeln zur standardmäßigen EKS-Clust er-SG hinzuzufügen, die mit EFA kompatibel sind.	14. April 2025
Berechtigungen für Amazon hinzugefügt EKSServic eRolePolicy.	Die Berechtigung zum Beenden von EC2 Instances, die mit EKS Auto Mode erstellt wurden, wurde hinzugefügt.	28. Februar 2025

Änderungen	Beschreibung	Date
Berechtigungen zur EBSCSIDriverAmazon-Richtlinie hinzugefügt.	Eine neue Anweisung wurde hinzugefügt, die den EBS CSI-Treiber zur Wiederherstellung aller Snapshots autorisiert. Dies war zuvor durch die vorhandene Richtlinie zulässig, aufgrund einer Änderung in der Handhabung von IAM für `CreateVolume` ist jedoch eine neue explizite Anweisung erforderlich. Dem EBS CSI-Treiber wurde die Möglichkeit hinzugefügt, Tags auf vorhandenen Volumes zu ändern. Der EBS CSI-Treiber kann Tags vorhandener Volumes über Parameter in VolumeAttributesClasses Kubernetes ändern. Dem EBS-CSI-Treiber wurde die Möglichkeit hinzugefügt, Fast Snapshot Restore (FSR) in EBS-Volumes zu aktivieren. Der EBS-CSI-Treiber kann FSR auf neuen Volumes über Parameter in Kubernetes-Speicherklassen aktivieren.	13. Januar 2025

Änderungen	Beschreibung	Date
Berechtigungen zu the section called “AWS verwaltete Richtlinie: Amazon EKSLoad BalancingPolicy” hinzugefügt.	`AmazonEKSLoadBalancingPolicy` wurde aktualisiert, um die Auflistung und Beschreibung von Netzwerk- und IP-Adressressourcen zu ermöglichen.	26. Dezember 2024
Berechtigungen zu the section called “AWS verwaltete Richtlinie: AWSService RoleForAmazon EKSNodegroup” hinzugefügt.	`AWSServiceRoleForAmazonEKSNodegroup` für Kompatibilität mit chinesischen Regionen aktualisiert.	22. November 2024
Berechtigungen zu the section called “AWS verwaltete Richtlinie: Amazon EKSLocal OutpostClusterPolicy” hinzugefügt	Es wurde eine `ec2:DescribeAvailabilityZones` Berechtigung hinzugefügt, `AmazonEKSLocalOutpostClusterPolicy` damit der AWS Cloud Controller Manager auf der Cluster-Steuerebene die Availability Zone identifizieren kann, in der sich jeder Knoten befindet.	21. November 2024

Änderungen	Beschreibung	Date
Berechtigungen zu the section called “AWS verwaltete Richtlinie: AWSServiceRoleForAmazon EKSNodegroup” hinzugefügt.	Die `AWSServiceRoleForAmazonEKSNodegroup` - Richtlinie wurde so aktualisiert, dass Instances, die von Amazon EKS verwalteten Knotengruppen erstellt wurden, `ec2:RebootInstances` zulassen können. Die `ec2:CreateTags` Berechtigungen für EC2 Amazon-Ressourcen wurden eingeschränkt.	20. November 2024
Berechtigungen zu the section called “AWS verwaltete Richtlinie: Amazon EKSServiceRolePolicy” hinzugefügt.	EKS hat die AWS verwaltete Richtlinie aktualisiertAmazonEKSServiceRolePolicy . Berechtigungen für EKS-Zugriffsrichtlinien, Load-Balancer-Verwaltung und automatische Bereinigung von Cluster-Ressourcen hinzugefügt.	16. November 2024
Einführung von the section called “AWS verwaltete Richtlinie: EKSCompute Amazon-Richtlinie”.	EKS hat die AWS verwaltete Richtlinie aktualisiertAmazonEKSComputePolicy . Aktualisierte Ressourcenberechtigungen für die Aktion `iam:AddRoleToInstanceProfile` .	7. November 2024
Einführung von the section called “AWS verwaltete Richtlinie: EKSCompute Amazon-Richtlinie”.	AWS führte das einAmazonEKSComputePolicy .	1. November 2024

Änderungen	Beschreibung	Date
Berechtigungen zu `AmazonEKSClusterPolicy` hinzugefügt	Berechtigung `ec2:DescribeInstanceTopology` hinzugefügt, damit Amazon EKS Topologieinformationen als Beschriftungen an den Knoten anfügen kann.	1. November 2024
Einführung von the section called “AWS verwaltete Richtlinie: Amazon EKSBlockStoragePolicy”.	AWS führte die ein`AmazonEKSBlockStoragePolicy` .	30. Oktober 2024
Einführung von the section called “AWS verwaltete Richtlinie: Amazon EKSLoadBalancingPolicy”.	AWS führte die ein`AmazonEKSLoadBalancingPolicy` .	30. Oktober 2024
Berechtigungen für Amazon hinzugefügt EKSServiceRolePolicy.	Es wurden `cloudwatch:PutMetricData` Berechtigungen hinzugefügt, die es Amazon EKS ermöglichen, Metriken auf Amazon zu veröffentlichen CloudWatch.	29. Oktober 2024
Einführung von the section called “AWS verwaltete Richtlinie: EKSNetworking Amazon-Richtlinie”.	AWS führte das ein`AmazonEKSNetworkingPolicy` .	28. Oktober 2024
Berechtigungen zu `AmazonEKSServicePolicy` und `AmazonEKSServiceRolePolicy` hinzugefügt.	`ec2:GetSecurityGroupsForVpc` und zugehörige Tag-Berechtigungen hinzugefügt, damit EKS Sicherheitsgruppeninformationen lesen und zugehörige Tags aktualisieren kann.	10. Oktober 2024

Änderungen	Beschreibung	Date
Amazon eingeführt EKSWorkerNodeMinimalPolicy.	AWS führte das einAmazonEKSWorkerNodeMinimalPolicy .	3. Oktober 2024
Berechtigungen zu AWSServiceRoleForAmazonEKSNodegroup hinzugefügt.	autoscaling:ResumeProcesses - und autoscaling:SuspendProcesses -Berechtigungen hinzugefügt, damit Amazon EKS AZRebalance in von Amazon EKS verwalteten Auto-Scaling-Gruppen anhalten und fortsetzen kann.	21. August 2024
Berechtigungen zu AWSServiceRoleForAmazonEKSNodegroup hinzugefügt.	ec2:DescribeCapacityReservations -Berechtigung hinzugefügt, um Amazon EKS das Beschreiben der Kapazitätsreservierung im Benutzerkonto zu ermöglichen. autoscaling:PutScheduledUpdateGroupAction -Berechtigung hinzugefügt, um die Einstellung der geplanten Skalierung für CAPACITY_BLOCK - Knotengruppen zu ermöglichen.	27. Juni 2024

Änderungen	Beschreibung	Date
AmazonEKS_CNI_Policy – Aktualisierung einer vorhandenen Richtlinie	Amazon EKS hat neue `ec2:DescribeSubnets` -Berechtigungen hinzugefügt, damit das Amazon-VPC-CNI-Plugin für Kubernetes die Anzahl der freien IP-Adressen in Ihren Amazon-VPC-Subnetzen sehen kann. Das VPC CNI kann die freien IP-Adressen in jedem Subnetz verwenden, um die Subnetze mit den meisten freien IP-Adressen für die Erstellung einer elastischen Netzwerkschnittstelle auszuwählen.	4. März 2024
Amazon EKSWorkerNodePolicy — Aktualisierung einer bestehenden Richtlinie	Amazon EKS wurden neue Berechtigungen hinzugefügt, die EKS-Pod-Identitäten zulassen. Der Amazon EKS Pod Identity-Agent verwendet die Knotenrolle.	26. November 2023
Einführung der EFSCSIDriverAmazon-Richtlinie.	AWS führte die ein`AmazonEFSCSIDriverPolicy` .	26. Juli 2023
Berechtigungen zur EKSClusterAmazon-Richtlinie hinzugefügt.	Die `ec2:DescribeAvailabilityZones` -Berechtigung wurde hinzugefügt, damit Amazon EKS die AZ-Details während der automatischen Subnetzerkennung beim Erstellen von Load Balancern abrufen kann.	07. Februar 2023

Änderungen	Beschreibung	Date
Die Versicherungsbedingungen in den EBSCSIDriverAmazon-Richtlinien wurden aktualisiert.	Ungültige Richtlinienbedingungen mit Platzhalterzeichen im `StringLike` -Schlüsselfeld wurden entfernt. Außerdem wurde eine neue Bedingung `ec2:ResourceTag/kubernetes.io/created-for/pvc/name: "*"` zu `ec2:DeleteVolume` hinzugefügt, die es dem EBS-CSI-Treiber ermöglicht, vom In-Tree-Plugin erstellte Volumes zu löschen.	17. November 2022
Berechtigungen für Amazon hinzugefügt EKSLocalOutpostServiceRolePolicy.	`ec2:DescribeVPCAttribute` , `ec2:GetConsoleOutput` und `ec2:DescribeSecret` wurden hinzugefügt, um eine bessere Validierung der Voraussetzungen und eine bessere Kontrolle des Lebenszyklus zu ermöglichen. Außerdem wurde `ec2:DescribePlacementGroups` und hinzugefügt`"arn:aws:ec2:*:*:placement-group/*"` , `ec2:RunInstances` um die Platzierungskontrolle der EC2 Amazon-Instances auf der Kontrollebene auf Outposts zu unterstützen.	24. Oktober 2022

Änderungen	Beschreibung	Date
Aktualisieren Sie die Amazon Elastic Container Registry-Berechtigungen in Amazon EKSLocal OutpostClusterPolicy.	Die Aktion `ecr:GetDownloadUrlForLayer` wurde von allen Ressourcenabschnitten in einen Bereich mit begrenztem Umfang verschoben. Ressource `arn:aws: ecr:*:*:repository/eks/` wurde hinzugefügt. Entfernen Sie die Ressource `arn:aws: ecr:`. Diese Ressource wird durch die hinzugefügte `arn:aws: ecr:*:*:repository/eks/*` - Ressource abgedeckt.	20. Oktober 2022
Berechtigungen für Amazon hinzugefügt EKSLocalOutpostClusterPolicy.	Das `arn:aws: ecr:*:*:repository/kubelet-config-updater` Repository der Amazon-Elastic-Container-Registry wurde hinzugefügt, damit die Cluster-Steuerebenen-Instances einige `kubelet`-Argumente aktualisieren können.	31. August 2022
Amazon eingeführt EKSLocalOutpostClusterPolicy.	AWS führte das ein`AmazonEKSLocalOutpostClusterPolicy` .	24. August 2022
Amazon eingeführt EKSLocalOutpostServiceRolePolicy.	AWS führte das ein`AmazonEKSLocalOutpostServiceRolePolicy` .	23. August 2022

Änderungen	Beschreibung	Date
Einführung der EBSCSIDriverAmazon-Richtlinie.	AWS führte die einAmazonEBSCSIDriverPolicy .	4. April 2022
Berechtigungen für Amazon hinzugefügt EKSWorker NodePolicy.	Hinzugefügtec2:DescribeInstanceTypes , um Amazon EKS-Optimized zu aktivieren AMIs , das Eigenschaften auf Instance-Ebene auto erkennen kann.	21. März 2022
Berechtigungen zu AWSServiceRoleForAmazonEKSNodegroup hinzugefügt.	autoscaling:EnableMetricsCollection - Berechtigung hinzugefügt, um Amazon EKS die Aktivierung der Kennzahlenerfassung zu ermöglichen.	13. Dezember 2021
Berechtigungen zur EKSClusterAmazon-Richtlinie hinzugefügt.	ec2:DescribeAccountAttributes -, ec2:DescribeAddresses - und ec2:DescribeInternetGateways -Berechtigungen hinzugefügt, damit Amazon EKS eine serviceverknüpfte Rolle für einen Network Load Balancer erstellen kann.	17. Juni 2021
Amazon EKS hat mit der Nachverfolgung von Änderungen begonnen.	Amazon EKS hat damit begonnen, Änderungen für seine AWS verwalteten Richtlinien nachzuverfolgen.	17. Juni 2021

Fehlersuche bei IAM

In diesem Thema werden einige häufige Fehler bei der Verwendung von Amazon EKS mit IAM sowie deren Lösung behandelt.

AccessDeniedException

Wenn Sie `AccessDeniedException` beim Aufrufen eines AWS API-Vorgangs eine Meldung erhalten, verfügen die von Ihnen verwendeten IAM-Prinzipalanmeldedaten nicht über die erforderlichen Berechtigungen, um diesen Aufruf durchzuführen.

```
An error occurred (AccessDeniedException) when calling the DescribeCluster operation:
User: arn:aws: iam::111122223333:user/user_name is not authorized to perform:
eks:DescribeCluster on resource: arn:aws: eks:region:111122223333:cluster/my-cluster
```

In der vorigen Beispielmeldung hat der Benutzer keine Berechtigung zum Aufruf der API-Operation `DescribeCluster` von Amazon EKS. Um einem IAM-Prinzipal Amazon-EKS-Administratorrechte zu erteilen, lesen Sie the section called “Identitätsbasierte Richtlinien”.

Weitere allgemeine Informationen zu IAM finden Sie unterSteuern des Zugriffs mit RichtlinienimIAM User Guideaus.

Sie können keine Knoten auf der Registerkarte „Compute“ oder etwas anderes auf der Registerkarte „Ressourcen“ sehen und Sie erhalten eine Fehlermeldung in AWS-Managementkonsole

Möglicherweise sehen Sie eine Konsolenfehlermeldung, die folgendermaßen lautet: `Your current user or role does not have access to Kubernetes objects on this EKS cluster`. Stellen Sie sicher, dass der IAM-Prizipalbenutzer, AWS-Managementkonsole mit dem Sie den verwenden, über die erforderlichen Berechtigungen verfügt. Weitere Informationen finden Sie unter the section called “Erforderliche Berechtigungen”.

Die aws-auth-`ConfigMap` gewährt keinen Zugriff auf den Cluster

Der AWS IAM Authenticator lässt keinen Pfad in dem Rollen-ARN zu, der in der `ConfigMap` verwendet wird. Entfernen Sie daher den Pfad, bevor Sie `rolearn` angeben. Sie können beispielsweise `arn:aws: iam::111122223333:role/team/developers/eks-admin` in `arn:aws: iam::111122223333:role/eks-admin` ändern.

Ich bin nicht berechtigt, iam auszuführen: PassRole

Wenn Sie die Fehlermeldung erhalten, dass Sie nicht zur Ausführung der `iam:PassRole`-Aktion autorisiert sind, müssen Ihre Richtlinien aktualisiert werden, um eine Rolle an Amazon EKS übergeben zu können.

Bei einigen AWS Diensten können Sie eine bestehende Rolle an diesen Dienst übergeben, anstatt eine neue Servicerolle oder eine dienstverknüpfte Rolle zu erstellen. Hierzu benötigen Sie Berechtigungen für die Übergabe der Rolle an den Service.

Der folgende Beispielfehler tritt auf, wenn ein IAM-Benutzer mit dem Namen `marymajor` versucht, die Konsole zu verwenden, um eine Aktion in Amazon EKS auszuführen. Die Aktion erfordert jedoch, dass der Service über Berechtigungen verfügt, die durch eine Servicerolle gewährt werden. Mary besitzt keine Berechtigungen für die Übergabe der Rolle an den Dienst.

```
User: {arn-aws}iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In diesem Fall müssen die Richtlinien von Mary aktualisiert werden, um die `iam:PassRole`-Aktion ausführen zu können.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich möchte Personen außerhalb meines AWS Kontos den Zugriff auf meine Amazon EKS-Ressourcen ermöglichen

Sie können eine Rolle erstellen, mit der Benutzer in anderen Konten oder Personen außerhalb Ihrer Organisation auf Ihre Ressourcen zugreifen können. Sie können festlegen, wem die Übernahme der Rolle anvertraut wird. Für Dienste, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (ACLs) unterstützen, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen dazu, ob Amazon EKS diese Features unterstützt, finden Sie unter the section called “Amazon EKS und IAM”.
- Informationen dazu, wie Sie den Zugriff auf Ihre Ressourcen mit Ihren AWS Konten gewähren können, finden Sie im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für einen IAM-Benutzer in einem anderen AWS Konto, das Sie besitzen.

- Informationen dazu, wie Sie AWS Konten von Drittanbietern Zugriff auf Ihre Ressourcen gewähren, finden Sie im IAM-Benutzerhandbuch unter Zugriff auf AWS Konten, die Dritten gehören.
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter Gewähren von Zugriff für extern authentifizierte Benutzer (Identitätsverbund) im IAM-Benutzerhandbuch.
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

Pod-Container erhalten folgenden Fehler: **`An error occurred (SignatureDoesNotMatch) when calling the GetCallerIdentity operation: Credential should be scoped to a valid region`**

Ihre Container erhalten diesen Fehler, wenn Ihre Anwendung explizit Anfragen an den globalen AWS STS-Endpunkt (`https://sts.amazonaws`) sendet und Ihr Kubernetes-Dienstkonto für die Verwendung eines regionalen Endpunkts konfiguriert ist. Sie können das Problem mit einer der folgenden Optionen beheben:

- Aktualisieren Sie Ihren Anwendungscode, um explizite Aufrufe an den globalen AWS STS-Endpunkt zu entfernen.
- Aktualisieren Sie Ihren Anwendungscode, um explizite Anrufe an regionale Endpunkte wie `https://sts.us-west-2.amazonaws.com` zu senden. Ihre Anwendung sollte über eine integrierte Redundanz verfügen, sodass bei einem Ausfall des Dienstes in der AWS Region eine andere AWS Region ausgewählt werden kann. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter AWS STS in einer AWS Region verwalten.
- Konfigurieren Sie Ihre Servicekonten so, dass sie den globalen Endpunkt verwenden. Cluster verwenden standardmäßig den regionalen Endpunkt. Weitere Informationen finden Sie unter the section called “STS-Endpunkte”.

Amazon-EKS-Cluster-IAM-Rolle

Für jeden Cluster ist eine IAM-Rolle für Amazon-EKS-Cluster erforderlich. Von Amazon EKS verwaltete Kubernetes-Cluster verwenden diese Rolle zur Verwaltung von Knoten, und der veraltete Cloud-Anbieter nutzt diese Rolle, um Load Balancer mit Elastic Load Balancing für Services zu erstellen.

Bevor Sie Amazon-EKS-Cluster erstellen können, müssen Sie eine IAM-Rolle mit einer der folgenden IAM-Richtlinien erstellen:

- EKSClusterAmazon-Richtlinie
- Eine benutzerdefinierten IAM-Richtlinie. Die folgenden Mindestberechtigungen ermöglichen es dem Kubernetes-Cluster, Knoten zu verwalten, erlauben es dem alten Cloud-Anbieter jedoch nicht, Load Balancer mit Elastic Load Balancing zu erstellen. Ihre benutzerdefinierte IAM-Richtlinie muss mindestens folgende Berechtigungen haben:

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:CreateTags"
      ],
      "Resource": "arn:aws:ec2:*:*:instance/*",
      "Condition": {
        "ForAnyValue:StringLike": {
          "aws:TagKeys": "kubernetes.io/cluster/*"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DescribeVpcs",
        "ec2:DescribeDhcpOptions",
        "ec2:DescribeAvailabilityZones",
        "ec2:DescribeInstanceTopology",
        "kms:DescribeKey"
      ],
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> Vor dem 3. Oktober 2023 war eine EKSClusterAmazon-Richtlinie für die IAM-Rolle für jeden Cluster erforderlich.
> Vor dem 16. April 2020 waren EKSServiceAmazon-Richtlinien und EKSClusterAmazon-Richtlinien erforderlich, und der vorgeschlagene Name für die Rolle lauteteeksServiceRole. Mit der AWSServiceRoleForAmazonEKS serviceverknüpften Rolle ist die EKSServiceAmazon-Richtlinienrichtlinie für Cluster, die am oder nach dem 16. April 2020 erstellt wurden, nicht mehr erforderlich.

Überprüfen, ob eine Clusterrolle vorhanden ist

Mit dem folgenden Verfahren können Sie feststellen, ob Ihr Konto bereits über die Amazon-EKS-Clusterrolle verfügt.

1. Öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im linken Navigationsbereich Roles aus.
3. Suchen Sie in der Liste der Rollen nach `eksClusterRole`. Wenn keine Rolle mit `eksClusterRole` vorhanden ist, informieren Sie sich unter the section called “Erstellen der Amazon-EKS-Cluster-Rolle”, wie Sie die Rolle erstellen können. Wenn eine Rolle mit `eksClusterRole` vorhanden ist, wählen Sie die Rolle aus, um die angefügten Richtlinien anzuzeigen.
4. Wählen Sie Berechtigungen.
5. Stellen Sie sicher, dass die von Amazon EKSCluster Policy verwaltete Richtlinie der Rolle zugeordnet ist. Wenn die Richtlinie angefügt ist, ist Ihre Amazon-EKS-Clusterrolle korrekt konfiguriert.
6. Wählen Sie Vertrauensstellungen und dann Vertrauensrichtlinie bearbeiten aus.
7. Überprüfen Sie, dass die Vertrauensstellung die folgende Richtlinie enthält. Wenn die Vertrauensstellung mit der folgenden Richtlinie übereinstimmt, wählen Sie Abbrechen aus. Andernfalls kopieren Sie die Richtlinie in das Fenster Vertrauensrichtlinie bearbeiten und wählen Sie Richtlinie aktualisieren aus.

```
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Erstellen der Amazon-EKS-Cluster-Rolle

Sie können die AWS-Managementkonsole oder die AWS CLI verwenden, um die Clusterrolle zu erstellen.

AWS-Managementkonsole

a. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

b. Wählen Sie Roles (Rollen) und anschließend Create Role (Rolle erstellen) aus.

c. Wählen Sie unter Vertrauenswürdiger Entitätstyp die Option AWS Dienst aus.

d. Wählen Sie aus der Dropdownliste Anwendungsfälle für andere AWS Dienste die Option EKS aus.

e. Wählen Sie EKS – Cluster für Ihren Anwendungsfall und dann Weiter aus.

f. Wählen Sie auf der Registerkarte Add permissions (Berechtigungen hinzufügen) die Option Next (Weiter) aus.

g. Geben Sie unter Role name (Rollenname) einen eindeutigen Namen für die Rolle ein, z. B. `eksClusterRole`.

h. Geben Sie unter Beschreibung einen beschreibenden Text, wie z. B. `Amazon EKS - Cluster role`, ein.

i. Wählen Sie Rolle erstellen aus.

AWS CLI

a. Kopieren Sie den folgenden Inhalt in eine Datei namens *`cluster-trust-policy.json`*.

```
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

b. Erstellen Sie die -Rolle. Sie können *eksClusterRole* mit einem beliebigen Namen ersetzen, den Sie wählen.

```
aws iam create-role \
  --role-name eksClusterRole \
  --assume-role-policy-document file://"cluster-trust-policy.json"
```

c. Fügen Sie der Rolle die erforderliche IAM-Richtlinie an.

```
aws iam attach-role-policy \
  --policy-arn arn:aws: iam::aws:policy/AmazonEKSClusterPolicy \
  --role-name eksClusterRole
```

Amazon-EKS-Knoten-IAM-Rolle

Der Amazon EKS Node `kubelet` Daemon ruft in Ihrem Namen AWS APIs an. Knoten erhalten über ein IAM-Instance-Profil und zugehörige Richtlinien Berechtigungen für diese API-Aufrufe. Bevor Sie Knoten starten und in einem Cluster registrieren können, müssen Sie eine IAM-Rolle erstellen, die diese Knoten beim Start verwenden können. Diese Anforderung gilt für Knoten, die mit dem von Amazon bereitgestellten Amazon EKS-optimierten AMI oder mit einem anderen Knoten AMIs , den Sie verwenden möchten, gestartet wurden. Darüber hinaus gilt diese Anforderung sowohl für verwaltete Knotengruppen als auch für selbstverwaltete Knoten.

> **Note**
>
> Sie können nicht dieselbe Rolle verwenden, die zum Erstellen von Clustern verwendet wurde.

Bevor Sie Knoten erstellen, müssen Sie eine IAM-Rolle mit folgenden Berechtigungen erstellen:

- Berechtigungen für `kubelet` die Beschreibung von EC2 Amazon-Ressourcen in der VPC, wie sie in der EKSWorkerNodePolicyAmazon-Richtlinie vorgesehen sind. Diese Richtlinie stellt auch die Berechtigungen für den Amazon EKS Pod Identity Agent bereit.
- Berechtigungen für die `kubelet` Verwendung von Container-Images aus Amazon Elastic Container Registry (Amazon ECR), wie sie beispielsweise in der EC2ContainerRegistryPullOnlyAmazon-Richtlinie vorgesehen sind. Die Berechtigungen zur Verwendung von Container-Images aus Amazon Elastic Container Registry (Amazon ECR) sind erforderlich, da die integrierten Add-Ons für Netzwerke Pods ausführen, die Container-Images von Amazon ECR verwenden.
- (Optional) Berechtigungen für den Amazon EKS Pod Identity Agent zum Verwenden der Aktion `eks-auth:AssumeRoleForPodIdentity`, um Anmeldeinformationen für Pods abzurufen. Wenn Sie Amazon nicht verwenden EKSWorkerNodePolicy, müssen Sie diese Genehmigung zusätzlich zu den EC2 Berechtigungen zur Verwendung von EKS Pod Identity erteilen.
- (Optional) Wenn Sie IRSA oder EKS Pod Identity nicht verwenden, um Berechtigungen für die VPC-CNI-Pods zu gewähren, müssen Sie Berechtigungen für das VPC-CNI in der Instance-Rolle bereitstellen. Sie können entweder die `AmazonEKS_CNI_Policy`verwaltete Richtlinie (wenn Sie Ihren Cluster mit der `IPv4` Familie erstellt haben) oder eine von Ihnen erstellte IPv6 Richtlinie (wenn Sie Ihren Cluster mit der `IPv6` Familie erstellt haben) verwenden. Anstatt die Richtlinie jedoch an diese Rolle anzuhängen, empfehlen wir Ihnen, die Richtlinie an eine separate Rolle anzuhängen, die speziell für das Amazon VPC CNI-Add-on verwendet wird. Weitere Informationen zum Erstellen einer separaten Rolle für das Amazon VPC CNI-Add-on finden Sie unter the section called “Konfiguration für IRSA”.

> **Note**
>
> Die EC2 Amazon-Knotengruppen müssen eine andere IAM-Rolle als das Fargate-Profil haben. Weitere Informationen finden Sie unter the section called “IAM-Rolle zur Pod-Ausführung”.

Nach einer vorhandenen Knotenrolle suchen

Mit dem folgenden Verfahren können Sie feststellen, ob Ihr Konto bereits über die Amazon-EKS-Knotenrolle verfügt.

1. Öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/

2. Wählen Sie im linken Navigationsbereich Roles aus.
3. Suchen Sie in der Liste der Rollen nach `eksNodeRole`, `AmazonEKSNodeRole` oder `NodeInstanceRole`. Wenn eine Rolle mit einem dieser Namen nicht existiert, erfahren Sie unter the section called “Erstellen der Amazon-EKS-Knoten-IAM-Rolle”, wie Sie die Rolle erstellen. Wenn eine Rolle mit `eksNodeRole`, `AmazonEKSNodeRole` oder `NodeInstanceRole` vorhanden ist, wählen Sie die Rolle aus, um die angehängten Richtlinien anzuzeigen.
4. Wählen Sie Berechtigungen.
5. Stellen Sie sicher, dass die von Amazon EKSWorker NodePolicy und Amazon EC2 ContainerRegistryPullOnly verwalteten Richtlinien an die Rolle angehängt sind oder dass eine benutzerdefinierte Richtlinie mit den Mindestberechtigungen angehängt ist.

> **Note**
>
> Wenn die AmazonEKS_CNI_Policy-Richtlinie an die Rolle angehängt ist, empfehlen wir, sie zu entfernen und einer IAM-Rolle anzuhängen, die stattdessen dem `aws-node`-Kubernetes-Servicekonto zugeordnet ist. Weitere Informationen finden Sie unter the section called “Konfiguration für IRSA”.

6. Wählen Sie Trust Relationships (Vertrauensstellungen) und dann Edit trust policy (Vertrauensrichtlinie bearbeiten) aus.
7. Überprüfen Sie, dass die Vertrauensstellung die folgende Richtlinie enthält. Wenn die Vertrauensstellung mit der folgenden Richtlinie übereinstimmt, wählen Sie Abbrechen aus. Andernfalls kopieren Sie die Richtlinie in das Fenster Vertrauensrichtlinie bearbeiten und wählen Sie Richtlinie aktualisieren aus.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRole"
            ],
            "Principal": {
                "Service": [
                    "ec2.amazonaws.com"
                ]
            }
```

```
        }
    ]
}
```

Erstellen der Amazon-EKS-Knoten-IAM-Rolle

Sie können die Node-IAM-Rolle mit der AWS-Managementkonsole oder der AWS CLI erstellen.

AWS-Managementkonsole

a. Öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
b. Wählen Sie im linken Navigationsbereich Roles aus.
c. Klicken Sie auf der Seite Roles (Rollen) auf Create role (Rolle erstellen).
d. Gehen Sie auf der Seite Select trusted entity (Vertrauenswürdige Entität auswählen) wie folgt vor:
    i. Wählen Sie im Abschnitt Vertrauenswürdiger Entitätstyp die Option AWS Service aus.
    ii. Wählen Sie unter Use case (Anwendungsfall) EC2 aus.
    iii. Wählen Sie Weiter aus.
e. Führen Sie auf der Seite Add permissions (Berechtigungen hinzufügen) die folgenden Schritte aus:
    i. Geben Sie im Feld Filter policies (Filterrichtlinien) `AmazonEKSWorkerNodePolicy` ein.
    ii. Aktivieren Sie EKSWorker NodePolicy in den Suchergebnissen das Kontrollkästchen links neben Amazon.
    iii. Wählen Sie Clear filters (Filter löschen) aus.
    iv. Geben Sie im Feld Filter policies (Filterrichtlinien) `AmazonEC2ContainerRegistryPullOnly` ein.
    v. Aktivieren Sie EC2 ContainerRegistryPullOnly in den Suchergebnissen das Kontrollkästchen links neben Amazon.

        Entweder die von Amazoneks_CNI_Policy verwaltete Richtlinie oder eine von Ihnen erstellte IPv6 Richtlinie muss ebenfalls entweder dieser Rolle oder einer anderen Rolle zugewiesen werden, die dem Kubernetes-Dienstkonto zugeordnet ist. `aws-node` Wir empfehlen, die Richtlinie der Rolle zuzuweisen, die dem Kubernetes-Dienstkonto zugeordnet ist, anstatt sie dieser Rolle zuzuweisen. Weitere Informationen finden Sie unter the section called “Konfiguration für IRSA”.

vi. Wählen Sie Weiter aus.

f. Gehen Sie auf der Seite Name, review, and create (Benennen, überprüfen und erstellen) wie folgt vor:

i. Geben Sie unter Role name (Rollenname) einen eindeutigen Namen für die Rolle ein, z. B. `AmazonEKSNodeRole`.

ii. Ersetzen Sie unter Description (Beschreibung) den aktuellen Text durch beschreibenden Text wie beispielsweise `Amazon EKS - Node role`.

iii. Fügen Sie der Rolle unter Tags hinzufügen (optional) Metadaten hinzu, indem Sie Tags als Schlüssel-Wert-Paare anfügen. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen im IAM-Benutzerhandbuch.

iv. Wählen Sie Rolle erstellen aus.

AWS CLI

a. Führen Sie den folgenden Befehl aus, um die Datei `node-role-trust-relationship.json` zu erstellen.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRole"
            ],
            "Principal": {
                "Service": [
                    "ec2.amazonaws.com"
                ]
            }
        }
    ]
}
```

b. Erstellen Sie die IAM-Rolle.

```
aws iam create-role \
  --role-name AmazonEKSNodeRole \
  --assume-role-policy-document file://"node-role-trust-relationship.json"
```

c. Hängen Sie die beiden erforderlichen verwalteten IAM-Richtlinien an die IAM-Rolle an.

```
aws iam attach-role-policy \
  --policy-arn arn:aws: iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name AmazonEKSNodeRole
aws iam attach-role-policy \
  --policy-arn arn:aws: iam::aws:policy/AmazonEC2ContainerRegistryPullOnly \
  --role-name AmazonEKSNodeRole
```

d. Hängen Sie eine der folgenden IAM-Richtlinien an die IAM-Rolle an, je nachdem, mit welcher IP-Familie Sie Ihren Cluster erstellt haben. Die Richtlinie muss an diese Rolle oder an eine Rolle angefügt werden, die dem Kubernetes `aws-node`-Servicekonto zugeordnet ist, das für das Amazon-VPC-CNI-Plugin verwendet wird. Wir empfehlen, die Richtlinie der Rolle zuzuweisen, die dem Kubernetes-Servicekonto zugeordnet ist. Um die Richtlinie der Rolle zuzuweisen, die dem Kubernetes-Servicekonto zugeordnet ist, siehe the section called “Konfiguration für IRSA”.

- IPv4

```
aws iam attach-role-policy \
  --policy-arn arn:aws: iam::aws:policy/AmazonEKS_CNI_Policy \
  --role-name AmazonEKSNodeRole
```

- IPv6

A. Kopieren Sie den folgenden Text und speichern Sie ihn in einer Datei mit dem Namen `vpc-cni-ipv6-policy.json`.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AssignIpv6Addresses",
                "ec2:DescribeInstances",
                "ec2:DescribeTags",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeInstanceTypes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
```

```
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ]
        }
    ]
}
```

B. Erstellen Sie die IAM-Richtlinie.

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-
document file://vpc-cni-ipv6-policy.json
```

C. Fügen Sie die IAM-Richtlinie an die IAM-Rolle an. Ersetzen Sie *111122223333* durch Ihre Konto-ID.

```
aws iam attach-role-policy \
  --policy-arn arn:aws: iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy \
  --role-name AmazonEKSNodeRole
```

IAM-Rolle für Cluster von Amazon EKS Auto Mode

Für jeden Cluster ist eine IAM-Rolle für Amazon-EKS-Cluster erforderlich. Von Amazon EKS verwaltete Kubernetes-Cluster nutzen diese Rolle, um Routineaufgaben für Speicher, Netzwerk und automatische Skalierung von Rechenleistung zu automatisieren.

Bevor Sie Amazon-EKS-Cluster erstellen können, müssen Sie eine IAM-Rolle mit den für EKS Auto Mode erforderlichen Richtlinien erstellen. Sie können entweder die vorgeschlagenen AWS IAM-verwalteten Richtlinien anhängen oder benutzerdefinierte Richtlinien mit entsprechenden Berechtigungen erstellen.

- EKSComputeAmazon-Richtlinie
- AmazonEKSBlockStoragePolicy
- AmazonEKSLoadBalancingPolicy
- EKSNetworkingAmazon-Richtlinie
- EKSClusterAmazon-Richtlinie

Überprüfen, ob eine Clusterrolle vorhanden ist

Mit dem folgenden Verfahren können Sie feststellen, ob Ihr Konto bereits über die Amazon-EKS-Clusterrolle verfügt.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich Roles aus.
3. Suchen Sie in der Liste der Rollen nach `AmazonEKSAutoClusterRole`. Wenn eine Rolle, die `AmazonEKSAutoClusterRole` enthält, nicht vorhanden ist, befolgen Sie die Anweisungen im nächsten Abschnitt, um die Rolle zu erstellen. Wenn eine Rolle mit `AmazonEKSAutoClusterRole` vorhanden ist, wählen Sie die Rolle aus, um die angefügten Richtlinien anzuzeigen.
4. Wählen Sie Berechtigungen.
5. Stellen Sie sicher, dass die von Amazon EKSCluster Policy verwaltete Richtlinie der Rolle zugeordnet ist. Wenn die Richtlinie angefügt ist, ist Ihre Amazon-EKS-Clusterrolle korrekt konfiguriert.
6. Wählen Sie Vertrauensstellungen und dann Vertrauensrichtlinie bearbeiten aus.
7. Überprüfen Sie, dass die Vertrauensstellung die folgende Richtlinie enthält. Wenn die Vertrauensstellung mit der folgenden Richtlinie übereinstimmt, wählen Sie Abbrechen aus. Andernfalls kopieren Sie die Richtlinie in das Fenster Vertrauensrichtlinie bearbeiten und wählen Sie Richtlinie aktualisieren aus.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
```

> **Note**
>
> AWS benötigt den Namen `AmazonEKSAutoClusterRole` für diese Rolle nicht.

Erstellen der Amazon-EKS-Cluster-Rolle

Sie können die AWS-Managementkonsole oder die AWS CLI verwenden, um die Clusterrolle zu erstellen.

AWS-Managementkonsole

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie Roles (Rollen) und anschließend Create Role (Rolle erstellen) aus.
3. Wählen Sie unter Vertrauenswürdiger Entitätstyp die Option AWS Dienst aus.
4. Wählen Sie aus der Dropdownliste Anwendungsfälle für andere AWS Dienste die Option EKS aus.
5. Wählen Sie EKS – Cluster für Ihren Anwendungsfall und dann Weiter aus.
6. Wählen Sie auf der Registerkarte Berechtigungen hinzufügen die Richtlinien aus und klicken Sie dann auf Weiter.
   - EKSComputeAmazon-Richtlinie
   - AmazonEKSBlockStoragePolicy
   - AmazonEKSLoadBalancingPolicy
   - EKSNetworkingAmazon-Richtlinie
   - EKSClusterAmazon-Richtlinie
7. Geben Sie unter Rollenname einen eindeutigen Namen für die Rolle ein, z. B. `AmazonEKSAutoClusterRole`.
8. Geben Sie unter Beschreibung einen beschreibenden Text, wie z. B. `Amazon EKS - Cluster role`, ein.
9. Wählen Sie Rolle erstellen aus.

AWS CLI

1. Kopieren Sie den folgenden Inhalt in eine Datei namens *`cluster-trust-policy.json`*.

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
```

2. Erstellen Sie die -Rolle. Sie können *AmazonEKSAutoClusterRole* mit einem beliebigen Namen ersetzen, den Sie wählen.

```
aws iam create-role \
  --role-name AmazonEKSAutoClusterRole \
  --assume-role-policy-document file://"cluster-trust-policy.json"
```

3. Fügen Sie der Rolle die erforderlichen IAM-Richtlinien an:

EKSClusterAmazon-Richtlinie:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoClusterRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSClusterPolicy
```

EKSComputeAmazon-Richtlinie:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoClusterRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSComputePolicy
```

Amazon EKSBlock StoragePolicy:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoClusterRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSBlockStoragePolicy
```

Amazon EKSLoad BalancingPolicy:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoClusterRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSLoadBalancingPolicy
```

EKSNetworkingAmazon-Richtlinie:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoClusterRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSNetworkingPolicy
```

IAM-Rolle für Knoten im Amazon EKS Auto Mode

> **Note**
>
> Sie können nicht dieselbe Rolle verwenden, die zum Erstellen von Clustern verwendet wurde.

Bevor Sie Knoten erstellen, müssen Sie eine IAM-Rolle mit den folgenden Richtlinien oder entsprechenden Berechtigungen erstellen:

- AmazonEKSWorkerNodeMinimalPolicy
- AmazonEC2ContainerRegistryPullOnly

Nach einer vorhandenen Knotenrolle suchen

Mit dem folgenden Verfahren können Sie feststellen, ob Ihr Konto bereits über die Amazon-EKS-Knotenrolle verfügt.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich Roles aus.
3. Suchen Sie in der Liste der Rollen nach `AmazonEKSAutoNodeRole`. Wenn eine Rolle mit einem dieser Namen nicht vorhanden ist, lesen Sie die Anweisungen zum Erstellen der Rolle im nächsten Abschnitt. Wenn eine Rolle mit `AmazonEKSAutoNodeRole` vorhanden ist, wählen Sie die Rolle aus, um die angehängten Richtlinien anzuzeigen.
4. Wählen Sie Berechtigungen.

5. Stellen Sie sicher, dass die oben genannten erforderlichen Richtlinien oder entsprechende benutzerdefinierte Richtlinien angefügt sind.
6. Wählen Sie Vertrauensstellungen und dann Vertrauensrichtlinie bearbeiten aus.
7. Überprüfen Sie, dass die Vertrauensstellung die folgende Richtlinie enthält. Wenn die Vertrauensstellung mit der folgenden Richtlinie übereinstimmt, wählen Sie Abbrechen aus. Andernfalls kopieren Sie die Richtlinie in das Fenster Vertrauensrichtlinie bearbeiten und wählen Sie Richtlinie aktualisieren aus.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Erstellen der Amazon-EKS-Knoten-IAM-Rolle

Sie können die Node-IAM-Rolle mit der AWS-Managementkonsole oder der AWS CLI erstellen.

AWS-Managementkonsole

1. Öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im linken Navigationsbereich Roles aus.
3. Klicken Sie auf der Seite Roles (Rollen) auf Create role (Rolle erstellen).
4. Gehen Sie auf der Seite Select trusted entity (Vertrauenswürdige Entität auswählen) wie folgt vor:
   a. Wählen Sie im Abschnitt Vertrauenswürdiger Entitätstyp die Option AWS Service aus.
   b. Wählen Sie unter Use case (Anwendungsfall) EC2 aus.
   c. Wählen Sie Weiter aus.
5. Fügen Sie auf der Seite Berechtigungen hinzufügen die folgenden Richtlinien hinzu:
   - AmazonEKSWorkerNodeMinimalPolicy
   - AmazonEC2ContainerRegistryPullOnly

6. Gehen Sie auf der Seite Name, review, and create (Benennen, überprüfen und erstellen) wie folgt vor:
   a. Geben Sie unter Role name (Rollenname) einen eindeutigen Namen für die Rolle ein, z. B. `AmazonEKSAutoNodeRole`.
   b. Ersetzen Sie unter Description (Beschreibung) den aktuellen Text durch beschreibenden Text wie beispielsweise `Amazon EKS - Node role`.
   c. Fügen Sie der Rolle unter Tags hinzufügen (optional) Metadaten hinzu, indem Sie Tags als Schlüssel-Wert-Paare anfügen. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen im IAM-Benutzerhandbuch.
   d. Wählen Sie Rolle erstellen aus.

AWS CLI

Knoten-IAM-Rolle erstellen

Verwenden Sie die node-trust-policyJSON-Datei aus dem vorherigen Schritt, um zu definieren, welche Entitäten die Rolle übernehmen können. Führen Sie den folgenden Befehl aus, um die Knoten-IAM-Rolle zu erstellen:

```
aws iam create-role \
    --role-name AmazonEKSAutoNodeRole \
    --assume-role-policy-document file://node-trust-policy.json
```

Rollen-ARN beachten

Rufen Sie nach dem Erstellen der Rolle die ARN der Knoten-IAM-Rolle ab und speichern Sie sie. Sie benötigen diese ARN in den nachfolgenden Schritten. Um die ARN abzurufen, verwenden Sie den folgenden Befehl.

```
aws iam get-role --role-name AmazonEKSAutoNodeRole --query "Role.Arn" --output text
```

Erforderliche Richtlinien anfügen

Fügen Sie der Node-IAM-Rolle die folgenden AWS verwalteten Richtlinien hinzu, um die erforderlichen Berechtigungen bereitzustellen:

Um Amazon anzuhängen EKSWorkerNodeMinimalPolicy:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoNodeRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEKSWorkerNodeMinimalPolicy
```

Um Amazon anzuhängen EC2ContainerRegistryPullOnly:

```
aws iam attach-role-policy \
    --role-name AmazonEKSAutoNodeRole \
    --policy-arn arn:aws: iam::aws:policy/AmazonEC2ContainerRegistryPullOnly
```

IAM-Rolle für Amazon EKS-Funktionen

EKS-Funktionen erfordern die Konfiguration einer Capability-IAM-Rolle (oder Fähigkeitsrolle). Capabilities verwenden diese Rolle, um Aktionen für AWS Dienste auszuführen und über automatisch erstellte Zugriffseinträge auf Kubernetes-Ressourcen in Ihrem Cluster zuzugreifen.

Bevor Sie bei der Funktionserstellung eine Funktionsrolle angeben können, müssen Sie die IAM-Rolle mit den entsprechenden Vertrauensrichtlinien und Berechtigungen für den Funktionstyp erstellen. Sobald diese IAM-Rolle erstellt wurde, kann sie für eine beliebige Anzahl von Funktionsressourcen wiederverwendet werden.

Anforderungen an die Rolle der Fähigkeit

Die Fähigkeitsrolle muss die folgenden Anforderungen erfüllen:

- Die Rolle muss sich in demselben AWS Konto befinden wie der Cluster und die Capability-Ressource
- Die Rolle muss über eine Vertrauensrichtlinie verfügen, die es dem EKS-Capabilities-Service ermöglicht, die Rolle zu übernehmen
- Die Rolle muss über Berechtigungen verfügen, die dem Funktionstyp und den Anforderungen des Anwendungsfalls entsprechen (siehethe section called “Berechtigungen nach Funktionstyp”)

Vertrauensrichtlinie für Funktionsrollen

Alle Funktionsrollen müssen die folgende Vertrauensrichtlinie beinhalten:

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "capabilities.eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
```

Diese Vertrauensrichtlinie ermöglicht EKS:

- Übernehmen Sie die Rolle zur Durchführung von AWS API-Vorgängen
- Taggen Sie Sitzungen zu Prüf- und Nachverfolgungszwecken

Berechtigungen nach Funktionstyp

Die erforderlichen IAM-Berechtigungen hängen davon ab, welche Funktion Sie verwenden und welches Bereitstellungsmodell Sie verwenden.

> **Note**
>
> Bei Produktionsbereitstellungen, bei denen IAM-Rollenselektoren mit ACK verwendet werden, oder bei Verwendung von Kro- oder Argo-CD ohne AWS Serviceintegration sind für die Capability Role möglicherweise keine IAM-Berechtigungen erforderlich, die über die Vertrauensrichtlinie hinausgehen.

kro (Kube Resource Orchestrator)

Es sind keine IAM-Berechtigungen erforderlich. Sie können eine Funktionsrolle ohne angehängte Richtlinien erstellen. kro benötigt lediglich Kubernetes-RBAC-Berechtigungen, um Kubernetes-Ressourcen zu erstellen und zu verwalten.

AWS Controller für Kubernetes (ACK)

ACK unterstützt zwei Berechtigungsmodelle:

- Einfache Einrichtung (Entwicklung/Testen): Fügen Sie AWS Dienstberechtigungen direkt zur Capability-Rolle hinzu. Dies eignet sich gut für den Einstieg, für Bereitstellungen mit einem Konto oder wenn alle Benutzer dieselben Berechtigungen benötigen.
- Bewährte Methode für die Produktion: Verwenden Sie IAM-Rollenselektoren, um den Zugriff mit den geringsten Rechten zu implementieren. Bei diesem Ansatz benötigt die Capability-Rolle nur die `sts:AssumeRole` Erlaubnis, dienstspezifische Rollen anzunehmen. Sie fügen der Capability-Rolle selbst keine AWS Dienstberechtigungen (wie S3 oder RDS) hinzu — diese Berechtigungen werden einzelnen IAM-Rollen gewährt, die bestimmten Namespaces zugeordnet sind.

  IAM-Rollenselektoren ermöglichen:

  - Isolierung von Berechtigungen auf Namespace-Ebene
  - Kontoübergreifende Ressourcenverwaltung
  - Teamspezifische IAM-Rollen
  - Sicherheitsmodell mit den geringsten Rechten

  Beispiel für eine Capability Role Policy für den IAM Role Selector-Ansatz:

  ```
  {
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": "sts:AssumeRole",
        "Resource": [
          "arn:aws:iam::111122223333:role/ACK-S3-Role",
          "arn:aws:iam::111122223333:role/ACK-RDS-Role",
          "arn:aws:iam::444455556666:role/ACKCrossAccountRole"
        ]
      }
    ]
  }
  ```

  Eine ausführliche Konfiguration der ACK-Berechtigungen einschließlich IAM-Rollenselektoren finden Sie unter. the section called “Konfigurieren von -Berechtigungen”

Argo-CD

Standardmäßig sind keine IAM-Berechtigungen erforderlich. Optionale Berechtigungen können erforderlich sein für:

- AWS Secrets Manager: Wenn Sie Secrets Manager zum Speichern von Git-Repository-Anmeldeinformationen verwenden
- AWS CodeConnections: Bei Verwendung CodeConnections für die Git-Repository-Authentifizierung

  Beispielrichtlinie für Secrets Manager und CodeConnections:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue",
        "secretsmanager:DescribeSecret"
      ],
      "Resource": "arn:aws:secretsmanager:region:account-id:secret:argocd/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "codeconnections:UseConnection",
        "codeconnections:GetConnection"
      ],
      "Resource": "arn:aws:codeconnections:region:account-id:connection/*"
    }
  ]
}
```

  Ausführliche Informationen zu den Argo-CD-Genehmigungsanforderungen finden Sie unterthe section called “Überlegungen zu Argo CD”.

Suchen Sie nach einer vorhandenen Capability-Rolle

Mithilfe des folgenden Verfahrens können Sie überprüfen, ob Ihr Konto bereits über eine für Ihren Anwendungsfall geeignete Capability IAM-Rolle verfügt.

1. Öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im linken Navigationsbereich Roles aus.

3. Suchen Sie in der Rollenliste nach dem Namen Ihrer Funktionsrolle (z. B. `ACKCapabilityRole` oder`ArgoCDCapabilityRole`).
4. Wenn eine Rolle vorhanden ist, wählen Sie sie aus, um die angehängten Richtlinien und die Vertrauensstellung anzuzeigen.
5. Wählen Sie Vertrauensstellungen und dann Vertrauensrichtlinie bearbeiten aus.
6. Stellen Sie sicher, dass die Vertrauensstellung der Capability Trust-Richtlinie entspricht. Wenn sie nicht übereinstimmt, aktualisieren Sie die Vertrauensrichtlinie.
7. Wählen Sie Berechtigungen aus und stellen Sie sicher, dass die Rolle über die entsprechenden Berechtigungen für Ihren Funktionstyp und Ihren Anwendungsfall verfügt.

Eine Capability-IAM-Rolle erstellen

Sie können die AWS-Managementkonsole oder die AWS CLI verwenden, um eine Capability-Rolle zu erstellen.

AWS-Managementkonsole

a. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
b. Wählen Sie Roles (Rollen) und anschließend Create Role (Rolle erstellen) aus.
c. Wählen Sie unter Vertrauenswürdiger Entitätstyp die Option Benutzerdefinierte Vertrauensrichtlinie aus.
d. Kopieren Sie die Capability Trust Policy und fügen Sie sie in den Trust Policy Editor ein.
e. Wählen Sie Weiter aus.
f. Wählen Sie auf der Registerkarte „Berechtigungen hinzufügen“ Richtlinien aus, die Ihrem Funktionstyp entsprechen, oder erstellen Sie sie (siehethe section called “Berechtigungen nach Funktionstyp”). Für Kro können Sie diesen Schritt überspringen.
g. Wählen Sie Weiter aus.
h. Geben Sie unter Rollenname einen eindeutigen Namen für Ihre Rolle ein, z. B. `ACKCapabilityRoleArgoCDCapabilityRole`, oder`kroCapabilityRole`.
i. Geben Sie unter Beschreibung einen beschreibenden Text, wie z. B. `Amazon EKS - ACK capability role`, ein.
j. Wählen Sie Rolle erstellen aus.

AWS -CLI

a. Kopieren Sie die Capability Trust-Richtlinie in eine Datei mit dem Name`capability-trust-policy.json`.

b. Erstellen Sie die Rolle. `ACKCapabilityRole`Ersetzen Sie sie durch den gewünschten Rollennamen.

```
aws iam create-role \
  --role-name ACKCapabilityRole \
  --assume-role-policy-document file://capability-trust-policy.json
```

c. Hängen Sie die erforderlichen IAM-Richtlinien an die Rolle an. Fügen Sie für ACK Richtlinien für die AWS Dienste hinzu, die Sie verwalten möchten. Fügen Sie für Argo CD Richtlinien für Secrets Manager oder bei CodeConnections Bedarf hinzu. Für Kro können Sie diesen Schritt überspringen.

   Beispiel für ACK mit S3-Berechtigungen:

```
aws iam put-role-policy \
  --role-name ACKCapabilityRole \
  --policy-name S3Management \
  --policy-document file://s3-policy.json
```

Behebung von Problemen mit Funktionsrollen

Die Erstellung von Funktionen schlägt mit der Meldung „Ungültige IAM-Rolle“ fehl

Vergewissern Sie sich, dass folgende Bedingungen erfüllt sind:

- Die Rolle ist in demselben Konto wie der Cluster vorhanden
- Die Vertrauensrichtlinie entspricht der Capability Trust-Richtlinie
- Sie haben die `iam:PassRole` Erlaubnis für die Rolle

Die Funktion zeigt Berechtigungsfehler

Vergewissern Sie sich, dass folgende Bedingungen erfüllt sind:

- Die Rolle verfügt über die erforderlichen IAM-Berechtigungen für den Fähigkeitstyp
- Der Zugriffseintrag für die Rolle ist auf dem Cluster vorhanden
- Zusätzliche Kubernetes-Berechtigungen werden bei Bedarf konfiguriert (siehe) the section called “Zusätzliche Kubernetes-Berechtigungen”

ACK-Ressourcen schlagen mit der Fehlermeldung „Zugriff verweigert“ fehl

Vergewissern Sie sich, dass folgende Bedingungen erfüllt sind:

- Die Rolle verfügt über die erforderlichen IAM-Berechtigungen für Ihren Anwendungsfall
- Stellen Sie bei ACK-Controllern, die auf geheime Daten verweisen, sicher, dass Sie den `AmazonEKSSecretReaderPolicy` Geltungsbereich der Zugriffsrichtlinie den entsprechenden Namespaces zugeordnet haben.

Weitere Anleitungen zur Fehlerbehebung finden Sie unter. the section called “Überlegungen zu den Funktionen von EKS”

Überwachung der Cluster-Leistung und Anzeige von Protokollen

Sie können Ihre Daten in Amazon EKS mit vielen verfügbaren Überwachungs- oder Protokollierungstools beobachten. Ihre Amazon EKS-Protokolldaten können an AWS Services oder Partner-Tools zur Datenanalyse gestreamt werden. In der sind viele Dienste verfügbar AWS-Managementkonsole , die Daten zur Behebung Ihrer Amazon EKS-Probleme bereitstellen. Sie können auch eine AWS-unterstützte Open-Source-Lösung für die Überwachung der Amazon EKS-Infrastruktur verwenden.

Nachdem Sie im linken Navigationsbereich der Amazon-EKS-Konsole Cluster ausgewählt haben, können Sie den Zustand und die Details des Clusters anzeigen, indem Sie den Namen Ihres Clusters und die Registerkarte Beobachtbarkeit auswählen. Informationen zu vorhandenen Kubernetes-Ressourcen, die in Ihrem Cluster bereitgestellt werden, finden Sie unter the section called “Zugriff auf Cluster-Ressourcen”.

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung von Amazon EKS und Ihren AWS Lösungen. Wir empfehlen Ihnen, Überwachungsdaten aus allen Teilen Ihrer AWS Lösung zu sammeln. Auf diese Weise können Sie Ausfälle, die sich über mehrere Punkte erstrecken, leichter debuggen. Bevor Sie mit der Überwachung von Amazon EKS beginnen, sollten Sie sicherstellen, dass für Ihren Überwachungsplan folgende Fragen beantwortet sind.

- Was sind Ihre Ziele? Benötigen Sie Echtzeit-Benachrichtigungen, wenn Ihre Cluster drastisch skaliert werden?
- Welche Ressourcen müssen beachtet werden?
- Wie oft müssen Sie diese Ressourcen beobachten? Muss Ihr Unternehmen schnell auf Risiken reagieren?
- Welche Tools möchten Sie verwenden? Wenn Sie AWS Fargate bereits als Teil Ihres Starts ausführen, können Sie den integrierten Log-Router verwenden.
- Wer soll die Überwachungsaufgaben ausführen?
- An wen sollen Benachrichtigungen gesendet werden, wenn etwas schief geht?

Überwachung und Protokollierung in Amazon EKS

Amazon EKS bietet integrierte Tools für die Überwachung und Protokollierung. Für unterstützte Versionen bietet das Dashboard zur Beobachtbarkeit Einblick in die Leistung Ihres Clusters. Es unterstützt Sie dabei, Probleme schnell zu erkennen, zu beheben und zu lösen. Zusätzlich zu den Überwachungsfeatures enthält es Listen, die auf den Audit-Protokollen der Steuerebene basieren. Die Kubernetes-Steuerebene stellt eine Reihe von Metriken zur Verfügung, die auch außerhalb der Konsole abgerufen werden können.

Die Protokollierung der Steuerebene zeichnet alle API-Aufrufe zu Ihren Clustern. Diese enthält Überwachungsinformationen darüber, welche Benutzer welche Aktionen an Ihren Clustern durchgeführt haben, sowie rollenbasierte Informationen. Weitere Informationen finden Sie unter Protokollierung und Überwachung auf Amazon EKS in den AWS präskriptiven Leitlinien.

Die Protokollierung auf der Amazon EKS-Kontrollebene stellt Prüf- und Diagnoseprotokolle direkt von der Amazon EKS-Steuerebene zu den CloudWatch Protokollen in Ihrem Konto bereit. Diese Protokolle erleichtern Ihnen die Absicherung und Ausführung Ihrer Cluster. Sie können genau die Protokolltypen auswählen, die Sie benötigen, und die Protokolle werden als Protokollstreams an eine Gruppe für jeden Amazon EKS-Cluster gesendet CloudWatch. Weitere Informationen finden Sie unter the section called “Protokolle der Steuerebene”.

> **Note**
>
> Wenn Sie die Amazon EKS-Authentifikatorprotokolle in Amazon überprüfen CloudWatch, werden die Einträge angezeigt, die Text enthalten, der dem folgenden Beispieltext ähnelt.
>
> ```
> level=info msg="mapping IAM role" groups="[]" role="arn:aws:
>  iam::111122223333:role/XXXXXXXXXXXXXXXXX-NodeManagerRole-XXXXXXXX"
>  username="eks:node-manager"
> ```
>
> Einträge, die diesen Text enthalten, werden erwartet. Das `username` ist eine interne Amazon EKS-Servicerolle, die bestimmte Vorgänge für verwaltete Knotengruppen und Fargate ausführt.
> Für eine detaillierte, anpassbare Protokollierung ist Kubernetes-Protokollierung verfügbar.

Amazon EKS ist in einen Service integriert AWS CloudTrail, der eine Aufzeichnung der Aktionen bereitstellt, die von einem Benutzer, einer Rolle oder einem AWS Service in Amazon EKS ausgeführt

wurden. CloudTrail erfasst alle API-Aufrufe für Amazon EKS als Ereignisse. Zu den erfassten Aufrufen gehören Aufrufe von der Amazon-EKS-Konsole und Code-Aufrufe der Amazon-EKS-API-Operationen. Weitere Informationen finden Sie unter the section called “ AWS CloudTrail”.

Der Kubernetes-API-Server stellt eine Reihe von Metriken zur Verfügung, die für die Überwachung und Analyse hilfreich sind. Weitere Informationen finden Sie unter the section called “Prometheus-Metriken”.

Informationen zur Konfiguration von Fluent Bit für benutzerdefinierte CloudWatch Amazon-Protokolle finden Sie unter Fluent Bit einrichten im CloudWatch Amazon-Benutzerhandbuch.

Überwachungs- und Protokollierungs-Tools für Amazon EKS

Amazon Web Services bietet verschiedene Tools, mit deren Hilfe Sie Amazon EKS überwachen können. Sie können einige Tools konfigurieren, um eine automatische Überwachung einzurichten, einige erfordern jedoch manuelle Anrufe. Wir empfehlen, dass Sie Überwachungsaufgaben so weit automatisieren, wie es Ihre Umgebung und Ihr vorhandenes Toolset zulassen.

Die folgende Tabelle beschreibt verschiedene Optionen für Überwachungs-Tools.

Bereiche	Tool	Description	Einrichtung
Steuerebene	Dashboard zur Beobachtbarkeit	Für unterstützte Versionen bietet das Dashboard zur Beobachtbarkeit Einblick in die Leistung Ihres Clusters. Es unterstützt Sie dabei, Probleme schnell zu erkennen, zu beheben und zu lösen.	Einrichtungsverfahren
Anwendungen/Steuerebene	Prometheus	Prometheus kann zur Überwachung von Metriken und Warnmeldungen für	Einrichtungsverfahren

Bereiche	Tool	Description	Einrichtung
		Anwendungen und die Steuerebene eingesetzt werden.	
Anwendungen	CloudWatch Container Insights	CloudWatch Container Insights sammelt, aggregiert und fasst Metriken und Protokolle aus Ihren containerisierten Anwendungen und Microservices zusammen.	Einrichtungsverfahren
Anwendungen	AWS Distribution für (ADOT) OpenTelemetry	ADOT kann korrelierte Metriken, Rückverfolgungsdaten und Metadaten sammeln und an AWS Überwachungsdienste oder Partner senden. Es kann über CloudWatch Container Insights eingerichtet werden.	Einrichtungsverfahren
Anwendungen	DevOpsAmazon-Guru	Amazon DevOps Guru erkennt die Betriebsleistung und Verfügbarkeit auf Knotenebene.	Einrichtungsverfahren

Bereiche	Tool	Description	Einrichtung
Anwendungen	AWS X-Ray	AWS X-Ray empfängt Trace-Daten über Ihre Anwendung. Diese Trace-Daten enthalten ein- und ausgehende Anfragen und Metadaten zu diesen. Für Amazon EKS ist für die Implementierung das OpenTelemetry Add-on erforderlich.	Einrichtungsverfahren
Anwendungen	Amazon CloudWatch	CloudWatch stellt einige grundlegende Amazon EKS-Metriken in unterstützten Versionen kostenlos zur Verfügung. Sie können diese Funktionalität mit dem CloudWatch Observability Operator erweitern, um das Sammeln von Metriken, Protokollen und Trace-Daten zu verwalten.	Einrichtungsverfahren

Die folgende Tabelle beschreibt verschiedene Optionen für Protokollierungs-Tools.

Bereiche	Tool	Description	Einrichtung
Steuerebene	Dashboard zur Beobachtbarkeit	Für unterstützte Versionen zeigt das Beobachtbarkeits-Dashboard Listen an, die auf den Audit-Protokollen der Steuerebe ne basieren. Es enthält auch Links zu Protokollen von Kontrollflugzeugen bei Amazon CloudWatch.	Einrichtungsverfahren
Anwendungen	Einblicke in Amazon CloudWatch Container	Amazon CloudWatc h Container Insights sammelt, aggregier t und fasst Metriken und Protokolle aus Ihren container isierten Anwendung en und Microservices zusammen.	Einrichtungsverfahren
Steuerebene	CloudWatch Amazon-Protokolle	Sie können Prüf- und Diagnosep rotokolle direkt von der Amazon EKS-Steuerebene an die CloudWatch Protokoll e in Ihrem Konto senden.	Einrichtungsverfahren
Steuerebene	AWS CloudTrail	Protokolliert API-Aufru fe eines Benutzers,	Einrichtungsverfahren

Bereiche	Tool	Description	Einrichtung
		einer Rolle oder eines Service.	
Mehrere Bereiche für AWS Fargate-Instanzen	AWS Fargate-Protokollrouter	Bei AWS Fargate-Instances streamt der Log-Router Logs an AWS Dienste oder Partner-Tools. Für Fluent Bit wird AWS verwendet. Protokolle können an andere AWS Dienste oder Partner-Tools gestreamt werden.	Einrichtungsverfahren

Überwachen Ihres Clusters im Dashboard „Beobachtbarkeit“

Die Amazon-EKS-Konsole enthält ein Beobachtbarkeits-Dashboard, das Einblick in die Leistung Ihres Clusters bietet. Die darin enthaltenen Informationen helfen Ihnen, Probleme schnell zu erkennen, zu beheben und zu lösen. Sie können den entsprechenden Abschnitt des Beobachtbarkeits-Dashboards öffnen, indem Sie einen Eintrag in der Zusammenfassung zu Zustand und Leistung auswählen. Diese Zusammenfassung ist an mehreren Stellen enthalten, darunter auch auf der Registerkarte Beobachtbarkeit.

Das Dashboard zur Beobachtbarkeit ist in mehrere Registerkarten unterteilt.

Zusammenfassung

In der Zusammenfassung zu Integrität und Leistung wird die Anzahl der Elemente in verschiedenen Kategorien aufgeführt. Jede Zahl dient als Hyperlink zu einer Stelle im Dashboard Beobachtbarkeit mit einer Liste für diese Kategorie.

Cluster-Integrität

Cluster-Integrität liefert wichtige Benachrichtigungen, die Sie beachten sollten. Bei einigen davon müssen Sie möglicherweise so schnell wie möglich Maßnahmen ergreifen. In dieser Liste finden

Sie Beschreibungen und die betroffenen Ressourcen. Cluster-Integrität besteht aus zwei Tabellen: Integritätsprobleme und Konfigurationserkenntnisse. Um den Status der Integritätsprobleme zu aktualisieren, klicken Sie auf die Schaltfläche „Aktualisieren“ (↻). Konfigurationserkenntnisse werden automatisch alle 24 Stunden aktualisiert und können nicht manuell aktualisiert werden

Weitere Informationen zu Integritätsprobleme finden Sie unter the section called “Häufig gestellte Fragen zur Cluster-Integrität und Fehlercodes mit Lösungspfaden”. Weitere Informationen zu Konfigurationserkenntnisse finden Sie unter the section called “Cluster-Einblicke”.

Überwachen der Steuerebene

Die Registerkarte Überwachung der Steuerebene ist in drei Abschnitte unterteilt, die Ihnen jeweils bei der Überwachung und Fehlerbehebung der Steuerebene Ihres Clusters behilflich sind.

Metriken

Für Cluster mit Kubernetes-Version `1.28` und höher werden im Abschnitt „Metriken“ Diagramme mit verschiedenen Metriken angezeigt, die für verschiedene Komponenten der Steuerebene erfasst wurden.

Sie können den Zeitbereich für die X-Achse jedes Diagramms festlegen, indem Sie oben im Abschnitt eine Auswahl treffen. Mit der Schaltfläche „Aktualisieren“ (↻) können Sie die Daten aktualisieren. Für jedes einzelne Diagramm öffnet die Schaltfläche mit den vertikalen Ellipsen () ein Menü mit Optionen von CloudWatch.

Diese und weitere Metriken sind automatisch als grundlegende Überwachungsmetriken CloudWatch unter dem `AWS/EKS` Namespace verfügbar. Weitere Informationen finden Sie unter Grundlegende Überwachung und detaillierte Überwachung im CloudWatch Amazon-Benutzerhandbuch. Ausführlichere Metriken, Visualisierungen und Einblicke finden Sie unter Container Insights im CloudWatch Amazon-Benutzerhandbuch. Wenn Sie eine Prometheus-basierte Überwachung bevorzugen, lesen Sie the section called “Prometheus-Metriken”.

In der folgenden Tabelle werden die verfügbaren Metriken beschrieben.

Metrik	Description
APIServer Anfragen	Die Anzahl der Anfragen pro Minute, die an den API-Server gestellt werden.

Metrik	Description
APIServer Gesamtzahl der Anfragen 4XX	Die Anzahl der API-Serveranfragen pro Minute, die HTTP-4XX-Antwortcodes (clientseitige Fehler) aufwiesen.
APIServer Gesamtzahl der Anfragen 5XX	Die Anzahl der API-Serveranfragen pro Minute, die HTTP-5XX-Antwortcodes (serverseitige Fehler) aufwiesen.
APIServer Gesamtzahl der Anfragen 429	Die Anzahl der API-Serveranfragen pro Minute mit HTTP-429-Antwortcodes (zu viele Anfragen).
Speichergröße	Die Größe der Speicherdatenbank (`etcd`).
Scheduler-Versuche	Die Anzahl der Versuche, Pods zu planen, nach den Ergebnissen „nicht planbar“, „Fehler“ und „geplant“.
Ausstehende Pods	Die Anzahl der ausstehenden Pods nach Warteschlangentyp „aktiv“, „Backoff“, „nicht planbar“ und „gesperrt“.
Latenzzeit bei API-Serveranfragen	Die Latenzzeit für API-Serveranfragen.
Aktuelle laufende Anfragen an den API-Server	Die aktuellen laufenden Anfragen für den API-Server.
Webhook-Anfragen	Die Webhook-Anfragen pro Minute.
Abgelehnte Webhook-Anfragen	Die Anzahl der abgelehnten Webhook-A nfragen.
Latenzzeit bei Webhook-Anfragen P99	Die 99. Perzentil-Latenzzeit von externen Webhook-Anfragen von Drittanbietern.

CloudWatch Einblicke protokollieren

Im Abschnitt CloudWatch Log Insights werden verschiedene Listen angezeigt, die auf den Auditprotokollen der Kontrollebene basieren. Die Amazon EKS-Steuerebenenprotokolle müssen aktiviert sein, um diese Funktion nutzen zu können. Dies können Sie im CloudWatch Abschnitt Kontrollebenenprotokolle anzeigen im tun.

Sobald ausreichend Zeit für die Datenerfassung vergangen ist, können Sie Alle Abfragen ausführen oder die Option Abfrage für eine einzelne Liste ausführen wählen. Jedes Mal, CloudWatch wenn Sie Abfragen ausführen, fallen zusätzliche Kosten an. Wählen Sie oben in diesem Abschnitt den Zeitraum aus, für den Sie die Ergebnisse anzeigen möchten. Wenn Sie eine erweiterte Steuerung für eine Abfrage wünschen, können Sie „Anzeigen in CloudWatch“ wählen. Auf diese Weise können Sie eine Abfrage CloudWatch an Ihre Bedürfnisse anpassen.

Weitere Informationen finden Sie unter Analysieren von Protokolldaten mit CloudWatch Logs Insights im Amazon CloudWatch Logs-Benutzerhandbuch.

Logs auf der Kontrollebene anzeigen CloudWatch

Wählen Sie Protokollierung verwalten, um die verfügbaren Protokolltypen zu aktualisieren. Nachdem Sie die Protokollierung aktiviert haben, dauert es einige Minuten, bis die CloudWatch Protokolle in den Protokollen angezeigt werden. Wenn ausreichend Zeit verstrichen ist, wählen Sie einen der Links Anzeigen in diesem Abschnitt, um zum entsprechenden Protokoll zu navigieren.

Weitere Informationen finden Sie unter the section called “Protokolle der Steuerebene”.

Cluster-Erkenntnisse

Die Tabelle Upgrade-Erkenntnisse zeigt sowohl Probleme auf und empfiehlt Korrekturmaßnahmen, wodurch der Validierungsprozess für das Upgrade auf neue Kubernetes-Versionen beschleunigt wird. Amazon EKS scannt Cluster automatisch anhand einer Liste potenzieller Probleme, die sich auf das Upgrade der Kubernetes-Version auswirken könnten. Die Tabelle Upgrade-Erkenntnisse listet die von Amazon EKS für diesen Cluster durchgeführten Erkenntnisprüfungen zusammen mit den zugehörigen Statusmeldungen auf.

Amazon EKS verwaltet und aktualisiert regelmäßig die Liste der Prüfungen, die auf der Grundlage von Bewertungen der Änderungen im Kubernetes-Projekt sowie der Amazon-EKS-Serviceänderungen im Zusammenhang mit neuen Versionen durchgeführt werden müssen. Die Amazon-EKS-Konsole aktualisiert automatisch den Status jeder Erkenntnis, der in der Spalte „Letzte Aktualisierungszeit“ angezeigt wird.

Weitere Informationen finden Sie unter the section called “Cluster-Einblicke”.

Probleme mit dem Zustand des Knotens

Der Amazon-EKS-Knotenüberwachungsagent liest automatisch Knotenprotokolle, um Zustandsprobleme zu erkennen. Unabhängig von der Einstellung für die automatische Reparatur werden alle Probleme mit der Knoten-Integrität gemeldet, sodass Sie diese bei Bedarf untersuchen können. Wenn ein Problemtyp ohne Beschreibung aufgeführt ist, finden Sie die Beschreibung in dem zugehörigen Popover-Element.

Wenn Sie die Seite aktualisieren, werden alle gelösten Probleme aus der Liste entfernt. Wenn die automatische Reparatur aktiviert ist, können vorübergehend einige Integritätsprobleme gemeldet werden, die ohne Ihr Eingreifen behoben werden. Probleme, die von der automatischen Reparatur nicht unterstützt werden, erfordern je nach Art möglicherweise manuelles Eingreifen.

Damit Probleme mit der Knotenintegrität gemeldet werden, muss Ihr Cluster Amazon EKS Auto Mode verwenden oder über das Add-On für den Knoten-Überwachungsagenten verfügen. Weitere Informationen finden Sie unter the section called “Knotenintegrität”.

EKS-Funktionen

Im Abschnitt Capabilities werden der Status und der Zustand Ihrer EKS-Capability-Ressourcen im Cluster angezeigt. Integritäts- und Statusbenachrichtigungen für beide Funktionen und ihre verwalteten Kubernetes-Ressourcen in Ihrem Cluster können hier überwacht werden. Wenn Sie die Seite aktualisieren, werden alle gelösten Probleme aus der Liste entfernt.

Weitere Informationen finden Sie unter the section called “Mit Fähigkeiten arbeiten”.

Überwachen Sie den Kubernetes-Workload-Traffic mit Container Network Observability

Amazon EKS bietet erweiterte Funktionen zur Netzwerkbeobachtbarkeit, die tiefere Einblicke in Ihre Container-Netzwerkumgebung bieten. Diese Funktionen helfen Ihnen dabei, Ihre Kubernetes-Netzwerklandschaft in besser zu verstehen, zu überwachen und Fehler zu beheben. AWS Dank der verbesserten Beobachtbarkeit von Container-Netzwerken können Sie detaillierte, netzwerkbezogene Metriken für eine bessere proaktive Erkennung von Anomalien im Cluster-Traffic, AZ-übergreifenden Flows und Diensten nutzen. AWS Mithilfe dieser Metriken können Sie die Systemleistung messen und die zugrunde liegenden Metriken mithilfe Ihres bevorzugten Observability-Stacks visualisieren.

Darüber hinaus bietet Amazon EKS jetzt Netzwerküberwachungsvisualisierungen in der AWS Konsole, die die präzise Fehlerbehebung beschleunigen und verbessern und so eine schnellere Ursachenanalyse ermöglichen. Sie können diese visuellen Funktionen auch nutzen, um Top-Talker und Netzwerkflüsse zu identifizieren, die zu erneuten Übertragungen und Timeouts bei der erneuten Übertragung führen, wodurch blinde Flecken bei Vorfällen beseitigt werden.

Diese Funktionen werden von Amazon CloudWatch Network Flow Monitor aktiviert.

Anwendungsfälle

Messen Sie die Netzwerkleistung, um Anomalien zu erkennen

Mehrere Teams verwenden standardmäßig einen Observability-Stack, der es ihnen ermöglicht, die Leistung ihres Systems zu messen, Systemmetriken zu visualisieren und bei Überschreitung eines bestimmten Schwellenwerts alarmiert zu werden. Die Beobachtbarkeit von Containernetzwerken in EKS passt genau darauf an, indem wichtige Systemmetriken offengelegt werden, die Sie auswerten können, um die Beobachtbarkeit der Netzwerkleistung Ihres Systems auf Pod- und Worker-Node-Ebene zu erweitern.

Nutzen Sie Konsolenvisualisierungen für eine genauere Fehlerbehebung

Falls Ihr Überwachungssystem einen Alarm ausgibt, sollten Sie sich auf den Cluster und die Arbeitslast konzentrieren, von denen das Problem herrührt. Um dies zu unterstützen, können Sie Visualisierungen in der EKS-Konsole nutzen, um den Untersuchungsumfang auf Clusterebene einzugrenzen und die Offenlegung der Netzwerkflüsse zu beschleunigen, die für die meisten Neuübertragungen, Zeitüberschreitungen bei der erneuten Übertragung und das übertragene Datenvolumen verantwortlich sind.

Verfolgen Sie Top-Talker in Ihrer Amazon EKS-Umgebung

Viele Teams verwenden EKS als Grundlage für ihre Plattformen und machen es somit zum Mittelpunkt der Netzwerkaktivitäten einer Anwendungsumgebung. Mithilfe der Netzwerküberwachungsfunktionen in dieser Funktion können Sie nachverfolgen, welche Workloads für den meisten Datenverkehr (gemessen am Datenvolumen) innerhalb des Clusters AZs, über und für den Verkehr zu externen Zielen innerhalb AWS (DynamoDB und S3) und außerhalb der AWS Cloud (Internet oder lokal) verantwortlich sind. Darüber hinaus können Sie die Leistung jedes dieser Datenflüsse anhand von Neuübertragungen, Zeitüberschreitungen bei der erneuten Übertragung und übertragenen Daten überwachen.

Features

1. Leistungsmetriken — Mit dieser Funktion können Sie netzwerkbezogene Systemmetriken für Pods und Worker-Knoten direkt aus dem Network Flow Monitor (NFM) -Agenten abrufen, der in Ihrem EKS-Cluster ausgeführt wird.
2. Service Map — Diese Funktion visualisiert dynamisch die Interkommunikation zwischen Workloads im Cluster, sodass Sie wichtige Kennzahlen (Retransmissions — RT, Retransmission Timeouts — RTO und übertragene Daten — DT) im Zusammenhang mit Netzwerkströmen zwischen kommunizierenden Pods schnell offenlegen können.
3. Flow-Tabelle — Mit dieser Tabelle können Sie die Top-Talker der Kubernetes-Workloads in Ihrem Cluster aus drei verschiedenen Blickwinkeln überwachen: Serviceansicht, Clusteransicht und externe Ansicht. AWS Für jede Ansicht können Sie die erneuten Übertragungen, die Timeouts für die erneute Übertragung und die zwischen dem Quell-Pod und seinem Ziel übertragenen Daten sehen.
   - AWS Serviceansicht: Zeigt die Top-Talker zu AWS Diensten an (DynamoDB und S3)
   - Cluster-Ansicht: Zeigt die Top-Talker innerhalb des Clusters an (von Ost ← nach → West)
   - Externe Ansicht: Zeigt die Top-Talker zu clusterexternen Zielen außerhalb AWS

Erste Schritte

Aktivieren Sie zunächst Container Network Observability in der EKS-Konsole für einen neuen oder vorhandenen Cluster. Dadurch wird die Erstellung von Network Flow Monitor (NFM) -Abhängigkeiten (Scope - und Monitor-Ressourcen) automatisiert. Darüber hinaus müssen Sie das Network Flow Monitor Agent-Add-on installieren. Alternativ können Sie diese Abhängigkeiten mithilfe von EKS APIs (für das `AWS CLI` Add-On), NFM APIs oder Infrastructure as Code (wie Terraform) installieren. Sobald diese Abhängigkeiten eingerichtet sind, können Sie Ihr bevorzugtes Überwachungstool so konfigurieren, dass Netzwerkleistungskennzahlen für Pods und Worker-Knoten vom NFM-Agenten abgerufen werden. Um die Netzwerkaktivität und Leistung Ihrer Workloads zu visualisieren, können Sie im Observability-Dashboard des Clusters unter der Registerkarte „Netzwerk“ zur EKS-Konsole navigieren.

Wenn Sie Network Flow Monitor in EKS verwenden, können Sie Ihren bestehenden Observability-Workflow und Ihren Technologie-Stack beibehalten und gleichzeitig eine Reihe zusätzlicher Funktionen nutzen, die es Ihnen ermöglichen, die Netzwerkschicht Ihrer EKS-Umgebung besser zu verstehen und zu optimieren. Weitere Informationen zu den Preisen von Network Flow Monitor finden Sie hier.

Voraussetzungen und wichtige Hinweise

1. Wie oben erwähnt, werden die zugrunde liegenden NFM-Ressourcenabhängigkeiten (Scope und Monitor) automatisch in Ihrem Namen erstellt, wenn Sie Container Network Observability von der EKS-Konsole aus aktivieren, und Sie werden durch den Installationsprozess des EKS-Add-ons für NFM geführt.
2. Wenn Sie diese Funktion mithilfe von Infrastructure as Code (IaC) wie Terraform aktivieren möchten, müssen Sie die folgenden Abhängigkeiten in Ihrem IaC definieren: NFM Scope, NFM Monitor, EKS-Add-on für NFM. Darüber hinaus müssen Sie dem EKS-Add-On mithilfe von Pod Identity - oder IAM-Rollen für Dienstkonten (IRSA) die entsprechenden Berechtigungen gewähren.
3. Sie müssen eine Mindestversion von 1.1.0 für das EKS-Add-on des NFM-Agenten ausführen.
4. Sie müssen Version 6.21.0 oder höher des Terraform AWS Providers verwenden, um die Network Flow Monitor-Ressourcen zu unterstützen.

Erforderliche IAM-Berechtigungen

EKS-Add-on für den NFM-Agenten

Sie können die `CloudWatchNetworkFlowMonitorAgentPublishPolicy` AWS verwaltete Richtlinie mit Pod Identity verwenden. Diese Richtlinie enthält Berechtigungen für den NFM-Agenten, Telemetrieberichte (Metriken) an einen Network Flow Monitor-Endpunkt zu senden.

```
{
  "Version" : "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "networkflowmonitor:Publish"
      ],
      "Resource" : "*"
    }
  ]
}
```

Container-Netzwerk-Observability in der EKS-Konsole

Die folgenden Berechtigungen sind erforderlich, um die Funktion zu aktivieren und die Service-Map und die Flow-Tabelle in der Konsole zu visualisieren.

```
{
  "Version" : "2012-10-17",
  "Statement" : [
    {
      "Effect": "Allow",
      "Action": [
        "networkflowmonitor:ListScopes",
        "networkflowmonitor:ListMonitors",
        "networkflowmonitor:GetScope",
        "networkflowmonitor:GetMonitor",
        "networkflowmonitor:CreateScope",
        "networkflowmonitor:CreateMonitor",
        "networkflowmonitor:TagResource",
        "networkflowmonitor:StartQueryMonitorTopContributors",
        "networkflowmonitor:StopQueryMonitorTopContributors",
        "networkflowmonitor:GetQueryStatusMonitorTopContributors",
        "networkflowmonitor:GetQueryResultsMonitorTopContributors"
      ],
      "Resource": "*"
    }
  ]
}
```

Verwenden von AWS CLI, EKS-API und NFM-API

```
#!/bin/bash

# Script to create required Network Flow Monitor resources
set -e

CLUSTER_NAME="my-eks-cluster"
CLUSTER_ARN="arn:aws:eks:{Region}:{Account}:cluster/{ClusterName}"
REGION="us-west-2"
AGENT_NAMESPACE="amazon-network-flow-monitor"

echo "Creating Network Flow Monitor resources..."

# Check if Network Flow Monitor agent is running in the cluster
echo "Checking for Network Flow Monitor agent in cluster..."
if kubectl get pods -n "$AGENT_NAMESPACE" --no-headers 2>/dev/null | grep -q "Running";
 then
    echo "Network Flow Monitor agent exists and is running in the cluster"
```

```
else
    echo "Network Flow Monitor agent not found. Installing as EKS addon..."
    aws eks create-addon \
        --cluster-name "$CLUSTER_NAME" \
        --addon-name "$AGENT_NAMESPACE" \
        --region "$REGION"
    echo "Network Flow Monitor addon installation initiated"
fi

# Get Account ID
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)

echo "Cluster ARN: $CLUSTER_ARN"
echo "Account ID: $ACCOUNT_ID"

# Check for existing scope
echo "Checking for existing Network Flow Monitor Scope..."
EXISTING_SCOPE=$(aws networkflowmonitor list-scopes --region $REGION --query
 'scopes[0].scopeArn' --output text 2>/dev/null || echo "None")

if [ "$EXISTING_SCOPE" != "None" ] && [ "$EXISTING_SCOPE" != "null" ]; then
    echo "Using existing scope: $EXISTING_SCOPE"
    SCOPE_ARN=$EXISTING_SCOPE
else
    echo "Creating new Network Flow Monitor Scope..."
    SCOPE_RESPONSE=$(aws networkflowmonitor create-scope \
        --targets "[{\"targetIdentifier\":{\"targetId\":{\"accountId\":
\"${ACCOUNT_ID}\"},\"targetType\":\"ACCOUNT\"},\"region\":\"${REGION}\"}]" \
        --region $REGION \
        --output json)

    SCOPE_ARN=$(echo $SCOPE_RESPONSE | jq -r '.scopeArn')
    echo "Scope created: $SCOPE_ARN"
fi

# Create Network Flow Monitor with EKS Cluster as local resource
echo "Creating Network Flow Monitor..."
MONITOR_RESPONSE=$(aws networkflowmonitor create-monitor \
    --monitor-name "${CLUSTER_NAME}-monitor" \
    --local-resources "type=AWS::EKS::Cluster,identifier=${CLUSTER_ARN}" \
    --scope-arn "$SCOPE_ARN" \
    --region $REGION \
    --output json)
```

```
MONITOR_ARN=$(echo $MONITOR_RESPONSE | jq -r '.monitorArn')

echo "Monitor created: $MONITOR_ARN"

echo "Network Flow Monitor setup complete!"
echo "Monitor ARN: $MONITOR_ARN"
echo "Scope ARN: $SCOPE_ARN"
echo "Local Resource: AWS::EKS::Cluster (${CLUSTER_ARN})"
```

Verwendung von Infrastruktur als Code (IaC)

Terraform

Wenn Sie Terraform zur Verwaltung Ihrer AWS Cloud-Infrastruktur verwenden, können Sie die folgenden Ressourcenkonfigurationen einbeziehen, um Container Network Observability für Ihren Cluster zu aktivieren.

NFM-Geltungsbereich

```
data "aws_caller_identity" "current" {}

resource "aws_networkflowmonitor_scope" "example" {
  target {
    region = "us-east-1"
    target_identifier {
      target_type = "ACCOUNT"
      target_id {
        account_id = data.aws_caller_identity.current.account_id
      }
    }
  }

  tags = {
    Name = "example"
  }
}
```

NFM-Monitor

```
resource "aws_networkflowmonitor_monitor" "example" {
  monitor_name = "eks-cluster-name-monitor"
```

```
  scope_arn    = aws_networkflowmonitor_scope.example.scope_arn

  local_resource {
    type       = "AWS::EKS::Cluster"
    identifier = aws_eks_cluster.example.arn
  }

  remote_resource {
    type       = "AWS::Region"
    identifier = "us-east-1" # this must be the same region that the cluster is in
  }

  tags = {
    Name = "example"
  }
}
```

EKS-Add-on für NFM

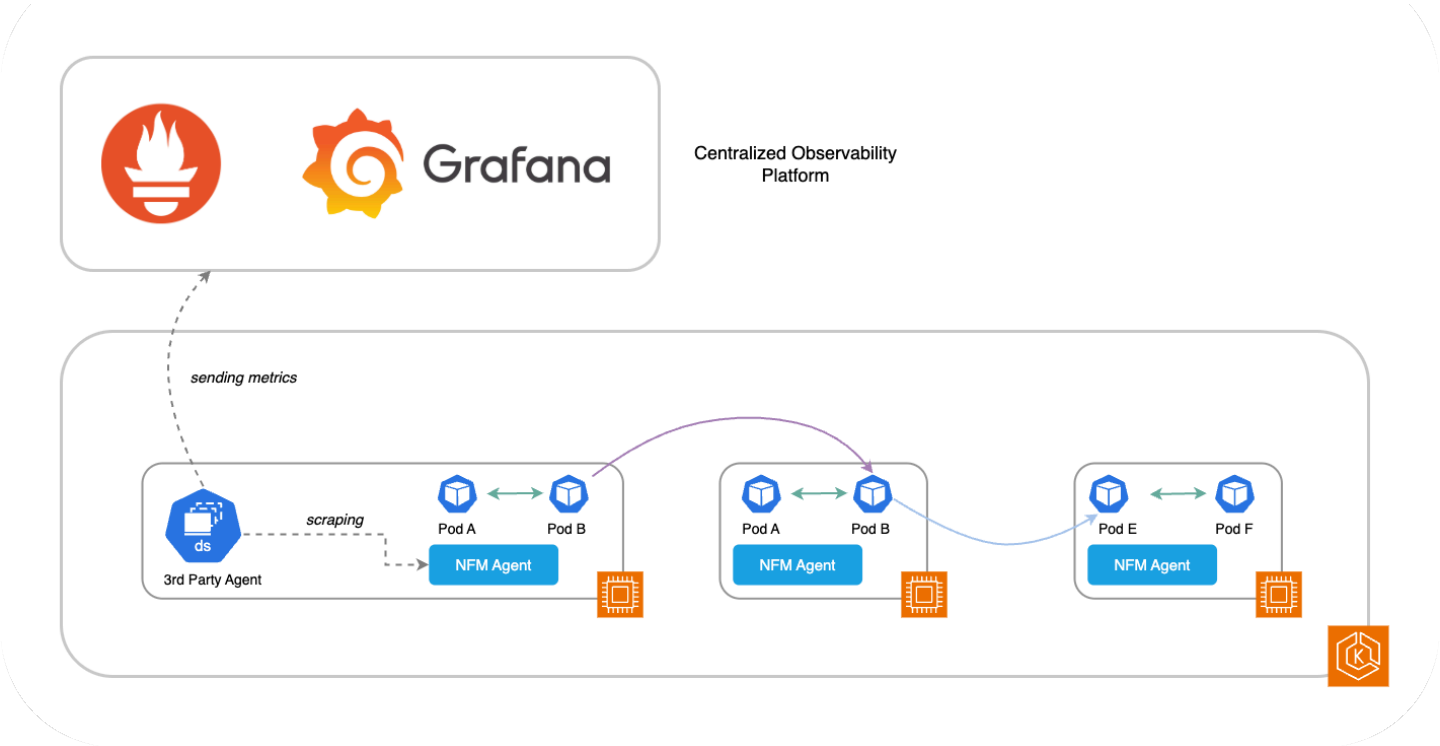
```
resource "aws_eks_addon" "example" {
  cluster_name                = aws_eks_cluster.example.name
  addon_name                  = "aws-network-flow-monitoring-agent"
}
```

Funktionsweise

Leistungsmetriken

Systemmetriken

Wenn Sie Drittanbieter-Tools (3P) zur Überwachung Ihrer EKS-Umgebung (wie Prometheus und Grafana) verwenden, können Sie die unterstützten Systemmetriken direkt vom Network Flow Monitor-Agenten abrufen. Diese Metriken können an Ihren Monitoring-Stack gesendet werden, um die Messung der Netzwerkleistung Ihres Systems auf Pod- und Worker-Node-Ebene zu erweitern. Die verfügbaren Metriken sind in der Tabelle unter Unterstützte Systemmetriken aufgeführt.



Um diese Metriken zu aktivieren, überschreiben Sie während des Installationsvorgangs die folgenden Umgebungsvariablen mithilfe der Konfigurationsvariablen (siehe: https://aws.amazon.com/blogs/amazon-eks-add-ons containers/ -advanced-configuration/):

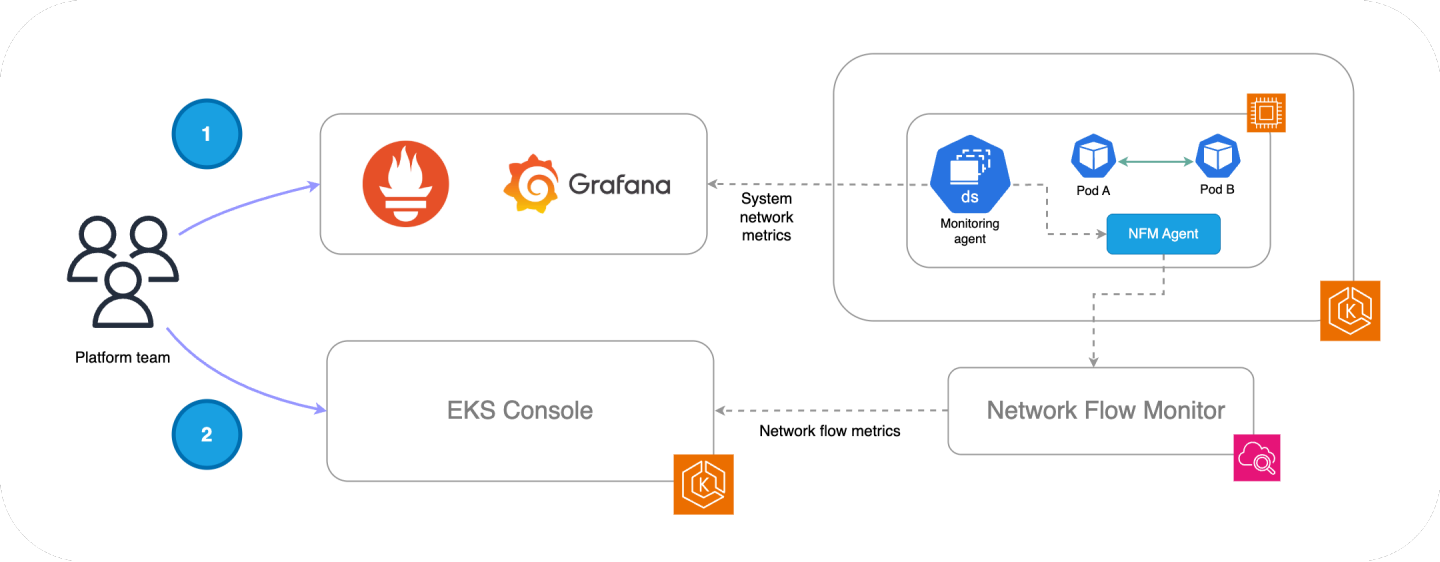
```
OPEN_METRICS:
    Enable or disable open metrics. Disabled if not supplied
    Type: String
    Values: [“on”, “off”]
OPEN_METRICS_ADDRESS:
    Listening IP address for open metrics endpoint. Defaults to 127.0.0.1 if not
 supplied
    Type: String
OPEN_METRICS_PORT:
    Listening port for open metrics endpoint. Defaults to 80 if not supplied
    Type: Integer
    Range: [0..65535]
```

Metriken auf Flussebene

Darüber hinaus erfasst Network Flow Monitor Netzwerkflussdaten zusammen mit Metriken auf Flussebene: Neuübertragungen, Zeitüberschreitungen bei der erneuten Übertragung und übertragene Daten. Diese Daten werden von Network Flow Monitor verarbeitet und in der EKS-

Konsole visualisiert, um den Datenverkehr in der Umgebung Ihres Clusters zu ermitteln und anhand dieser Metriken auf Flussebene zu ermitteln, wie er abschneidet.

Das folgende Diagramm zeigt einen Workflow, in dem beide Arten von Metriken (System- und Flow-Ebene) genutzt werden können, um mehr betriebliche Informationen zu gewinnen.



1. Das Plattformteam kann Systemmetriken in seinem Monitoring-Stack sammeln und visualisieren. Wenn Warnmeldungen vorhanden sind, können sie mithilfe der Systemmetriken des NFM-Agenten Netzwerkanomalien oder Probleme erkennen, die sich auf Pods oder Worker-Knoten auswirken.
2. Als nächsten Schritt können Plattformteams die nativen Visualisierungen in der EKS-Konsole nutzen, um den Untersuchungsbereich weiter einzugrenzen und die Fehlerbehebung auf der Grundlage von Ablaufdarstellungen und den zugehörigen Metriken zu beschleunigen.

Wichtiger Hinweis: Das Scraping von Systemmetriken aus dem NFM-Agenten und der Prozess, bei dem der NFM-Agent Metriken auf Flow-Ebene an das NFM-Backend überträgt, sind unabhängige Prozesse.

Unterstützte Systemmetriken

Wichtiger Hinweis: Systemmetriken werden im OpenMetricsFormat exportiert.

Metrikname	Typ	Dimensionen	Description
ingress_flow	Messinstrument	instance_id, iface, pod, namespace, node	Anzahl eingehender TCP-Datenflüsse () TcpPassiveOpens
Ausgangsfluss	Messinstrument	instance_id, iface, pod, namespace, node	Anzahl der ausgehenden TCP-Datenflüsse () TcpActiveOpens
Eingangspakete	Messinstrument	instance_id, iface, pod, namespace, node	Anzahl eingehender Pakete (Delta)
ausgehende Pakete	Messinstrument	instance_id, iface, pod, namespace, node	Anzahl der ausgehenden Pakete (Delta)
Ingress_Bytes	Messinstrument	instance_id, iface, pod, namespace, node	Anzahl der eingehend en Byte (Delta)
Ausgangsbytes	Messinstrument	instance_id, iface, pod, namespace, node	Anzahl der ausgehenden Byte (Delta)
bw_in_allowance_ex ceeded	Messinstrument	instance_id, eni, Knoten	Pakete queued/dr opped aufgrund der Bandbreitenbeschrä nkung für eingehende Nachrichten
bw_out_allowance_e xceeded	Messinstrument	instance_id, eni, Knoten	Pakete queued/dr opped aufgrund der Bandbreitenbeschrä nkung für ausgehende Nachrichten

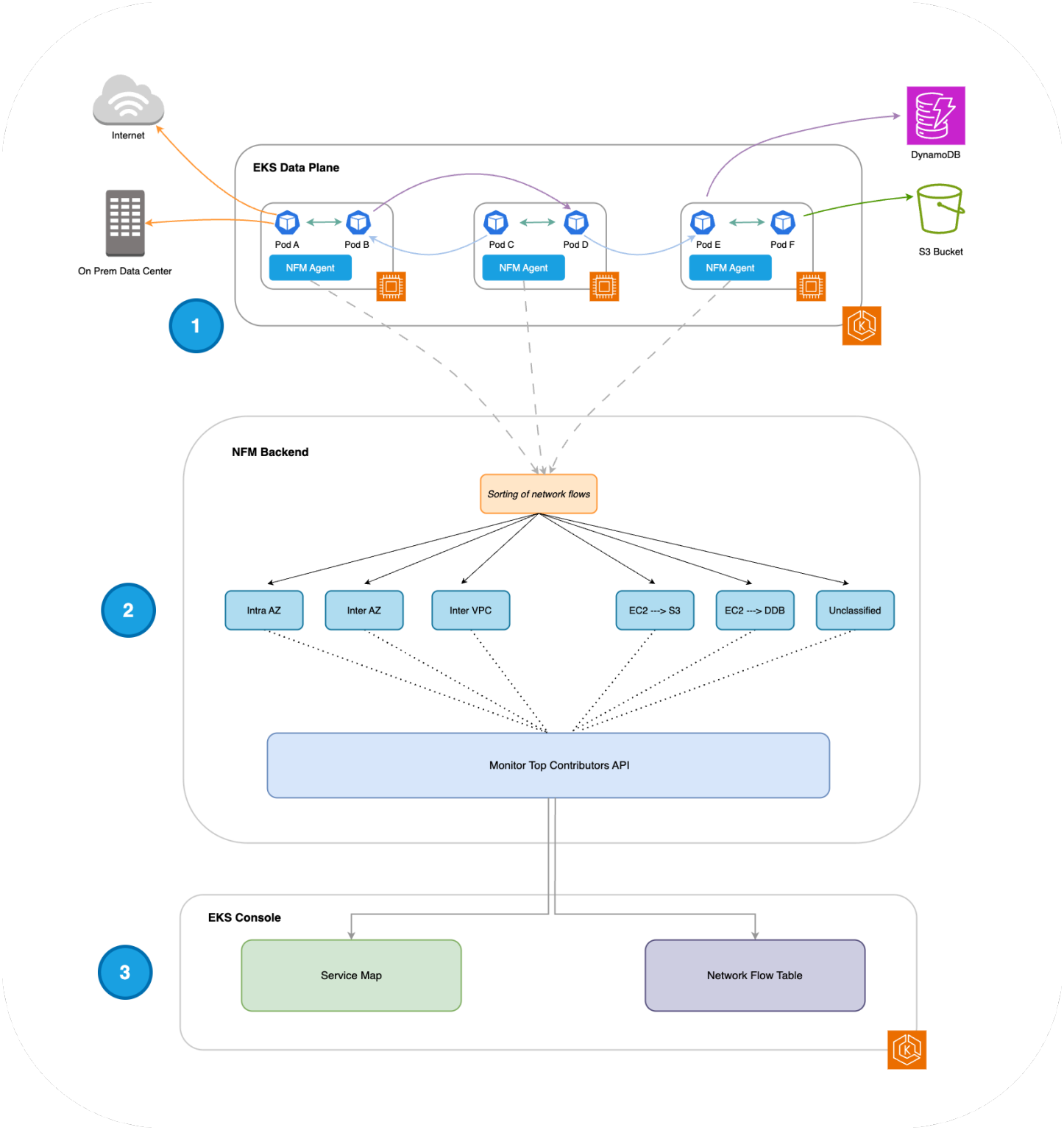
Metrikname	Typ	Dimensionen	Description
pps_allowance_exceeded	Messinstrument	instance_id, eni, Knoten	Pakete queued/dropped aufgrund des bidirektionalen PPS-Limits
conntrack_allowance_exceeded	Messinstrument	instance_id, eni, Knoten	Pakete wurden aufgrund des Limits für die Verbindungsverfolgung verworfen
linklocal_allowance_exceeded	Messinstrument	instance_id, eni, Knoten	Pakete wurden aufgrund des PPS-Limits für den lokalen Proxydienst verworfen

Unterstützte Metriken auf Flussebene

Metrikname	Typ	Description
TCP-Neuübertragungen	Zähler	Gibt an, wie oft ein Absender ein Paket erneut sendet, das während der Übertragung verloren gegangen oder beschädigt wurde.
Timeouts bei der TCP-Neuübertragung	Zähler	Gibt an, wie oft ein Absender eine Wartezeit eingeleitet hat, um festzustellen, ob ein Paket während der Übertragung verloren gegangen ist.
Übertragene Daten (Byte)	Zähler	Datenmenge, die für einen bestimmten Datenfluss

Metrikname	Typ	Description
		zwischen einer Quelle und einem Ziel übertragen wird.

Servicekarte und Datenflusstabelle

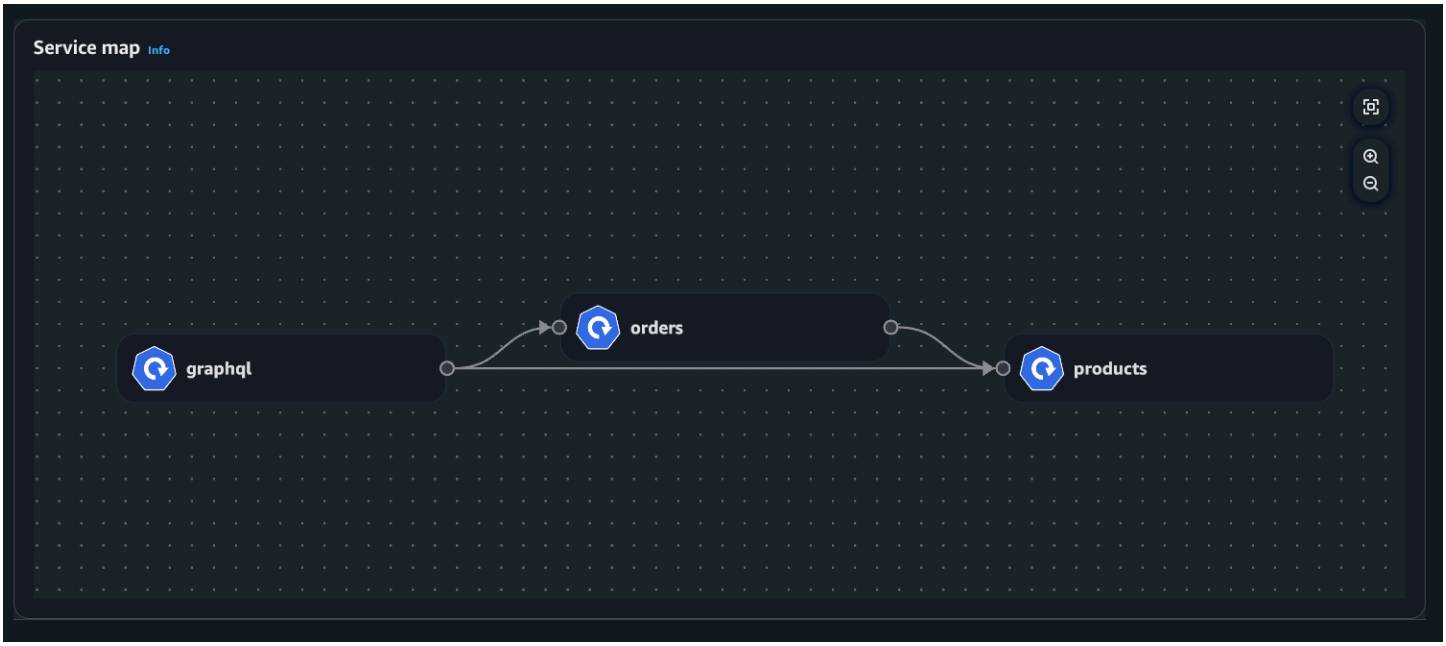


1. Nach der Installation wird der Network Flow Monitor-Agent DaemonSet auf jedem Worker-Node ausgeführt und erfasst alle 30 Sekunden die 500 wichtigsten Netzwerkflüsse (basierend auf dem übertragenen Datenvolumen).
2. Diese Netzwerkflüsse sind in die folgenden Kategorien unterteilt: Intra AZ, Inter AZ, EC2 → S3, EC2 → DynamoDB (DDB) und Unclassified. Jedem Datenfluss sind drei Metriken zugeordnet: Neuübertragungen, Zeitüberschreitungen bei der erneuten Übertragung und übertragene Daten (in Byte).
   - Intra AZ — Netzwerkflüsse zwischen Pods in derselben AZ
   - Inter AZ — Netzwerkflüsse zwischen Pods in verschiedenen AZs
   - EC2 → S3 - Netzwerkflüsse von Pods zu S3
   - EC2 → DDB — Netzwerkflüsse von Pods zu DDB
   - Nicht klassifiziert — Netzwerkflüsse von Pods ins Internet oder vor Ort
3. Netzwerkflüsse aus der Network Flow Monitor Top Contributors API werden verwendet, um die folgenden Funktionen in der EKS-Konsole bereitzustellen:
   - Service-Map: Visualisierung der Netzwerkflüsse innerhalb des Clusters (Intra AZ und Inter AZ).
   - Flow-Tabelle: Tabellarische Darstellung der Netzwerkflüsse innerhalb des Clusters (Intra AZ und Inter AZ), von Pods zu AWS Diensten (EC2 → S3 und EC2 → DDB) und von Pods zu externen Zielen (nicht klassifiziert).

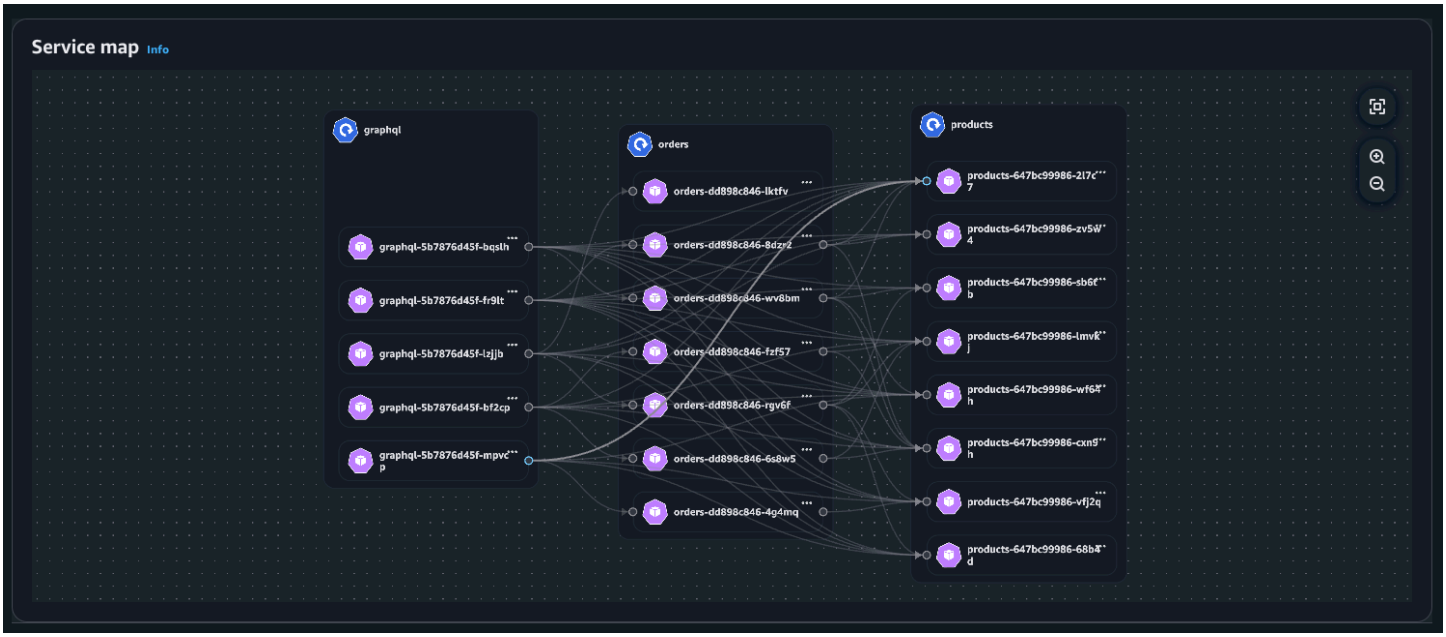
Die von der Top Contributors API abgerufenen Netzwerkflüsse sind auf einen Zeitraum von 1 Stunde begrenzt und können bis zu 500 Datenflüsse aus jeder Kategorie umfassen. Für die Service Map bedeutet dies, dass bis zu 1000 Datenflüsse aus den Datenflusskategorien Intra AZ und Inter AZ über einen Zeitraum von 1 Stunde abgerufen und dargestellt werden können. Für die Flow-Tabelle bedeutet dies, dass bis zu 3000 Netzwerkflüsse aus allen 6 Netzwerkflusskategorien über einen Zeitraum von 2 Stunden abgerufen und dargestellt werden können.

Beispiel: Serviceübersicht

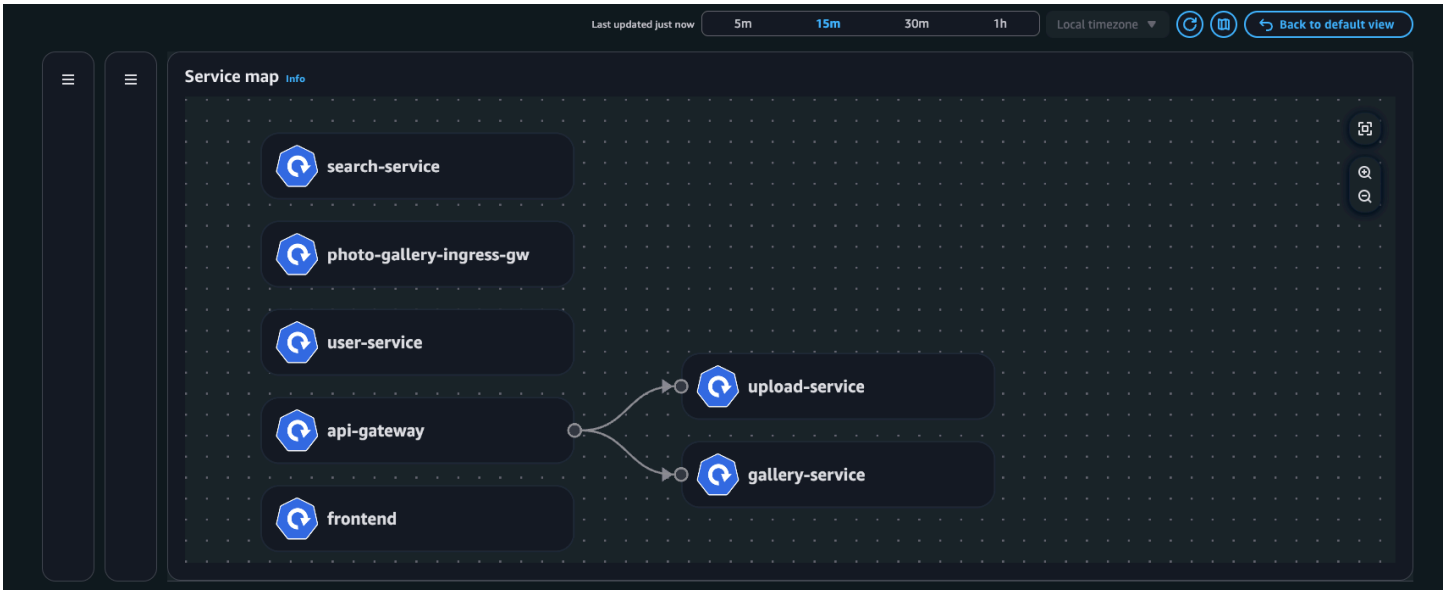
Bereitstellungsansicht



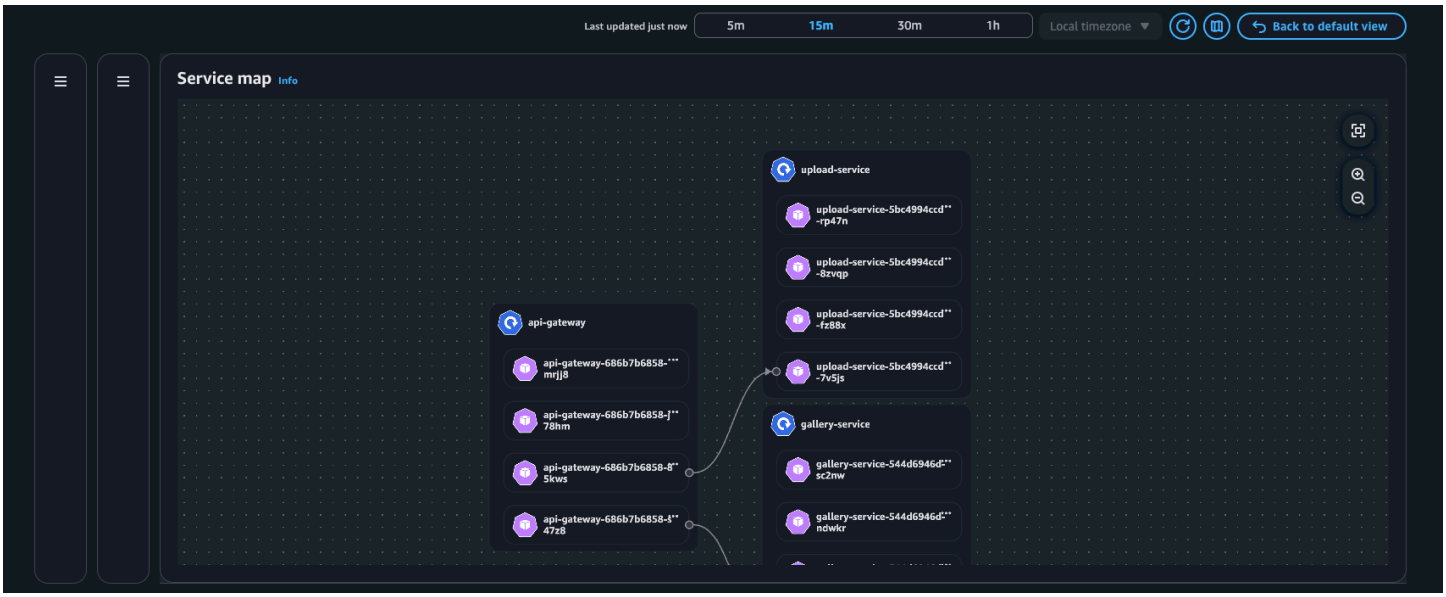
Pod-Ansicht



Bereitstellungsansicht

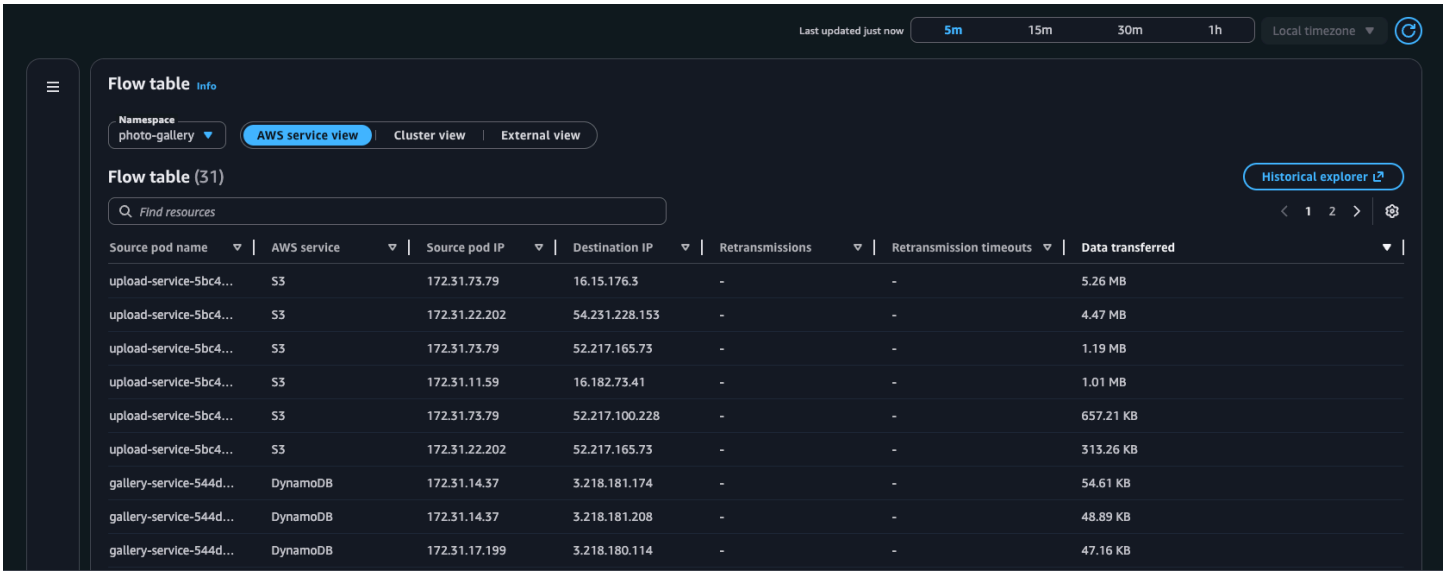


Pod-Ansicht

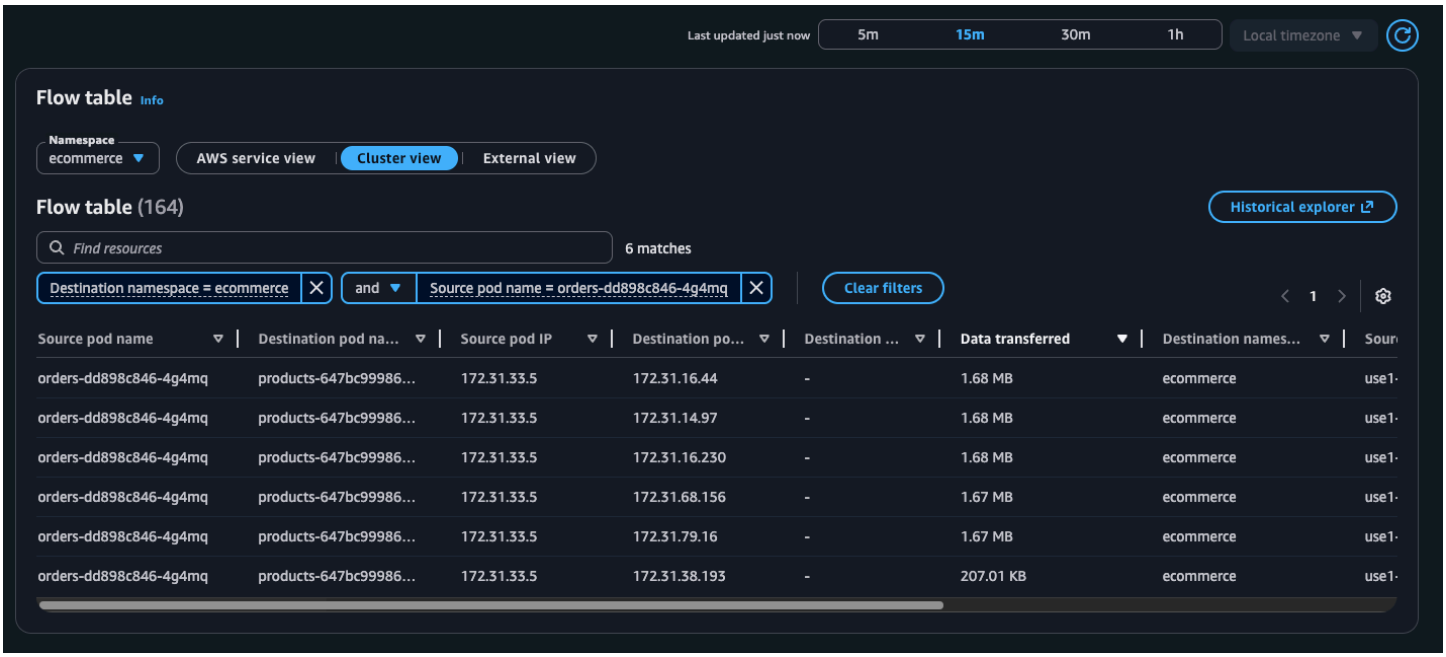


Beispiel: Flow-Tabelle

AWS Serviceansicht



Cluster-Ansicht



Überlegungen und Einschränkungen

- Container Network Observability in EKS ist nur in Regionen verfügbar, in denen Network Flow Monitor unterstützt wird.
- Unterstützte Systemmetriken sind OpenMetrics formatiert und können direkt vom Network Flow Monitor (NFM) -Agenten abgerufen werden.

- Um Container Network Observability in EKS mithilfe von Infrastructure as Code (IaC) wie Terraform zu aktivieren, müssen Sie die folgenden Abhängigkeiten in Ihren Konfigurationen definiert und erstellt haben: NFM-Bereich, NFM-Monitor und NFM-Agent.
- Network Flow Monitor unterstützt bis zu etwa 5 Millionen Datenflüsse pro Minute. Das sind ungefähr 5.000 EC2 Instanzen (EKS-Worker-Knoten), auf denen der Network Flow Monitor-Agent installiert ist. Die Installation von Agenten auf mehr als 5000 Instanzen kann die Überwachungsleistung beeinträchtigen, bis zusätzliche Kapazität verfügbar ist.
- Sie müssen eine Mindestversion von 1.1.0 für das EKS-Add-on des NFM-Agenten ausführen.
- Sie müssen Version 6.21.0 oder höher des Terraform AWS Providers verwenden, um die Network Flow Monitor-Ressourcen zu unterstützen.
- Um die Netzwerkflüsse mit Pod-Metadaten anzureichern, sollten Ihre Pods in ihrem eigenen isolierten Netzwerk-Namespace ausgeführt werden, nicht im Host-Netzwerk-Namespace.

Überwachung Ihrer Cluster-Metriken mit Prometheus

Prometheus ist eine Überwachungs- und Zeitreihendatenbank, die Endpunkte durchsucht. Sie bietet die Möglichkeit, gesammelte Daten abzufragen, zu aggregieren und zu speichern. Sie können sie auch für Warnungen und für die Aggregation von Warnungen verwenden. In diesem Thema wird erläutert, wie Sie Prometheus entweder als verwaltete oder als Open-Source-Option einrichten. Die Überwachung der Metriken der Amazon-EKS-Steuerebene ist ein häufiger Anwendungsfall.

Amazon Managed Service for Prometheus ist ein Prometheus-kompatibler Überwachungs- und Warndienst, der es einfach macht, containerisierte Anwendungen und Infrastruktur im großen Maßstab zu überwachen. Es ist ein vollständig verwalteter Service, der die Aufnahme, Speicherung, Abfrage und Warnung Ihrer Metriken automatisch skaliert. Es lässt sich auch in AWS-Sicherheits-Services integrieren, um einen schnellen und sicheren Zugriff auf Ihre Daten zu ermöglichen. Sie können die Open-Source-PromQL-Abfragesprache verwenden, um Ihre Metriken abzufragen und darauf zu warnen. Außerdem können Sie den Alarmmanager im Amazon Managed Service für Prometheus verwenden, um Warnregeln für kritische Alarme einzurichten. Diese kritischen Warnmeldungen können Sie dann als Benachrichtigungen an ein Amazon-SNS-Thema senden.

Es gibt mehrere verschiedene Möglichkeiten, Prometheus mit Amazon EKS zu verwenden:

- Sie können die Prometheus-Metriken beim erstmaligen Erstellen eines Amazon-EKS-Clusters aktivieren oder einen eigenen Prometheus-Scraper für vorhandene Cluster erstellen. Beide Optionen werden in diesem Thema behandelt.

- Sie können Prometheus mithilfe von Helm bereitstellen. Weitere Informationen finden Sie unter the section called “Bereitstellung mit Helm”.
- Sie können die Rohdaten der Steuerebene im Prometheus-Format anzeigen. Weitere Informationen finden Sie unter the section called “Steuerebene”.

Schritt 1: Prometheus-Metriken aktivieren

> **Important**
>
> Amazon Managed Service für Prometheus-Ressourcen liegen außerhalb des Cluster-Lebenszyklus und müssen unabhängig vom Cluster verwaltet werden. Wenn Sie Ihren Cluster löschen, löschen Sie auch alle relevanten Scraper, um damit verbundenen Kosten zu vermeiden. Weitere Informationen finden Sie unter Suchen und Löschen eine Scrapers im Benutzerhandbuch von Amazon Managed Service für Prometheus.

Prometheus entdeckt und erfasst Metriken aus Ihrem Cluster mithilfe eines Abruf-basierten Modells, das als Scraping bezeichnet wird. Scraper sind dafür eingerichtet, Daten aus Ihrer Cluster-Infrastruktur und containerisierten Anwendungen zu sammeln. Wenn Sie die Option zum Senden von Prometheus-Metriken aktivieren, stellt Amazon Managed Service for Prometheus einen vollständig verwalteten, agentenlosen Scraper zur Verfügung.

Falls Sie den Cluster noch nicht erstellt haben, können Sie die Option zum Senden von Metriken an Prometheus bei der erstmaligen Erstellung des Clusters aktivieren. In der Amazon-EKS-Konsole finden Sie diese Option im Schritt Beobachtbarkeit konfigurieren bei der Erstellung eines neuen Clusters. Weitere Informationen finden Sie unter the section called “Erstellen eines Clusters”.

Wenn Sie bereits über einen vorhandenen Cluster verfügen, können Sie Ihren eigenen Prometheus-Scraper erstellen. Navigieren Sie dazu in der Amazon-EKS-Konsole zur Registerkarte Beobachtbarkeit Ihres Clusters und wählen Sie die Schaltfläche Scraper hinzufügen. Wenn Sie dies lieber mit der AWS-API oder AWS CLI tun möchten, finden Sie weitere Informationen unter Erstellen eines Scrapers im Benutzerhandbuch für Amazon Managed Service für Prometheus.

Beim Erstellen des Scrapers mit der Amazon-EKS-Konsole stehen die folgenden Optionen zur Verfügung.

Scraper-Alias

(Optional) Geben Sie einen eindeutigen Alias für den Scraper ein.

Zieladresse

Wählen Sie einen Workspace für Amazon Managed Service für Prometheus aus. Ein Workspace ist ein logischer Bereich, der der Speicherung und Abfrage von Prometheus-Metriken gewidmet ist. Mit diesem Arbeitsbereich können Sie Prometheus-Metriken für alle Konten anzeigen, die darauf zugreifen können. Die Option Neuen Workspace erstellen weist Amazon EKS an, unter Verwendung des von Ihnen angegebenen Workspace-Alias einen Workspace in Ihrem Namen zu erstellen. Mit der Option Bestehenden Workspace auswählen können Sie einen vorhandenen Workspace aus einer Dropdown-Liste auswählen. Weitere Informationen zu Workspaces finden Sie unter Workspaces verwalten im Benutzerhandbuch von Amazon Managed Service für Prometheus.

Service-Zugriff

Dieser Abschnitt fasst die Berechtigungen zusammen, die Sie beim Senden von Prometheus-Metriken gewähren:

- Zulassen, dass Amazon Managed Service für Prometheus den gescrapten Amazon-EKS-Cluster beschreibt
- Schreiben aus der Ferne in den Amazon-Managed-Prometheus-Arbeitsbereich zulassen

Wenn `AmazonManagedScraperRole` bereits vorhanden ist, verwendet der Scraper diese Rolle. Wählen Sie den `AmazonManagedScraperRole`-Link aus, um die Berechtigungsdetails anzuzeigen. Wenn `AmazonManagedScraperRole` noch nicht vorhanden ist, wählen Sie den Link Berechtigungsdetails anzeigen, um die spezifischen Berechtigungen anzuzeigen, die Sie durch das Senden von Prometheus-Metriken gewähren.

Subnets

Ändern Sie die Subnetze, die der Scraper übernehmen soll, nach Bedarf. Wenn Sie eine ausgegraute Subnetzoption hinzufügen müssen, kehren Sie zum Schritt der Cluster-Erstellung unter Netzwerk angeben zurück.

Scraper-Konfiguration

Ändern Sie die Scraper-Konfiguration nach Bedarf im YAML-Format. Verwenden Sie dazu das Formular oder laden Sie eine YAML-Ersatzdatei hoch. Weitere Informationen finden Sie unter Scraper-Konfiguration im Benutzerhandbuch von Amazon Managed Service für Prometheus.

Amazon Managed Service für Prometheus bezieht sich auf den agentenlosen Scraper, der zusammen mit dem Cluster als verwalteter AWS-Sammler erstellt wird. Weitere Informationen zu von AWS verwalteten Kollektoren finden Sie unter Erfassen von Metriken mit von AWS verwalteten Kollektoren im Benutzerhandbuch zu Amazon Managed Service für Prometheus.

Important

- Wenn Sie einen Prometheus-Scraper mithilfe der AWS CLI oder AWS-API erstellen, müssen Sie seine Konfiguration anpassen, um dem Scraper Berechtigungen im Cluster zu gewähren. Weitere Informationen finden Sie unter Konfigurieren Ihres Amazon-EKS-Clusters im Benutzerhandbuch von Amazon Managed Service für Prometheus.
- Wenn Sie einen Prometheus-Scraper haben, der vor dem 11. November 2024 erstellt wurde und `aws-auth ConfigMap` anstelle von Zugriffseinträgen verwendet, müssen Sie ihn aktualisieren, um auf zusätzliche Metriken aus der Amazon-EKS-Cluster-Steuerebene zugreifen zu können. Informationen zur aktualisierten Konfiguration finden Sie unter Manuelle Konfiguration von Amazon EKS für Scraper-Zugriff im Benutzerhandbuch zu Amazon Managed Service für Prometheus.

Schritt 2: Prometheus-Metriken verwenden

Weitere Informationen zur Verwendung der Prometheus-Metriken, nachdem Sie sie für Ihren Cluster aktiviert haben, finden Sie im Benutzerhandbuch für Amazon Managed Service für Prometheus.

Schritt 3: Prometheus-Scraper verwalten

Wählen Sie zur Verwaltung von Scrapern in der Amazon-EKS-Konsole die Registerkarte Beobachtbarkeit aus. Eine Tabelle zeigt eine Liste der Scraper für den Cluster, einschließlich Informationen wie Scraper-ID, Alias, Status und Erstellungsdatum. Sie können weitere Scraper hinzufügen, Scraper bearbeiten, Scraper löschen oder weitere Informationen zu den aktuellen Scrapern anzeigen.

Um weitere Details zu einem Scraper anzuzeigen, wählen Sie den Scraper-ID-Link. Sie können beispielsweise ARN, Umgebung, Arbeitsbereichs-ID, IAM-Rolle, Konfiguration und Netzwerkinformationen anzeigen. Sie können die Scraper-ID als Eingabe für Operationen in der Amazon-Managed-Service-für-Prometheus-API wie `DescribeScraper`, `UpdateScraper` und `DeleteScraper` verwenden. Weitere Informationen zur Verwendung der Prometheus-API finden Sie in der API-Referenz zu Amazon Managed Service für Prometheus.

Prometheus mithilfe von Helm bereitstellen

Als Alternative zur Verwendung von Amazon Managed Service für Prometheus können Sie Prometheus mit Helm in Ihrem Cluster bereitstellen. Wenn Sie Helm bereits installiert haben, können Sie Ihre Version mit dem `helm version`-Befehl überprüfen. Helm ist ein Paket-Manager für Kubernetes-Cluster. Weitere Informationen zu Helm und zur Installation finden Sie unter the section called “Apps mit Helm bereitstellen”.

Nachdem Sie Helm für Ihren Amazon-EKS-Cluster konfiguriert haben, können Sie damit die folgenden Schritte ausführen, um Prometheus bereitzustellen.

1. Erstellen Sie einen Prometheus-Namespace.

```
kubectl create namespace prometheus
```

2. Fügen Sie das `prometheus-community`-Diagramm-Repository hinzu.

```
helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
```

3. Stellen Sie Prometheus bereit.

```
helm upgrade -i prometheus prometheus-community/prometheus \
    --namespace prometheus \
    --set alertmanager.persistence.storageClass="gp2" \
    --set server.persistentVolume.storageClass="gp2"
```

> **Note**
>
> Wenn Sie bei der Ausführung dieses Befehls die Fehlermeldung `Error: failed to download "stable/prometheus" (hint: running helm repo update may help)` erhalten, führen Sie `helm repo update prometheus-community` aus, und versuchen Sie dann erneut, den Befehl aus Schritt 2 auszuführen.

Wenn Sie die Fehlermeldung `Error: rendered manifests contain a resource that already exists` erhalten, führen Sie `helm uninstall` *`your-release-name`* `-n` *`namespace`* aus, und versuchen Sie dann, den Befehl aus Schritt 3 erneut auszuführen.

4. Stellen Sie sicher, dass alle Pods im `prometheus`-Namespace den Status `READY` haben.

```
kubectl get pods -n prometheus
```

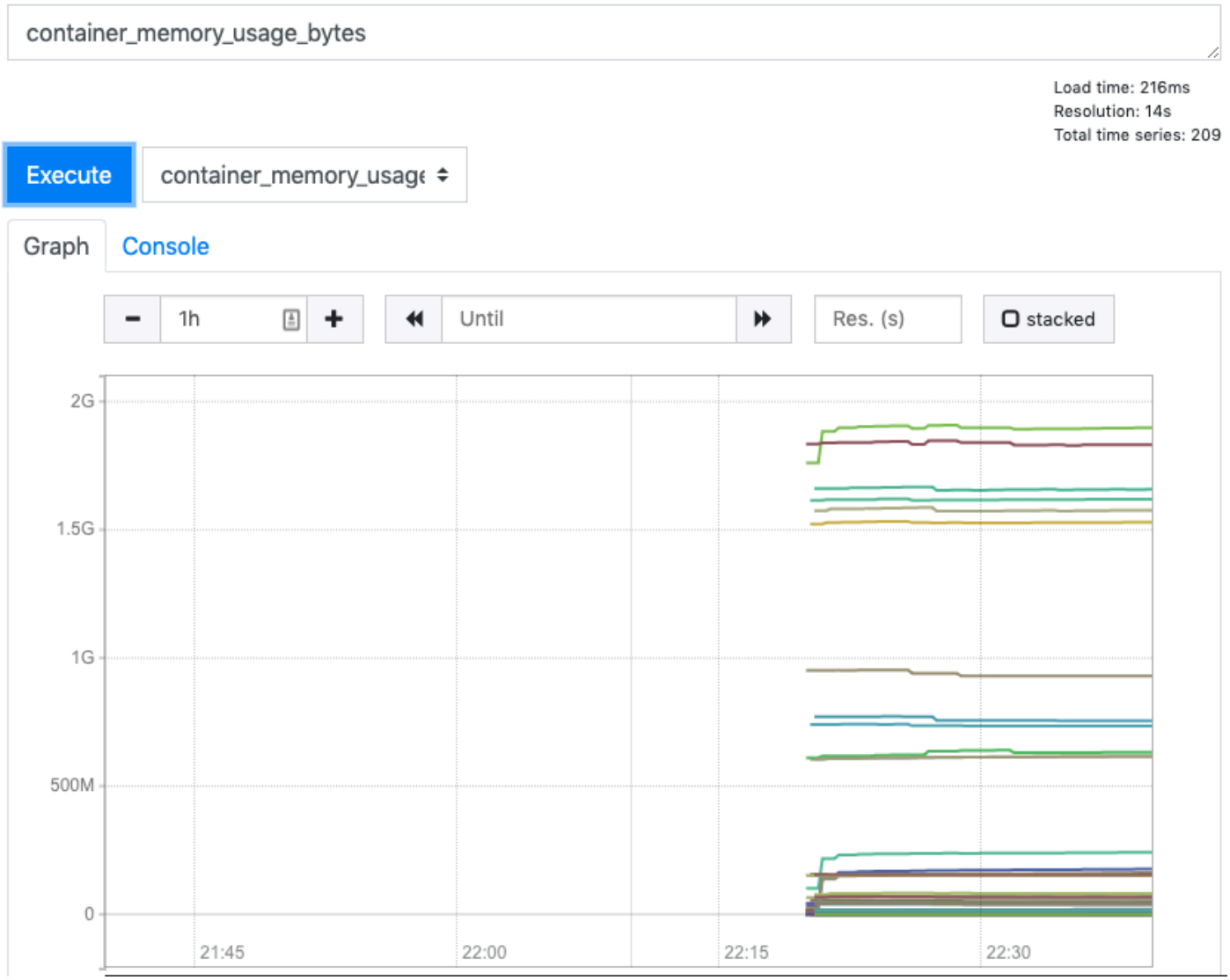
Eine Beispielausgabe sieht wie folgt aus.

```
NAME                                             READY   STATUS    RESTARTS   AGE
prometheus-alertmanager-59b4c8c744-r7bgp         1/2     Running   0          48s
prometheus-kube-state-metrics-7cfd87cf99-jkz2f   1/1     Running   0          48s
prometheus-node-exporter-jcjqz                   1/1     Running   0          48s
prometheus-node-exporter-jxv2h                   1/1     Running   0          48s
prometheus-node-exporter-vbdks                   1/1     Running   0          48s
prometheus-pushgateway-76c444b68c-82tnw          1/1     Running   0          48s
prometheus-server-775957f748-mmht9               1/2     Running   0          48s
```

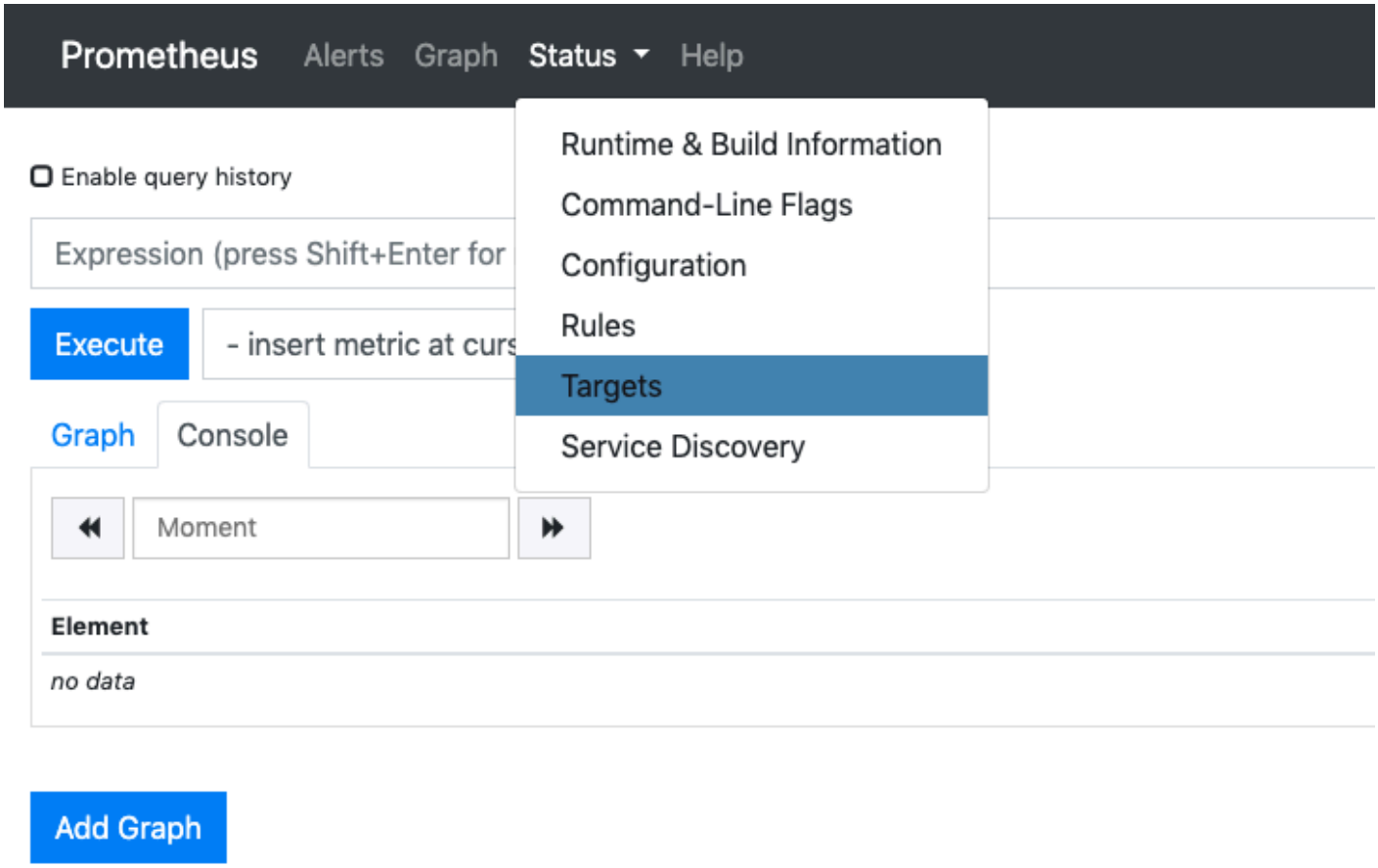
5. Verwenden Sie `kubectl`, um die Prometheus-Konsole auf Ihren lokalen Computer weiterzuleiten.

```
kubectl --namespace=prometheus port-forward deploy/prometheus-server 9090
```

6. Richten Sie einen Web-Browser auf `http://localhost:9090`, um die Prometheus-Konsole anzuzeigen.
7. Wählen Sie eine Metrik aus dem Menü - insert metric at cursor (Metrik bei Cursor eingeben) und danach Execute (Ausführen). Wählen Sie die Registerkarte Graph (Diagramm) aus, um die Metrik im Zeitverlauf anzuzeigen. Das folgende Image zeigt `container_memory_usage_bytes` im Zeitverlauf.



8. Wählen Sie in der oberen Navigationsleiste Status, dann Targets (Ziele).



Es werden alle Kubernetes-Endpunkte angezeigt, die über die Service-Erkennung mit Prometheus verbunden sind.

Rohmetriken der Steuerebene im Prometheus-Format abrufen

Die Kubernetes-Steuerebene stellt eine Reihe von Metriken bereit, die im Prometheus-Format dargestellt werden. Diese Metriken sind für die Überwachung und Analyse hilfreich. Sie werden intern über Metrik-Endpunkte bereitgestellt und können ohne vollständige Bereitstellung von Prometheus abgerufen werden. Die Bereitstellung von Prometheus vereinfacht jedoch die Analyse von Metriken im Zeitverlauf.

Um die Rohdaten der Metriken anzuzeigen, ersetzen Sie `endpoint` und führen Sie den folgenden Befehl aus.

```
kubectl get --raw endpoint
```

Mit diesem Befehl können Sie einen beliebigen Endpunktpfad übergeben und erhalten die Rohdaten als Antwort zurück. Die Ausgabe listet zeilenweise verschiedene Metriken auf, wobei jede Zeile einen Metriknamen, Tags und einen Wert enthält.

```
metric_name{tag="value"[,...]} value
```

Metriken vom API-Server abrufen

Der allgemeine API-Server-Endpunkt wird in der Amazon-EKS-Steuerebene bereitgestellt. Dieser Endpunkt ist vor allem nützlich, wenn Sie eine bestimmte Metrik betrachten möchten.

```
kubectl get --raw /metrics
```

Eine Beispielausgabe sieht wie folgt aus.

```
[...]
# HELP rest_client_requests_total Number of HTTP requests, partitioned by status code,
 method, and host.
# TYPE rest_client_requests_total counter
rest_client_requests_total{code="200",host="127.0.0.1:21362",method="POST"} 4994
rest_client_requests_total{code="200",host="127.0.0.1:443",method="DELETE"} 1
rest_client_requests_total{code="200",host="127.0.0.1:443",method="GET"} 1.326086e+06
rest_client_requests_total{code="200",host="127.0.0.1:443",method="PUT"} 862173
rest_client_requests_total{code="404",host="127.0.0.1:443",method="GET"} 2
rest_client_requests_total{code="409",host="127.0.0.1:443",method="POST"} 3
rest_client_requests_total{code="409",host="127.0.0.1:443",method="PUT"} 8
# HELP ssh_tunnel_open_count Counter of ssh tunnel total open attempts
# TYPE ssh_tunnel_open_count counter
ssh_tunnel_open_count 0
# HELP ssh_tunnel_open_fail_count Counter of ssh tunnel failed open attempts
# TYPE ssh_tunnel_open_fail_count counter
ssh_tunnel_open_fail_count 0
```

Diese Rohausgabe zeigt exakt das, was der API-Server bereitstellt.

Metriken der Steuerebene mit `metrics.eks.amazonaws.com` abrufen

Für Cluster mit Kubernetes-Version `1.28` und höher stellt Amazon EKS auch Metriken unter der API-Gruppe `metrics.eks.amazonaws.com` bereit. Diese Metriken umfassen Steuerebenen-Komponenten wie `kube-scheduler` und `kube-controller-manager`.

> **Note**
>
> Wenn Sie über eine Webhook-Konfiguration verfügen, welche die Erstellung der neuen `APIService`-Ressource `v1.metrics.eks.amazonaws.com` in Ihrem Cluster blockieren könnte, ist das Feature des Metrik-Endpunkts möglicherweise nicht verfügbar. Sie können dies im Überwachungsprotokoll `kube-apiserver` überprüfen, indem Sie nach dem Schlüsselwort `v1.metrics.eks.amazonaws.com` suchen.

`kube-scheduler`-Metriken abrufen

Verwenden Sie zum Abrufen von `kube-scheduler`-Metriken den folgenden Befehl.

```
kubectl get --raw "/apis/metrics.eks.amazonaws.com/v1/ksh/container/metrics"
```

Eine Beispielausgabe sieht wie folgt aus.

```
# TYPE scheduler_pending_pods gauge
scheduler_pending_pods{queue="active"} 0
scheduler_pending_pods{queue="backoff"} 0
scheduler_pending_pods{queue="gated"} 0
scheduler_pending_pods{queue="unschedulable"} 18
# HELP scheduler_pod_scheduling_attempts [STABLE] Number of attempts to successfully
 schedule a pod.
# TYPE scheduler_pod_scheduling_attempts histogram
scheduler_pod_scheduling_attempts_bucket{le="1"} 79
scheduler_pod_scheduling_attempts_bucket{le="2"} 79
scheduler_pod_scheduling_attempts_bucket{le="4"} 79
scheduler_pod_scheduling_attempts_bucket{le="8"} 79
scheduler_pod_scheduling_attempts_bucket{le="16"} 79
scheduler_pod_scheduling_attempts_bucket{le="+Inf"} 81
[...]
```

`kube-controller-manager`-Metriken abrufen

Verwenden Sie zum Abrufen von `kube-controller-manager`-Metriken den folgenden Befehl.

```
kubectl get --raw "/apis/metrics.eks.amazonaws.com/v1/kcm/container/metrics"
```

Eine Beispielausgabe sieht wie folgt aus.

```
[...]
workqueue_work_duration_seconds_sum{name="pvprotection"} 0
workqueue_work_duration_seconds_count{name="pvprotection"} 0
workqueue_work_duration_seconds_bucket{name="replicaset",le="1e-08"} 0
workqueue_work_duration_seconds_bucket{name="replicaset",le="1e-07"} 0
workqueue_work_duration_seconds_bucket{name="replicaset",le="1e-06"} 0
workqueue_work_duration_seconds_bucket{name="replicaset",le="9.999999999999999e-06"} 0
workqueue_work_duration_seconds_bucket{name="replicaset",le="9.999999999999999e-05"} 19
workqueue_work_duration_seconds_bucket{name="replicaset",le="0.001"} 109
workqueue_work_duration_seconds_bucket{name="replicaset",le="0.01"} 139
workqueue_work_duration_seconds_bucket{name="replicaset",le="0.1"} 181
workqueue_work_duration_seconds_bucket{name="replicaset",le="1"} 191
workqueue_work_duration_seconds_bucket{name="replicaset",le="10"} 191
workqueue_work_duration_seconds_bucket{name="replicaset",le="+Inf"} 191
workqueue_work_duration_seconds_sum{name="replicaset"} 4.265655885000002
[...]
```

Metriken des Schedulers und des Controller-Managers verstehen

In der folgenden Tabelle werden die Scheduler- und Controller-Manager-Metriken beschrieben, die für das Scraping im Prometheus-Stil verfügbar gemacht werden. eitere Informationen zu diesen Metriken finden Sie in der Kubernetes-Metriken-Referenz in der Kubernetes-Dokumentation.

Metrik	Komponente der Steuerebene	Beschreibung
scheduler_pending_pods	Scheduler	Die Anzahl der Pods, die darauf warten, zur Ausführun g in einen Knoten geplant zu werden.
scheduler_schedule_attempts _total	Scheduler	Die Anzahl der Versuche, Pods zu planen.
scheduler_preemption_attemp ts_total	Scheduler	Die Anzahl der Versuche des Schedulers, Pods mit höherer Priorität zu planen, indem Pods mit niedrigerer Priorität entfernt werden.

Metrik	Komponente der Steuerebene	Beschreibung
scheduler_preemption_victims	Scheduler	Die Anzahl der Pods, die zur Bereinigung ausgewählt wurden, um Platz für Pods mit höherer Priorität zu schaffen.
scheduler_pod_scheduling_attempts	Scheduler	Die Anzahl der Versuche, einen Pod erfolgreich zu planen.
scheduler_scheduling_attempt_duration_seconds	Scheduler	Gibt an, wie schnell oder langsam der Scheduler in der Lage ist, einen geeigneten Ausführungsort für einen Pod zu finden, basierend auf verschiedenen Faktoren wie Ressourcenverfügbarkeit und Planungsregeln.
scheduler_pod_scheduling_sli_duration_seconds	Scheduler	Die durchgängige Latenz für einen Pod, der geplant wird, ab dem Zeitpunkt, an dem der Pod in die Planungswarteschlange aufgenommen wird. Dies kann mehrere Planungsversuche umfassen.
cronjob_controller_job_creation_skew_duration_seconds	controller manager	Der Zeitraum zwischen der geplanten Ausführung eines Cronjobs und der Erstellung des entsprechenden Auftrags.
workqueue_depth	controller manager	Die aktuelle Ebene der Warteschlange.

Metrik	Komponente der Steuerebene	Beschreibung
workqueue_adds_total	controller manager	Die Gesamtzahl der von der Arbeitswarteschlange bearbeiteten Einträge.
workqueue_queue_duration_seconds	controller manager	Die Zeit in Sekunden, die ein Element in der Arbeitswarteschlange verbleibt, bevor es angefordert wird.
workqueue_work_duration_seconds	controller manager	Die Zeit in Sekunden, die für die Bearbeitung eines Elements aus der Arbeitswarteschlange benötigt wird.

Prometheus-Scraper einsetzen, um Metriken konsistent zu scrapen

Um einen Prometheus-Scraper einzusetzen, der die Metriken konsistent scrapt, verwenden Sie die folgende Konfiguration:

```
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: prometheus-conf
data:
  prometheus.yml: |-
    global:
      scrape_interval: 30s
    scrape_configs:
    # apiserver metrics
    - job_name: apiserver-metrics
      kubernetes_sd_configs:
      - role: endpoints
      scheme: https
      tls_config:
        ca_file: /var/run/secrets/kubernetes.io/serviceaccount/ca.crt
        insecure_skip_verify: true
      bearer_token_file: /var/run/secrets/kubernetes.io/serviceaccount/token
```

```
      relabel_configs:
      - source_labels:
          [
            __meta_kubernetes_namespace,
            __meta_kubernetes_service_name,
            __meta_kubernetes_endpoint_port_name,
          ]
        action: keep
        regex: default;kubernetes;https
    # Scheduler metrics
    - job_name: 'ksh-metrics'
      kubernetes_sd_configs:
      - role: endpoints
      metrics_path: /apis/metrics.eks.amazonaws.com/v1/ksh/container/metrics
      scheme: https
      tls_config:
        ca_file: /var/run/secrets/kubernetes.io/serviceaccount/ca.crt
        insecure_skip_verify: true
      bearer_token_file: /var/run/secrets/kubernetes.io/serviceaccount/token
      relabel_configs:
      - source_labels:
          [
            __meta_kubernetes_namespace,
            __meta_kubernetes_service_name,
            __meta_kubernetes_endpoint_port_name,
          ]
        action: keep
        regex: default;kubernetes;https
    # Controller Manager metrics
    - job_name: 'kcm-metrics'
      kubernetes_sd_configs:
      - role: endpoints
      metrics_path: /apis/metrics.eks.amazonaws.com/v1/kcm/container/metrics
      scheme: https
      tls_config:
        ca_file: /var/run/secrets/kubernetes.io/serviceaccount/ca.crt
        insecure_skip_verify: true
      bearer_token_file: /var/run/secrets/kubernetes.io/serviceaccount/token
      relabel_configs:
      - source_labels:
          [
            __meta_kubernetes_namespace,
            __meta_kubernetes_service_name,
            __meta_kubernetes_endpoint_port_name,
```

```
          ]
        action: keep
        regex: default;kubernetes;https
---
apiVersion: v1
kind: Pod
metadata:
  name: prom-pod
spec:
  containers:
  - name: prom-container
    image: prom/prometheus
    ports:
    - containerPort: 9090
    volumeMounts:
    - name: config-volume
      mountPath: /etc/prometheus/
  volumes:
  - name: config-volume
    configMap:
      name: prometheus-conf
```

Die folgende Berechtigung ist erforderlich, damit der Pod auf den neuen Metrik-Endpunkt zugreifen kann.

```
{
  "effect": "allow",
  "apiGroups": [
    "metrics.eks.amazonaws.com"
  ],
  "resources": [
    "kcm/metrics",
    "ksh/metrics"
  ],
  "verbs": [
    "get"
  ] },
```

Um die verwendete Rolle zu patchen, können Sie den folgenden Befehl verwenden.

```
kubectl patch clusterrole <role-name> --type=json -p='[
  {
    "op": "add",
```

```
    "path": "/rules/-",
    "value": {
      "verbs": ["get"],
      "apiGroups": ["metrics.eks.amazonaws.com"],
      "resources": ["kcm/metrics", "ksh/metrics"]
    }
  }
]'
```

Anschließend können Sie das Prometheus-Dashboard anzeigen, indem Sie den Port des Prometheus-Scrapers auf Ihren lokalen Port umstellen.

```
kubectl port-forward pods/prom-pod 9090:9090
```

Für Ihren Amazon-EKS-Cluster werden die Kernmetriken der Kubernetes-Steuerebene auch unter dem `AWS/EKS`-Namespace in Amazon CloudWatch-Metriken aufgenommen. Um sie anzuzeigen, öffnen Sie die CloudWatch-Konsole und wählen Sie im linken Navigationsbereich Alle Metriken aus. Wählen Sie auf der Auswahlseite Metriken den `AWS/EKS`-Namespace und eine Metrikdimension für Ihren Cluster aus.

Überwachen Sie Clusterdaten mit Amazon CloudWatch

Amazon CloudWatch ist ein Überwachungsdienst, der Metriken und Protokolle aus Ihren Cloud-Ressourcen sammelt. CloudWatch stellt einige grundlegende Amazon EKS-Metriken kostenlos zur Verfügung, wenn Sie einen neuen Cluster der Version `1.28` und höher verwenden. Wenn Sie den CloudWatch Observability Operator jedoch als Amazon EKS-Add-on verwenden, können Sie erweiterte Observability-Funktionen nutzen.

Grundlegende Kennzahlen in Amazon CloudWatch

Für Cluster der Kubernetes-Version `1.28` und höher erhalten Sie CloudWatch Metriken kostenlos im Namespace. `AWS/EKS` Die folgende Tabelle enthält eine Liste der grundlegenden Metriken, die für die unterstützten Versionen verfügbar sind. Jede aufgeführte Metrik hat eine Frequenz von einer Minute.

Metrikname	Description
`apiserver_flowcontrol_current_executing_seats`	Die Anzahl der Sitze, die derzeit für die Ausführung von API-Anfragen verwendet

Metrikname	Description
	werden. Die Sitzplatzzuweisung wird durch die Konfiguration priority_level und flow_schema in der Kubernetes-API-Funktion Priority and Fairness bestimmt. Einheiten: Anzahl Gültige Statistiken: Max
`scheduler_schedule_attempts_total`	Die Gesamtzahl der Versuche des Scheduler s, Pods im Cluster für einen bestimmten Zeitraum zu planen. Diese Metrik unterstützt die Überwachung der Workload des Scheduler s und kann auf Planungsdruck oder potenzielle Probleme bei der Pod-Platzierung hinweisen. Einheiten: Anzahl Gültige Statistiken: Summe
`scheduler_schedule_attempts_SCHEDULED`	Die Anzahl der erfolgreichen Versuche des Schedulers, Pods für einen bestimmten Zeitraum auf Knoten im Cluster zu planen. Einheiten: Anzahl Gültige Statistiken: Summe
`scheduler_schedule_attempts_UNSCHEDULABLE`	Die Anzahl der Versuche, Pods zu planen, die für einen bestimmten Zeitraum aufgrund gültiger Einschränkungen, wie z. B. unzureich ender CPU oder Speicher auf einem Knoten, nicht planbar waren. Einheiten: Anzahl Gültige Statistiken: Summe

Metrikname	Description
scheduler_schedule_attempts_ERROR	Die Anzahl der Versuche, Pods zu planen, die für einen bestimmten Zeitraum aufgrund eines internen Problems mit dem Planer selbst, wie z. B. Verbindungsproblemen des API-Servers, fehlgeschlagen sind. Einheiten: Anzahl Gültige Statistiken: Summe
scheduler_pending_pods	Die Gesamtzahl der ausstehenden Pods, die vom Scheduler im Cluster für einen bestimmten Zeitraum geplant werden sollen. Einheiten: Anzahl Gültige Statistiken: Summe
scheduler_pending_pods_ACTIVEQ	Die Anzahl der ausstehenden Pods in ActiveQ, die darauf warten, für einen bestimmten Zeitraum im Cluster geplant zu werden. Einheiten: Anzahl Gültige Statistiken: Summe
scheduler_pending_pods_UNSCHEDULABLE	Die Anzahl der ausstehenden Pods, die der Planer zu planen versucht hat, bei denen dies jedoch fehlgeschlagen ist, und die für einen erneuten Versuch in einem nicht planbaren Zustand gehalten werden. Einheiten: Anzahl Gültige Statistiken: Summe

Metrikname	Description
`scheduler_pending_pods_BACKOFF`	Die Anzahl der ausstehenden Pods in `backoffQ` im Backoff-Zustand, die auf den Ablauf ihrer Backoff-Periode warten. Einheiten: Anzahl Gültige Statistiken: Summe
`scheduler_pending_pods_GATED`	Die Anzahl der ausstehenden Pods, die derzeit in einem gesperrten Zustand warten, da sie erst geplant werden können, wenn sie die erforderl ichen Bedingungen erfüllen. Einheiten: Anzahl Gültige Statistiken: Summe
`apiserver_request_total`	Die Anzahl der HTTP-Anfragen, die über alle API-Server im Cluster gestellt wurden. Einheiten: Anzahl Gültige Statistiken: Summe
`apiserver_request_total_4XX`	Die Anzahl der HTTP-Anfragen an alle API-Server im Cluster, die zu 4XX-Statuscodes (clientseitige Fehler) führten. Einheiten: Anzahl Gültige Statistiken: Summe

Metrikname	Description
apiserver_request_total_429	Die Anzahl der HTTP-Anfragen an alle API-Server im Cluster, die zu 429Statuscodes geführt haben, die auftreten, wenn Clients die Schwellenwerte für die Ratenbegrenzung überschreiten. Einheiten: Anzahl Gültige Statistiken: Summe
apiserver_request_total_5XX	Die Anzahl der HTTP-Anfragen an alle API-Server im Cluster, die zu 5XX-Statuscodes (Serverfehler) führten. Einheiten: Anzahl Gültige Statistiken: Summe
apiserver_request_total_LIST_PODS	Die Anzahl der LIST-Pods-Anfragen an alle API-Server im Cluster. Einheiten: Anzahl Gültige Statistiken: Summe
apiserver_request_duration_seconds_PUT_P99	Das 99. Perzentil der Latenz für PUT-Anfragen, berechnet aus allen Anfragen aller API-Server im Cluster. Stellt die Antwortzeit dar, unter der 99 % aller PUT-Anfragen abgeschlossen werden. Einheiten: Sekunden Gültige Statistiken: Durchschnitt

Metrikname	Description
apiserver_request_duration_seconds_PATCH_P99	Das 99. Perzentil der Latenz für PATCH-Anfragen, berechnet aus allen Anfragen aller API-Server im Cluster. Stellt die Antwortzeit dar, unter der 99 % aller PATCH-Anfragen abgeschlossen werden. Einheiten: Sekunden Gültige Statistiken: Durchschnitt
apiserver_request_duration_seconds_POST_P99	Das 99. Perzentil der Latenz für POST-Anfragen, berechnet aus allen Anfragen aller API-Server im Cluster. Stellt die Antwortzeit dar, unter der 99 % aller POST-Anfragen abgeschlossen werden. Einheiten: Sekunden Gültige Statistiken: Durchschnitt
apiserver_request_duration_seconds_GET_P99	Das 99. Perzentil der Latenz für GET-Anfragen, berechnet aus allen Anfragen aller API-Server im Cluster. Stellt die Antwortzeit dar, unter der 99 % aller GET-Anfragen abgeschlossen werden. Einheiten: Sekunden Gültige Statistiken: Durchschnitt

Metrikname	Description
`apiserver_request_duration_seconds_LIST_P99`	Das 99. Perzentil der Latenz für `LIST`-Anfragen, berechnet aus allen Anfragen aller API-Server im Cluster. Stellt die Antwortzeit dar, unter der 99 % aller `LIST`-Anfragen abgeschlossen werden. Einheiten: Sekunden Gültige Statistiken: Durchschnitt
`apiserver_request_duration_seconds_DELETE_P99`	Das 99. Perzentil der Latenz für `DELETE`-Anfragen, berechnet aus allen Anfragen aller API-Server im Cluster. Stellt die Antwortzeit dar, unter der 99 % aller `DELETE`-Anfragen abgeschlossen werden. Einheiten: Sekunden Gültige Statistiken: Durchschnitt
`apiserver_current_inflight_requests_MUTATING`	Die Anzahl der mutierenden Anfragen (`POST`, `PUT`, `DELETE`, `PATCH`), die derzeit auf allen API-Servern im Cluster verarbeitet werden. Diese Metrik stellt Anfragen dar, die sich in der Bearbeitung befinden und deren Verarbeitung noch nicht abgeschlossen ist Einheiten: Anzahl Gültige Statistiken: Summe

Metrikname	Description
`apiserver_current_inflight_requests_READONLY`	Die Anzahl der schreibgeschützten Anfragen (`GET`, `LIST`), die derzeit auf allen API-Server im Cluster verarbeitet werden. Diese Metrik stellt Anfragen dar, die sich in der Bearbeitung befinden und deren Verarbeitung noch nicht abgeschlossen ist Einheiten: Anzahl Gültige Statistiken: Summe
`apiserver_admission_webhook_request_total`	Die Anzahl der Zulassungs-Webhook-Anfragen, die über alle API-Server im Cluster gestellt wurden Einheiten: Anzahl Gültige Statistiken: Summe
`apiserver_admission_webhook_request_total_ADMIT`	Die Anzahl der mutierenden Zulassungs-Webhook-Anfragen, die über alle API-Server im Cluster gestellt wurden. Einheiten: Anzahl Gültige Statistiken: Summe
`apiserver_admission_webhook_request_total_VALIDATING`	Die Anzahl der validierenden Zulassungs-Webhook-Anfragen, die über alle API-Server im Cluster gestellt wurden. Einheiten: Anzahl Gültige Statistiken: Summe

Metrikname	Description
`apiserver_admission_webhook_rejection_count`	Die Anzahl der Zulassungs-Webhook-Anfragen, die über alle API-Server im Cluster gestellt und abgelehnt wurden. Einheiten: Anzahl Gültige Statistiken: Summe
`apiserver_admission_webhook_rejection_count_ADMIT`	Die Anzahl der abgelehnten mutierenden Zulassungs-Webhook-Anfragen über alle API-Server im Cluster. Einheiten: Anzahl Gültige Statistiken: Summe
`apiserver_admission_webhook_rejection_count_VALIDATING`	Die Anzahl der Webhook-Anfragen zur Validierung der Zulassung, die über alle API-Server im Cluster gestellt und abgelehnt wurden. Einheiten: Anzahl Gültige Statistiken: Summe
`apiserver_admission_webhook_admission_duration_seconds`	Das 99. Perzentil der Latenz für Zulassung s-Webhook-Anfragen von Drittanbietern, berechnet aus allen Anfragen aller API-Server im Cluster. Stellt die Antwortzeit dar, unter der 99 % aller Zulassungs-Webhook-Anfragen von Drittanbietern abgeschlossen werden. Einheiten: Sekunden Gültige Statistiken: Durchschnitt

Metrikname	Description
apiserver_admission_webhook_admission_duration_seconds_ADMIT_P99	Das 99. Perzentil der Latenz für mutierende Zulassungs-Webhook-Anfragen von Drittanbietern, berechnet aus allen Anfragen aller API-Server im Cluster. Stellt die Antwortzeit dar, unter der 99 % aller mutierenden Zulassungs-Webhook-Anfragen von Drittanbietern abgeschlossen werden. Einheiten: Sekunden Gültige Statistiken: Durchschnitt
apiserver_admission_webhook_admission_duration_seconds_VALIDATING_P99	Das 99. Perzentil der Latenz für Webhook-Anfragen zur Validierung von Zulassungen durch Dritte, berechnet aus allen Anfragen aller API-Server im Cluster. Stellt die Antwortzeit dar, unter der 99 % aller Webhook-Anfragen von Drittanbietern zur Validierung der Zulassung abgeschlossen werden. Einheiten: Sekunden Gültige Statistiken: Durchschnitt
apiserver_storage_size_bytes	Die physische Größe in Bytes der etcd-Speicher-Datenbankdatei, die von den API-Servern im Cluster verwendet wird. Diese Metrik stellt den tatsächlich für den Speicher zugewiesenen Speicherplatz dar. Einheiten: Byte Gültige Statistiken: Maximum

Amazon CloudWatch Observability-Betreiber

Amazon CloudWatch Observability sammelt Protokolle, Metriken und Trace-Daten in Echtzeit. Es sendet sie an Amazon CloudWatch und AWS X-Ray. Sie können dieses Add-on installieren, um sowohl CloudWatch Application Signals als auch CloudWatch Container Insights mit verbesserter Observability für Amazon EKS zu aktivieren. Dies hilft Ihnen, den Zustand und die Leistung Ihrer Infrastruktur und containerisierten Anwendungen zu überwachen. Der Amazon CloudWatch Observability Operator ist für die Installation und Konfiguration der erforderlichen Komponenten konzipiert.

Amazon EKS unterstützt den CloudWatch Observability Operator als Amazon EKS-Add-on. Das Add-On ermöglicht den Einsatz von Container Insights sowohl auf Linux- als auch auf Windows-Worker-Knoten im Cluster. Um Container Insights unter Windows zu aktivieren, muss die Amazon-EKS-Add-On-Version mindestens `1.5.0` sein. Derzeit wird CloudWatch Application Signals auf Amazon EKS Windows nicht unterstützt.

In den folgenden Themen werden die ersten Schritte mit der Verwendung von CloudWatch Observability Operator für Ihren Amazon EKS-Cluster beschrieben.

- Anweisungen zur Installation dieses Add-ons finden Sie unter Installieren des CloudWatch Agenten mit dem Amazon CloudWatch Observability EKS-Add-on oder dem Helm-Diagramm im CloudWatch Amazon-Benutzerhandbuch.
- Weitere Informationen zu CloudWatch Application Signals finden Sie unter Application Signals im CloudWatch Amazon-Benutzerhandbuch.
- Weitere Informationen zu Container Insights finden Sie unter Using Container Insights im CloudWatch Amazon-Benutzerhandbuch.

Übermittlung von Steuerebenen-Protokollen an CloudWatch Logs

Die Protokollierung der Amazon-EKS-Steuerebene bietet Prüf- und Diagnoseprotokolle direkt von der Amazon-EKS-Steuerebene zu CloudWatch Logs in Ihrem Konto. Diese Protokolle erleichtern Ihnen die Absicherung und Ausführung Ihrer Cluster. Sie können die Protokolltypen auswählen, die Sie benötigen. Die Protokolle werden für jeden Amazon-EKS-Cluster in CloudWatch als Protokollstreams an eine Gruppe gesendet. Sie können CloudWatch-Abonnementfilter verwenden, um Echtzeitanalysen der Protokolle durchzuführen oder diese an andere Services weiterzuleiten (die Protokolle werden Base64-codiert und im gzip-Format komprimiert). Weitere Informationen finden Sie unter in der Amazon-CloudWatch-Protokollierung.

Sie können mit der Protokollierung der Amazon-EKS-Steuerebene beginnen, indem Sie die für jeden neuen oder bestehenden Amazon-EKS-Cluster zu aktivierenden Protokolltypen auswählen. Sie können die einzelnen Protokolltypen pro Cluster über die AWS-Managementkonsole, AWS CLI (Version `1.16.139` oder höher) oder die Amazon-EKS-API aktivieren oder deaktivieren. Wenn aktiviert, werden die Protokolle automatisch vom Amazon-EKS-Cluster an CloudWatch Logs im selben Konto gesendet.

Wenn Sie die Protokollierung der Amazon-EKS-Steuerebene verwenden, werden Ihnen für jeden betriebenen Cluster die Amazon-EKS-Standardpreise berechnet. Für alle Protokolle, die von Ihren Clustern an CloudWatch Logs gesendet werden, werden Ihnen die standardmäßigen CloudWatch-Logs-Datenaufnahme- und -speicherkosten in Rechnung gestellt. Ihnen werden außerdem alle als Teil Ihres Clusters bereitgestellten AWS-Ressourcen (z. B. Amazon-EC2-Instances oder Amazon-EBS-Volumes) berechnet.

Die folgenden Cluster-Steuerebenen-Protokolltypen sind verfügbar. Jeder Protokolltyp entspricht einer Komponente der Kubernetes-Steuerebene. Um mehr über diese Komponenten zu erfahren, lesen Sie Kubernetes-Komponenten in der Kubernetes-Dokumentation.

API-Server**`api`** ()

> Der API-Server Ihres Clusters ist die Steuerebenen-Komponente, welche die Kubernetes-API bereitstellt. Wenn Sie API-Serverprotokolle beim Starten des Clusters oder kurz danach aktivieren, enthalten die Protokolle API-Server-Markierungen, die zum Starten des API-Servers verwendet wurden. Weitere Informationen finden Sie unter kube-apiserver und in der Prüfungsrichtlinie in der Kubernetes-Dokumentation.

Prüfung**`audit`** ()

> Kubernetes-Auditprotokolle bieten eine Aufzeichnung der einzelnen Benutzer, Administratoren oder Systemkomponenten, die Ihren Cluster beeinflusst haben. Weitere Informationen finden Sie unter Prüfung in der Kubernetes-Dokumentation.

Authenticator**`authenticator`** ()

> Authenticator-Protokolle sind eindeutig für Amazon EKS. Diese Protokolle stellen die Steuerebenenkomponente dar, die Amazon EKS für die Kubernetes-Authentifizierung per rollenbasierter Zugriffssteuerung (RBAC) mit IAM-Anmeldeinformationen verwendet. Weitere Informationen finden Sie unter Clusterverwaltung.

Controller-Manager**`controllerManager`** ()

Der Controller-Manager verwaltet die zentralen Regelkreise, die mit Kubernetes ausgeliefert werden. Weitere Informationen finden Sie unter kube-controller-manager in der Kubernetes-Dokumentation.

Scheduler**`scheduler`** ()

Die Planer-Komponente verwaltet, wann und wo Pods in Ihrem Cluster ausgeführt werden. Weitere Informationen finden Sie unter kube-scheduler in der Kubernetes-Dokumentation.

Aktivierung oder Deaktivierung der Steuerebenen-Protokolle

Standardmäßig werden Protokolle der Cluster-Steuerebene nicht an CloudWatch Logs gesendet. Sie müssen jeden Protokolltyp einzeln aktivieren, um Protokolle für Ihren Cluster zu senden. CloudWatch Logs-Aufnahme, -Archivspeicher, und -Datenscanraten gelten für aktivierte Protokolle der Steuerungsebene. Weitere Informationen hierzu finden Sie unter Amazon CloudWatch – Preise.

Um die Protokollierungskonfiguration der Steuerebene zu aktualisieren, benötigt Amazon EKS bis zu fünf verfügbare IP-Adressen in jedem Subnetz. Wenn Sie einen Protokolltyp aktivieren, werden die Protokolle mit der Protokollausführungsstufe 2 gesendet.

Sie können Steuerebenen-Protokolle entweder mit AWS-Managementkonsole oder der AWS-CLI aktivieren oder deaktivieren.

AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie den Namen des Clusters aus, um Ihre Cluster-Informationen anzuzeigen.
3. Wählen Sie den Registerkarte Beobachtbarkeit.
4. Wählen Sie im Abschnitt Steuerebenen-Protokollierung die Option Protokollierung verwalten aus.
5. Wählen Sie für jeden Protokolltyp aus, ob er aktiviert oder deaktiviert sein soll. Standardmäßig sind alle Protokollierungstypen deaktiviert.
6. Wählen Sie Änderungen speichern, um den Vorgang abzuschließen.

AWS-CLI

1. Sie können Ihre AWS-CLI-Version mit dem folgenden Befehl überprüfen.

```
aws --version
```

Bei einer AWS-CLI-Version vor `1.16.139` müssen Sie zunächst auf die neueste Version aktualisieren. Informationen zum Installieren oder Aktualisieren der AWS-CLI finden Sie unter Installation der AWS-Befehlszeilenschnittstelle im Benutzerhandbuch zur AWS-Befehlszeilenschnittstelle.

2. Aktualisieren Sie die Exportkonfiguration des Steuerebenen-Protokolls Ihres Clusters mit dem folgenden AWS-CLI-Befehl. Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters und geben Sie die gewünschten Endpunkt-Zugriffswerte ein.

> **Note**
>
> Der folgende Befehl sendet alle verfügbaren Protokolltypen an CloudWatch Logs.

```
aws eks update-cluster-config \
    --region region-code \
    --name my-cluster \
    --logging '{"clusterLogging":[{"types":
["api","audit","authenticator","controllerManager","scheduler"],"enabled":true}]}'
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "update": {
        "id": "883405c8-65c6-4758-8cee-2a7c1340a6d9",
        "status": "InProgress",
        "type": "LoggingUpdate",
        "params": [
            {
                "type": "ClusterLogging",
                "value": "{\"clusterLogging\":[{\"types\":[\"api\",\"audit\",
\"authenticator\",\"controllerManager\",\"scheduler\"],\"enabled\":true}]}"
            }
        ],
        "createdAt": 1553271814.684,
        "errors": []
    }
```

```
}
```

3. Überwachen Sie den Status Ihres Protokoll-Konfigurationsupdates mit dem folgenden Befehl unter Verwendung des Cluster-Namens und der Update-ID, die vom vorherigen Befehl zurückgegeben wurden. Ihre Aktualisierung ist abgeschlossen, wenn als Status `Successful` angezeigt wird.

```
aws eks describe-update \
    --region region-code\
    --name my-cluster \
    --update-id 883405c8-65c6-4758-8cee-2a7c1340a6d9
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "update": {
        "id": "883405c8-65c6-4758-8cee-2a7c1340a6d9",
        "status": "Successful",
        "type": "LoggingUpdate",
        "params": [
            {
                "type": "ClusterLogging",
                "value": "{\"clusterLogging\":[{\"types\":[\"api\",\"audit\",
\"authenticator\",\"controllerManager\",\"scheduler\"],\"enabled\":true}]}"
            }
        ],
        "createdAt": 1553271814.684,
        "errors": []
    }
}
```

Cluster-Steuerebenen-Protokolle anzeigen

Nachdem Sie Steuerebenen-Protokolltypen für Ihren Amazon-EKS-Cluster aktiviert haben, können Sie diese in der CloudWatch-Konsole einsehen.

Weitere Informationen zum Anzeigen, Analysieren und Verwalten von Protokollen in CloudWatch finden Sie im Amazon-CloudWatch-Logs-Benutzerhandbuch.

1. Öffnen Sie die CloudWatch-Konsole. Der Link öffnet die Konsole und zeigt Ihre aktuell verfügbaren Protokollgruppen an und filtert sie mit dem `/aws/eks`-Präfix.

2. Wählen Sie den Cluster aus, für den Sie die Protokolle anzeigen möchten. Das Format für den Namen der Protokollgruppen lautet `/aws/eks/my-cluster/cluster`.
3. Wählen Sie den anzuzeigenden Protokollstream aus. Die folgende Liste beschreibt das Namensformat des Protokollstreams der einzelnen Protokolltypen.

   > **Note**
   >
   > Wenn die Daten des Protokollstreams ansteigen, werden die Namen des Protokollstreams rotiert. Wenn mehrere Protokollstreams für einen bestimmten Protokolltyp vorhanden sind, können Sie den neuesten Protokollstream anzeigen, indem Sie nach dem Namen des Protokollstreams mit der letzten Last event time (Uhrzeit des letzten Ereignisses) suchen.

   - Kubernetes-API-Server-Komponentenprotokolle (**`api`**) – `kube-apiserver-1234567890abcdef01234567890abcde`
   - Prüfung (**`audit`**) – `kube-apiserver-audit-1234567890abcdef01234567890abcde`
   - Authentifikator (**`authenticator`**) – `authenticator-1234567890abcdef01234567890abcde`
   - Controller-Manager (**`controllerManager`**) – `kube-controller-manager-1234567890abcdef01234567890abcde`
   - Scheduler (**`scheduler`**) – `kube-scheduler-1234567890abcdef01234567890abcde`
4. Sehen Sie sich die Ereignisse des Protokollstreams an.

   Sie sollten beispielsweise die anfänglichen API-Server-Flags für den Cluster sehen, wenn Sie den oberen Teil von `kube-apiserver-1234567890abcdef01234567890abcde` anzeigen.

   > **Note**
   >
   > Wenn die API-Serverprotokolle am Anfang des Protokollstreams nicht angezeigt werden, ist es wahrscheinlich, dass die API-Serverprotokolldatei auf dem Server rotiert wurde, bevor Sie die API-Serverprotokollierung auf dem Server aktiviert haben. Alle Protokolldateien, die rotiert werden, bevor die API-Serverprotokollierung aktiviert wird, können nicht nach CloudWatch exportiert werden.

Sie können jedoch einen neuen Cluster mit derselben Kubernetes-Version erstellen und beim Erstellen des Clusters die API-Serverprotokollierung aktivieren. Für Cluster mit derselben Plattformversion sind dieselben Flags aktiviert, daher sollten Ihre Flags mit den Flags des neuen Clusters übereinstimmen. Nachdem Sie die Flags für den neuen Cluster in CloudWatch angezeigt haben, können Sie den neuen Cluster löschen.

API-Aufrufe als AWS-CloudTrail-Ereignisse protokollieren

Amazon EKS ist in AWS CloudTrail integriert. CloudTrail ist ein Service, der eine Aufzeichnung der Aktionen eines Benutzers, einer Rolle oder eines AWS-Services in Amazon EKS bereitstellt. CloudTrail erfasst alle API-Aufrufe für Amazon EKS als Ereignisse. Dazu gehören Aufrufe von der Amazon-EKS-Konsole und von Code-Aufrufen der Amazon-EKS-API-Operationen.

Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Bereitstellung von CloudTrail-Ereignissen an einen Amazon S3-Bucket aktivieren. Dies umfasst Ereignisse von Amazon EKS. Wenn Sie keinen Verlauf konfigurieren, können Sie die neuesten Ereignisse in der CloudTrail-Konsole trotzdem in Ereignisverlauf anzeigen. Anhand der Informationen, die CloudTrail sammelt, können Sie mehrere Details zu einer Anfrage ermitteln. Sie können zum Beispiel ermitteln, wann eine Anfrage an Amazon EKS gestellt wurde, von welcher IP-Adresse aus die Anfrage gestellt wurde und wer die Anfrage gestellt hat.

Weitere Informationen zu CloudTrail finden Sie im AWS-CloudTrail-Benutzerhandbuch.

Themen

- Hilfreiche Referenzen für AWS CloudTrail anzeigen
- Analyse von AWS-CloudTrail-Protokolldateieinträgen
- Anzeigen von Metriken für Amazon-EC2-Auto-Scaling-Gruppen

Hilfreiche Referenzen für AWS CloudTrail anzeigen

Wenn Sie Ihr AWS-Konto erstellen, wird CloudTrail ebenfalls in Ihrem AWS-Konto aktiviert. Wenn in Amazon EKS eine Aktivität auftritt, wird sie in einem CloudTrail-Ereignis zusammen mit anderen AWS-Service-Ereignissen im Ereignisverlauf aufgezeichnet. Sie können die neusten Ereignisse in Ihr AWS-Konto herunterladen und dort suchen und anzeigen. Weitere Informationen finden Sie unter Anzeigen von Ereignissen mit dem CloudTrail-API-Ereignisverlauf.

Für eine fortlaufende Aufzeichnung der Ereignisse in Ihrem AWS-Konto, darunter Ereignisse für Amazon EKS, können Sie einen Pfad (Trail) erstellen. Ein Trail ermöglicht es CloudTrail,

Protokolldateien in einem Amazon-S3-Bucket bereitzustellen. Wenn Sie einen Trail in der Konsole anlegen, gilt dieser für alle AWS-Regionen. Der Trail protokolliert Ereignisse aus allen AWS-Regionen in der AWS-Partition und stellt die Protokolldateien in dem Amazon-S3-Bucket bereit, den Sie angeben. Darüber hinaus können Sie andere AWS-Services konfigurieren, um die in den CloudTrail-Protokollen erfassten Ereignisdaten weiter zu analysieren und entsprechend zu handeln. Weitere Informationen finden Sie in den folgenden Ressourcen.

- Übersicht zum Erstellen eines Trails
- In CloudTrail unterstützte Services und Integrationen
- Konfigurieren von Amazon SNS-Benachrichtigungen für CloudTrail
- Empfangen von CloudTrail-Protokolldateien aus mehreren Regionen und Empfangen von CloudTrail-Protokolldateien von mehreren Konten.

Alle Amazon EKS-Aktionen werden von CloudTrail protokolliert und sind in der Amazon EKS-API-Referenz dokumentiert. Beispielsweise generieren das Aufrufen der Abschnitte CreateCluster, ListClusters und DeleteCluster Einträge in den CloudTrail-Protokolldateien.

Jeder Ereignis- oder Protokolleintrag enthält Informationen über die Art der für die Abfrage genutzten IAM-Identität und dazu, welche Anmeldeinformationen verwendet wurden. Wenn temporäre Anmeldeinformationen verwendet wurden, zeigt der Eintrag, wie die Anmeldeinformationen bezogen wurden.

Weitere Informationen finden Sie unter CloudTrail-Element userIdentity.

Analyse von AWS-CloudTrail-Protokolldateieinträgen

Ein Trail ist eine Konfiguration, durch die Ereignisse als Protokolldateien an den von Ihnen angegebenen Amazon-S3-Bucket übermittelt werden. CloudTrail-Protokolldateien können einen oder mehrere Einträge enthalten. Ein Ereignis stellt eine einzelne Anfrage aus einer beliebigen Quelle dar und enthält unter anderem Informationen über die angeforderte Aktion. Dazu gehören Informationen zu Datum und Uhrzeit der Aktion sowie zu den verwendeten Anfrageparameter. CloudTrail-Protokolldateien sind keine geordnete Stack-Verfolgung der öffentlichen API-Aufrufe und werden daher nicht in einer bestimmten Reihenfolge angezeigt.

Das folgende Beispiel zeigt einen CloudTrail-Protokolleintrag, der die Aktion `CreateCluster` demonstriert.

```
{
```

```
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AKIAIOSFODNN7EXAMPLE",
    "arn": "arn:aws:iam::111122223333:user/username",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "username"
  },
  "eventTime": "2018-05-28T19:16:43Z",
  "eventSource": "eks.amazonaws.com",
  "eventName": "CreateCluster",
  "awsRegion": "region-code",
  "sourceIPAddress": "205.251.233.178",
  "userAgent": "PostmanRuntime/6.4.0",
  "requestParameters": {
    "resourcesVpcConfig": {
      "subnetIds": [
        "subnet-a670c2df",
        "subnet-4f8c5004"
      ]
    },
    "roleArn": "arn:aws:iam::111122223333:role/AWSServiceRoleForAmazonEKS-
CAC1G1VH3ZKZ",
    "clusterName": "test"
  },
  "responseElements": {
    "cluster": {
      "clusterName": "test",
      "status": "CREATING",
      "createdAt": 1527535003.208,
      "certificateAuthority": {},
      "arn": "arn:aws:eks:region-code:111122223333:cluster/test",
      "roleArn": "arn:aws:iam::111122223333:role/AWSServiceRoleForAmazonEKS-
CAC1G1VH3ZKZ",
      "version": "1.10",
      "resourcesVpcConfig": {
        "securityGroupIds": [],
        "vpcId": "vpc-21277358",
        "subnetIds": [
          "subnet-a670c2df",
          "subnet-4f8c5004"
        ]
      }
```

```
    }
  },
  "requestID": "a7a0735d-62ab-11e8-9f79-81ce5b2b7d37",
  "eventID": "eab22523-174a-499c-9dd6-91e7be3ff8e3",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

Protokolleinträge für serviceverknüpfte Amazon-EKS-Rollen

Die serviceverknüpften Amazon EKS-Rollen führen API-Aufrufe an AWS-Ressourcen aus. CloudTrail-Protokolleinträge mit `username: AWSServiceRoleForAmazonEKS` und `username: AWSServiceRoleForAmazonEKSNodegroup` werden für Aufrufe angezeigt, die von den serviceverknüpften Amazon-EKS-Rollen ausgeführt werden. Weitere Informationen zu Amazon EKS und serviceverknüpften Rollen finden Sie unter the section called “Service-verknüpfte Rollen”.

Das folgende Beispiel zeigt einen CloudTrail-Protokolleintrag, der eine `DeleteInstanceProfile`-Aktion der serviceverknüpften Rolle `AWSServiceRoleForAmazonEKSNodegroup` veranschaulicht, die im `sessionContext` vermerkt ist.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3WHGPEZ7SJ2CW55C5:EKS",
        "arn": "arn:aws:sts::111122223333:assumed-role/
AWSServiceRoleForAmazonEKSNodegroup/EKS",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA3WHGPEZ7SJ2CW55C5",
                "arn": "arn:aws:iam::111122223333:role/aws-service-role/eks-
nodegroup.amazonaws.com/AWSServiceRoleForAmazonEKSNodegroup",
                "accountId": "111122223333",
                "userName": "AWSServiceRoleForAmazonEKSNodegroup"
            },
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
```

```
                "creationDate": "2020-02-26T00:56:33Z"
            }
        },
        "invokedBy": "eks-nodegroup.amazonaws.com"
    },
    "eventTime": "2020-02-26T00:56:34Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "DeleteInstanceProfile",
    "awsRegion": "region-code",
    "sourceIPAddress": "eks-nodegroup.amazonaws.com",
    "userAgent": "eks-nodegroup.amazonaws.com",
    "requestParameters": {
        "instanceProfileName": "eks-11111111-2222-3333-4444-abcdef123456"
    },
    "responseElements": null,
    "requestID": "11111111-2222-3333-4444-abcdef123456",
    "eventID": "11111111-2222-3333-4444-abcdef123456",
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

Anzeigen von Metriken für Amazon-EC2-Auto-Scaling-Gruppen

Bei von Amazon EKS verwalteten Knotengruppen sind Gruppenmetriken von Amazon EC2 Auto Scaling standardmäßig und ohne zusätzliche Kosten aktiviert. Die Auto-Scaling-Gruppe übermittelt jede Minute Stichprobendaten an Amazon CloudWatch. Diese Metriken können nach dem Namen der Auto-Scaling-Gruppen verfeinert werden. Sie bieten Ihnen kontinuierliche Transparenz hinsichtlich des Verlaufs der Auto-Scaling-Gruppe, die Ihre verwalteten Knotengruppen unterstützt, beispielsweise hinsichtlich Änderungen der Gruppengröße im Laufe der Zeit. Die Metriken der Auto-Scaling-Gruppe sind in Amazon CloudWatch oder in der Auto-Scaling-Konsole verfügbar. Weitere Informationen finden Sie unter Überwachung von CloudWatch-Metriken für Ihre Auto-Scaling-Gruppen und -Instances.

Mit der Erfassung von Auto-Scaling-Gruppenmetriken können Sie die Skalierung von verwalteten Knotengruppen überwachen. Die Auto-Scaling-Gruppenmetriken zeigen die minimale, maximale und gewünschte Größe einer Auto-Scaling-Gruppe an. Sie können eine Warnung auslösen, wenn die Anzahl der Knoten in einer Knotengruppe unter die Mindestgröße fällt, was auf eine fehlerhafte Knotengruppe hindeuten würde. Die Nachverfolgung der Knotengruppengröße ist auch nützlich, um die maximale Anzahl anzupassen, damit Ihre Datenebene nicht an Kapazität verliert.

Wenn Sie diese Messwerte nicht erfassen lassen möchten, können Sie alle oder nur einige davon deaktivieren. So können Sie beispielsweise Störungen in Ihren CloudWatch-Dashboards vermeiden. Weitere Informationen finden Sie unter Amazon-CloudWatch-Metriken für Amazon EC2 Auto Scaling.

Senden von Metrik- und Nachverfolgungsdaten mit ADOT Operator

Amazon EKS unterstützt die Verwendung der AWS-Managementkonsole, AWS-CLI sowie der Amazon-EKS-API zur Installation und Verwaltung von AWS Distro for OpenTelemetry (ADOT)-Operator. Dadurch wird es einfacher, Ihre auf Amazon EKS ausgeführten Anwendungen zu aktivieren, um Metrik- und Nachverfolgungsdaten an mehrere Überwachungsservices wie Amazon CloudWatch, Prometheus und X-Ray zu senden.

Weitere Informationen finden Sie unter Erste Schritte mit AWS Distro für OpenTelemetry unter Verwendung von EKS-Add-Ons in der Dokumentation zu AWS Distro for OpenTelemetry.

Anzeigen aggregierter Daten zu Cluster-Ressourcen mit dem EKS-Dashboard

image::images/eks-dashboard.png[screenshot of account level cluster metrics]

Was ist das Amazon-EKS-Dashboard?

Das Amazon EKS-Dashboard bietet konsolidierten Einblick in Ihre Kubernetes-Cluster in mehreren AWS Regionen und AWS Konten. Mit diesem Dashboard können Sie:

- Verfolgen Sie Cluster, für die innerhalb der nächsten 90 Tage end-of-support automatische Upgrades geplant sind.
- Die Kosten für die EKS-Steuerebene für Cluster mit erweitertem Support ermitteln.
- Cluster mit Erkenntnissen überprüfen, die vor einem Upgrade beachtet werden sollten.
- Verwaltete Knotengruppen identifizieren, auf denen bestimmte AMI-Versionen ausgeführt werden.
- Die Verteilung der Cluster-Support-Typen (Standard-Support im Vergleich zu erweitertem Support) überwachen.

Das EKS-Dashboard ist in EKS Cluster Insights integriert, um Probleme mit Ihren Clustern aufzudecken, wie z. B. die Verwendung von veraltetem Kubernetes. APIs Weitere Informationen finden Sie unter the section called “Cluster-Einblicke”.

> **Note**
>
> Das EKS-Dashboard ist nicht in Echtzeit verfügbar und wird alle 12 Stunden aktualisiert. Informationen zur Echtzeit-Cluster-Überwachung finden Sie unter Cluster überwachen

Wie verwendet das Dashboard AWS Organizations?

Für die Funktionalität des Amazon EKS-Dashboards ist die Integration von AWS Organizations erforderlich. Es nutzt AWS Organizations, um Clusterinformationen auf sichere Weise kontenübergreifend zu sammeln. Diese Integration ermöglicht eine zentrale Verwaltung und Steuerung, wenn Ihre AWS Infrastruktur skaliert.

Wenn AWS Organizations für Ihre Infrastruktur nicht aktiviert ist, finden Sie Anweisungen zur Einrichtung im AWS Organizations-Benutzerhandbuch.

Regions- und kontoübergreifender Zugriff

Das EKS-Dashboard kann Clusterressourcen in jedem Konto anzeigen, das Mitglied der AWS Organisation ist. Informationen zum Generieren einer Liste der AWS Konten in Ihrer Organisation finden Sie unter Exportieren von Details für alle Konten in einer Organisation.

Die AWS Region us-east-1 generiert das Dashboard. Sie müssen sich in dieser Region anmelden, um das Dashboard anzuzeigen. Das Dashboard aggregiert Daten über AWS Regionen hinweg, beinhaltet jedoch keine Regionen in China. `GovCloud`

Wichtige Begriffe

- AWS Organisation: Eine einheitliche Verwaltungsstruktur für mehrere AWS Konten.
- Verwaltungskonto: Das Hauptkonto, das die AWS Organisation kontrolliert.
- Mitgliedskonto: Jedes Konto innerhalb der Organisation mit Ausnahme des Verwaltungskontos.
- Delegierter Administrator: Ein Mitgliedskonto, dem bestimmte kontoübergreifende Administratorrechte gewährt wurden. Innerhalb des Verwaltungskontos können Sie pro AWS Dienst ein delegiertes Administratorkonto auswählen.
- Vertrauenswürdiger Zugriff: Berechtigung für das EKS-Dashboard, auf Cluster-Informationen über Organisationskonten hinweg zuzugreifen.

- Serviceverknüpfte Rolle (SLRs): Eine einzigartige Art von IAM-Rolle, die direkt mit einem Dienst verknüpft ist. AWS Das EKS-Dashboard verwendet eine SLR, um Informationen zu Ihren Konten und Organisationen zu lesen.
- Weitere Informationen finden Sie unter Terminologie und Konzepte im AWS Organizations User Guide.

Allgemeine Übersicht

1. Greifen Sie auf das Verwaltungskonto Ihrer AWS Organisation zu.
   - Die Schritte für den Zugriff auf das Verwaltungskonto hängen davon ab, wie Sie Ihre AWS Organisation konfiguriert haben. Sie können beispielsweise über AWS Identity Center oder Okta auf das Verwaltungskonto zugreifen.
2. Aktivieren Sie den vertrauenswürdigen Zugriff über die EKS-Konsole.
3. Weisen Sie anhand seiner AWS Konto-ID einen delegierten Administrator zu.
4. Wechseln Sie zum Konto des delegierten Administrators.
5. Greifen Sie auf die erweiterte EKS-Konsole mit organisationsweiter Sichtbarkeit zu.

Aktivieren Sie das EKS-Dashboard über die AWS Konsole

> **Important**
>
> Sie müssen mit dem Verwaltungskonto Ihrer AWS Organisation angemeldet sein, um das EKS-Dashboard zu aktivieren.

Zugriff auf EKS-Dashboard-Einstellungen

1. Überprüfen Sie Folgendes:
   a. Sie haben AWS Organizations aktiviert und konfiguriert.
   b. Sie sind beim Verwaltungskonto der Organisation angemeldet.
   c. Sie sehen die AWS Management Console in der Region US-East-1.
2. Navigieren Sie zur EKS-Konsole.
3. Öffnen Sie in der linken Seitenleiste die Dashboard-Einstellungen.

Einrichtung des Zugriffs auf das Amazon-EKS-Dashboard

1. Suchen Sie die AWS Konto-ID des AWS Kontos, dem Sie die Anzeige des EKS-Dashboards ermöglichen möchten.
   a. Dieser Schritt ist optional, wird jedoch empfohlen. Wenn Sie dies nicht tun, können Sie nur über das Verwaltungskonto auf das Dashboard zugreifen. Als bewährte Methode sollten Sie den Zugriff auf das Verwaltungskonto beschränken.
2. Klicken Sie auf Vertrauenswürdigen Zugriff aktivieren.
   a. Sie können das Dashboard nun über das Verwaltungskonto anzeigen.
3. Klicken Sie auf Delegierten Administrator registrieren und geben Sie die Konto-ID des AWS Kontos ein, mit dem Sie das Dashboard aufrufen möchten.
   a. Sie können das Dashboard nun über das Konto des delegierten Administrators oder das Verwaltungskonto anzeigen.

Informationen zu den für die Aktivierung des Dashboards erforderlichen Berechtigungen finden Sie unter Erforderliche Mindest-IAM-Richtlinien.

EKS-Dashboard anzeigen

1. Melden Sie sich beim delegierten Administratorkonto (empfohlen) oder beim Verwaltungskonto an.
2. Melden Sie sich in der Region us-east-1 an.
3. Wechseln Sie zum EKS-Service und wählen Sie in der linken Seitenleiste „Dashboard“ aus.

> **Note**
> Überprüfen Sie die zum Anzeigen des EKS-Dashboards erforderlichen IAM-Berechtigungen.

Dashboard konfigurieren

Sie können die Ansicht des Dashboards konfigurieren und Ressourcen filtern.

Verfügbare Ressourcen

- Clusters: Zeigen Sie aggregierte Informationen zum Status und Standort von EKS-Clustern an.

  - Cluster mit Integritätsproblemen
  - Cluster mit erweitertem EKS-Support.
  - Aufschlüsselung der Cluster nach Kubernetes-Version.
- Verwaltete Knotengruppen: Sehen Sie sich die verwalteten Knotengruppen und EC2 Instanzen an.
  - Knotengruppen nach AMI-Typ, wie beispielsweise Amazon Linux oder Bottlerocket.
  - Probleme mit der Integrität der Knotengruppe.
  - Instance-Typ-Verteilung
- Add-Ons: Erfahren Sie, welche Amazon-EKS-Add-Ons Sie installiert haben und welchen Status sie haben.
  - Anzahl der Installationen pro Add-On.
  - Add-Ons mit Integritätsproblemen.
  - Versionsverteilung pro Add-On.

Verfügbare Ansichten

- Diagrammansicht
  - Eine anpassbare Widget-Ansicht, die Diagramme und Visualisierungen der ausgewählten Ressource anzeigt.
  - Änderungen an der Diagrammansicht, wie beispielsweise das Entfernen eines Widgets, sind für alle Benutzer des EKS-Dashboards sichtbar.
- Ressourcenansicht
  - Eine Listenansicht der ausgewählten Ressource, die Filter unterstützt.
- Kartenansicht
  - Zeigen Sie die geografische Verteilung der ausgewählten Ressource an.

EKS-Dashboard filtern

Sie können das EKS-Dashboard nach folgenden Kriterien filtern:

- AWS Konto
- Organisationseinheit, definiert durch AWS Organizations
- AWS Region

Deaktivieren Sie das EKS-Dashboard über die AWS Konsole

1. Überprüfen Sie Folgendes:
   a. Sie haben AWS Organizations aktiviert und konfiguriert.
   b. Sie sind beim Verwaltungskonto der Organisation angemeldet.
   c. Sie sehen die AWS Management Console in der Region US-East-1.
2. Navigieren Sie zur EKS-Konsole.
3. Öffnen Sie in der linken Seitenleiste die Dashboard-Einstellungen.
4. Klicken Sie auf Vertrauenswürdigen Zugriff deaktivieren.

Fehlerbehebung im EKS-Dashboard

Problem beim Aktivieren des EKS-Dashboards

- Sie müssen mit dem Verwaltungskonto einer AWS Organisation angemeldet sein.
  - Wenn Sie noch keine AWS Organisation haben, erstellen Sie eine. Erfahren Sie, wie Sie eine Organisation erstellen und konfigurieren.
  - Wenn Ihr AWS Konto bereits Mitglied einer AWS Organisation ist, identifizieren Sie den Administrator der Organisation.
- Sie müssen mit dem AWS Konto angemeldet sein und über ausreichende IAM-Berechtigungen verfügen, um AWS Organisationsressourcen erstellen und aktualisieren zu können.

Problem beim Anzeigen des EKS-Dashboards

- Sie müssen bei einem der folgenden AWS Konten angemeldet sein:
  - Das Verwaltungskonto der AWS Organisation
  - Ein delegiertes Administratorkonto, das in den EKS-Dashboard-Einstellungen des Verwaltungskontos angegeben ist.
- Wenn Sie das EKS-Dashboard erst kürzlich aktiviert haben, beachten Sie, dass die anfängliche Dateneingabe bis zu 12 Stunden dauern kann.
- Versuchen Sie, das Dashboard mithilfe der CLI erneut zu aktivieren, einschließlich der Erstellung der serviceverknüpften Rolle.

Dashboard-Widgets verschieben sich unerwartet

- Das EKS-Dashboard speichert die konfigurierbare Widget-Ansicht auf AWS Kontoebene. Wenn Sie die Widget-Ansicht ändern, können andere Personen, die dasselbe AWS Konto verwenden, die Änderungen sehen.

Konfiguration der EKS-Dashboard-Integration mit AWS Organizations

Dieser Abschnitt enthält step-by-step Anweisungen zur Konfiguration der Integration des EKS-Dashboards mit AWS Organizations. Sie erfahren, wie Sie den vertrauenswürdigen Zugriff zwischen Services aktivieren und deaktivieren sowie delegierte Administratorkonten registrieren und deregistrieren können. Jede Konfigurationsaufgabe kann entweder über die AWS Konsole oder die AWS CLI ausgeführt werden.

Den vertrauenswürdigen Zugriff aktivieren

Durch vertrauenswürdigen Zugriff wird das EKS-Dashboard autorisiert, sicher auf Cluster-Informationen aller Konten in Ihrem Unternehmen zuzugreifen.

Mithilfe der AWS Konsole

1. Melden Sie sich beim Verwaltungskonto Ihrer AWS Organisation an.
2. Navigieren Sie zur EKS-Konsole in der Region us-east-1.
3. Wählen Sie in der linken Seitenleiste „Dashboard-Einstellungen“ aus.
4. Klicken Sie auf Vertrauenswürdigen Zugriff aktivieren.

> **Note**
>
> Wenn Sie den vertrauenswürdigen Zugriff über die EKS-Konsole aktivieren, erstellt das System automatisch die serviceverknüpfte `AWSServiceRoleForAmazonEKSDashboard`-Rolle. Diese automatische Erstellung erfolgt nicht, wenn Sie den vertrauenswürdigen Zugriff über die AWS CLI oder die AWS Organisationskonsole aktivieren.

Verwenden der AWS CLI

1. Melden Sie sich beim Verwaltungskonto Ihrer AWS Organisation an.

2. Führen Sie die folgenden Befehle aus:

```
aws iam create-service-linked-role --aws-service-name dashboard.eks.amazonaws.com
aws organizations enable-aws-service-access --service-principal eks.amazonaws.com
```

So deaktivieren Sie den vertrauenswürdigen Zugriff

Durch die Deaktivierung des vertrauenswürdigen Zugriffs wird dem EKS-Dashboard die Berechtigung zum Zugriff auf Cluster-Informationen über die Konten Ihrer Organisation hinweg entzogen.

Verwenden der AWS Konsole

1. Melden Sie sich beim Verwaltungskonto Ihrer AWS Organisation an.
2. Navigieren Sie zur EKS-Konsole in der Region us-east-1.
3. Wählen Sie in der linken Seitenleiste „Dashboard-Einstellungen“ aus.
4. Klicken Sie auf Vertrauenswürdigen Zugriff deaktivieren.

Verwenden der AWS CLI

1. Melden Sie sich beim Verwaltungskonto Ihrer AWS Organisation an.
2. Führen Sie den folgenden Befehl aus:

```
aws organizations disable-aws-service-access --service-principal eks.amazonaws.com
```

Aktivieren eines delegierten Administrators

Ein delegierter Administrator ist ein Benutzerkonto, dem die Berechtigung zum Zugriff auf das EKS-Dashboard gewährt wurde.

Verwenden der AWS Konsole

1. Melden Sie sich beim Verwaltungskonto Ihrer AWS Organisation an.
2. Navigieren Sie zur EKS-Konsole in der Region us-east-1.
3. Wählen Sie in der linken Seitenleiste „Dashboard-Einstellungen“ aus.
4. Klicken Sie auf Delegierten Administrator registrieren.

5. Geben Sie die Konto-ID des AWS Kontos ein, das Sie als delegierten Administrator auswählen möchten.
6. Bestätigen Sie die Registrierung.

Verwenden der AWS CLI

1. Melden Sie sich beim Verwaltungskonto Ihrer AWS Organisation an.
2. Führen Sie den folgenden Befehl aus und ersetzen Sie `123456789012` durch Ihre Konto-ID:

```
aws organizations register-delegated-administrator --account-id 123456789012 --
service-principal eks.amazonaws.com
```

Deaktivierung eines delegierten Administratorkontos

Durch die Deaktivierung eines delegierten Administrators wird die Berechtigung des Kontos für den Zugriff auf das EKS-Dashboard entfernt.

Verwenden der AWS Konsole

1. Melden Sie sich beim Verwaltungskonto Ihrer AWS Organisation an.
2. Navigieren Sie zur EKS-Konsole in der Region us-east-1.
3. Wählen Sie in der linken Seitenleiste „Dashboard-Einstellungen“ aus.
4. Suchen Sie den delegierten Administrator in der Liste.
5. Klicken Sie neben dem Konto, das Sie als delegierten Administrator entfernen möchten, auf Abmelden.

Verwenden der AWS CLI

1. Melden Sie sich beim Verwaltungskonto Ihrer AWS Organisation an.
2. Führen Sie den folgenden Befehl aus und ersetzen Sie dabei `123456789012` durch die Konto-ID des delegierten Administrators:

```
aws organizations deregister-delegated-administrator --account-id 123456789012 --
service-principal eks.amazonaws.com
```

Erforderliche Mindestanforderungen an IAM-Richtlinien

In diesem Abschnitt werden die IAM-Mindestrichtlinien beschrieben, die erforderlich sind, um vertrauenswürdigen Zugriff zu ermöglichen und einen Administrator für die EKS-Dashboard-Integration mit AWS Organizations zu delegieren.

Richtlinie für die Gewährung vertrauenswürdigen Zugriffs

Um den vertrauenswürdigen Zugriff zwischen EKS Dashboard und AWS Organizations zu aktivieren, benötigen Sie die folgenden Berechtigungen:

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "organizations:EnableAWSServiceAccess",
                "organizations:DescribeOrganization",
                "organizations:ListAWSServiceAccessForOrganization"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/
dashboard.eks.amazonaws.com/AWSServiceRoleForAmazonEKSDashboard"
        }
    ]
}
```

Richtlinie zur Übertragung von Administratorrechten

Um einen delegierten Administrator für das EKS-Dashboard zu registrieren oder zu deregistrieren, benötigen Sie die folgenden Berechtigungen:

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": [
                "organizations:RegisterDelegatedAdministrator",
                "organizations:DeregisterDelegatedAdministrator",
                "organizations:ListDelegatedAdministrators"
            ],
            "Resource": "*"
        }
    ]
}
```

Richtlinie zum Anzeigen des EKS-Dashboards

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AmazonEKSDashboardReadOnly",
            "Effect": "Allow",
            "Action": [
                "eks:ListDashboardData",
                "eks:ListDashboardResources",
                "eks:DescribeClusterVersions"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AmazonOrganizationsReadOnly",
            "Effect": "Allow",
            "Action": [
                "organizations:DescribeOrganization",
                "organizations:ListAWSServiceAccessForOrganization",
                "organizations:ListRoots",
                "organizations:ListAccountsForParent",
                "organizations:ListOrganizationalUnitsForParent"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AmazonOrganizationsDelegatedAdmin",
            "Effect": "Allow",
            "Action": [
```

```
                "organizations:ListDelegatedAdministrators"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "organizations:ServicePrincipal": "eks.amazonaws.com"
                }
            }
        }
    ]
}
```

> **Note**
>
> Diese Richtlinien müssen dem IAM-Prinzipal (Benutzer oder Rolle) im Verwaltungskonto Ihrer AWS Organisation zugeordnet sein. Mitgliedskonten können keinen vertrauenswürdigen Zugriff aktivieren oder Administratoren delegieren.

Erweitern Sie EKS mit integrierten AWS Services

Zusätzlich zu den in anderen Abschnitten behandelten Services arbeitet Amazon EKS mit weiteren AWS Services zusammen, um zusätzliche Lösungen bereitzustellen. In diesem Thema werden einiger der weiteren Services aufgeführt, die mithilfe von Amazon EKS ihre Funktionalität erweitern, sowie Services, die Amazon EKS zur Ausführung von Aufgaben nutzt.

Themen

- Sichern Sie Ihre EKS-Cluster mit AWS Backup
- Amazon-EKS-Ressourcen mit AWS CloudFormation erstellen
- Connect zu Git-Repositorys her mit AWS CodeConnections
- Sicherheitsereignissen in EKS mit Amazon Detective analysieren
- Bedrohungen mit Amazon GuardDuty erkennen
- Starten Sie EKS-Cluster mit geringer Latenz in AWS Local Zones.
- Bewertung der Ausfallsicherheit von EKS-Clustern mit AWS Resilience Hub
- Anwendungsgeheimnisse mit AWS Secrets Manager verwalten
- EKS-Sicherheitsdaten mit Security Lake zentralisieren und analysieren
- Sichere Cluster-übergreifende Konnektivität mit Amazon VPC Lattice ermöglichen

Sichern Sie Ihre EKS-Cluster mit AWS Backup

AWS Backup unterstützt Backups von Amazon EKS-Clustern, einschließlich Kubernetes-Clusterstatus und persistentem Speicher, der über einen persistenten Volume-Claim an den EKS-Cluster angehängt ist (EBS-Volumes, EFS-Dateisysteme und S3-Buckets). Ein Amazon EKS-Backup erstellt einen zusammengesetzten Wiederherstellungspunkt, an dem für jede gesicherte Ressource ein untergeordneter Wiederherstellungspunkt eingerichtet wird.

- So sichern Sie Ressourcen: Erste Schritte mit AWS Backup
- Erstellung von Backup nach Ressourcentyp: Amazon EKS Backups
- So stellen Sie Amazon EKS-Cluster wieder her: Amazon EKS Backups wiederherstellen

Um über die EKS-Konsole, AWS Backup-Konsole oder CLI loszulegen, stellen Sie sicher, dass Ihre IAM-Rolle über die folgenden Berechtigungen verfügt:

- Diese finden Sie in der Managed Policy AWSBackupServiceRolePolicyForBackupvon AWS Backup. Dies beinhaltet die erforderlichen Berechtigungen für die Sicherung Ihres Amazon EKS-Clusters und des persistenten EBS- und EFS-Speichers
- Wenn Ihr EKS-Cluster einen S3-Bucket enthält, müssen Sie sicherstellen, dass die folgenden Richtlinien und Voraussetzungen für Ihren S3-Bucket wie dokumentiert hinzugefügt und aktiviert werden:
- AWSBackupServiceRolePolicyForS3 Backup
- Voraussetzungen für S3-Backups

Stellen Sie sicher, dass Ihre EKS-Cluster über die folgenden Einstellungen verfügen:

- Der EKS-Cluster-Autorisierungsmodus ist auf API oder API_AND _CONFIG_MAP für AWS Backup gesetzt, um Zugriffseinträge für den Zugriff auf den EKS-Cluster zu erstellen.

Amazon-EKS-Ressourcen mit AWS CloudFormation erstellen

Amazon EKS ist in AWS CloudFormation integriert. Dies ist ein Service, der Ihnen hilft, Ihre AWS-Ressourcen zu modellieren und einzurichten, so dass Sie weniger Zeit für die Erstellung und Verwaltung Ihrer Ressourcen und Infrastruktur aufwenden müssen. Sie erstellen eine Vorlage, in der alle gewünschten AWS-Ressourcen, wie etwa ein Amazon-EKS-Cluster, beschrieben werden, und AWS CloudFormation übernimmt die Bereitstellung und Konfigurierung dieser Ressourcen für Sie.

Wenn Sie AWS CloudFormation verwenden, können Sie Ihre Vorlage wiederverwenden, um Ihre Amazon-EKS-Ressourcen einheitlich und wiederholt einzurichten. Sie beschreiben Ihre Ressourcen dann einmal und können die gleichen Ressourcen dann in mehreren AWS-Konten und -Regionen immer wieder bereitstellen.

Amazon-EKS- und AWS-CloudFormation-Vorlagen

Um Ressourcen für Amazon EKS und verwandte Services bereitzustellen und zu konfigurieren, müssen Sie AWS-CloudFormation-Vorlagen kennen und verstehen. Vorlagen sind formatierte Textdateien in JSON oder YAML. Diese Vorlagen beschreiben die Ressourcen, die Sie in Ihren AWS-CloudFormation-Stacks bereitstellen möchten. Wenn Sie mit JSON oder YAML nicht vertraut sind, können Sie AWS CloudFormation Designer verwenden, um den Einstieg in AWS-CloudFormation-Vorlagen zu vereinfachen. Weitere Informationen finden Sie unter Was ist AWS CloudFormation Designer? im AWS-CloudFormation-Benutzerhandbuch.

Amazon EKS unterstützt das Erstellen von Clustern und Knotengruppen in AWS CloudFormation. Weitere Informationen, einschließlich Beispiele für JSON- und YAML-Vorlagen für Ihre Amazon-EKS-Ressourcen, finden Sie unter Amazon-EKS-Ressourcentypreferenz im AWS-CloudFormation-Handbuch.

Weitere Informationen zu AWS CloudFormation

Weitere Informationen zu AWS CloudFormation finden Sie in den folgenden Ressourcen:

- AWS CloudFormation
- AWS-CloudFormation-Benutzerhandbuch
- Benutzerhandbuch für die AWS-CloudFormation-Befehlszeilenschnittstelle

Connect zu Git-Repositorys her mit AWS CodeConnections

AWS CodeConnectionsbietet eine sichere Möglichkeit, AWS Dienste mit Quellcode-Repositorys von Drittanbietern zu verbinden. AWS CodeConnections unterstützt GitHub, GitLab, Bitbucket und andere Anbieter. Weitere Informationen und erste Schritte findest du unter Mit Verbindungen arbeiten.

Zur Verwendung AWS CodeConnections mit Argo CD

Wenn Sie die EKS-Funktion für Argo CD verwenden, können Sie wählen, ob Sie die sichere Authentifizierung bei Git-Repositorys aktivieren AWS CodeConnections möchten, ohne langlebige Anmeldeinformationen oder persönliche Zugriffstoken verwalten zu müssen. AWS CodeConnections verarbeitet den OAuth Authentifizierungsablauf und verwaltet die Verbindung zu Ihrem Git-Anbieter und bietet so einen sicheren und verwaltbaren Ansatz für den Zugriff auf Ihre GitOps Repositorys und Anwendungsmanifeste, die bei Git-Drittanbietern gespeichert sind.

Voraussetzungen

- Ein Amazon EKS-Cluster mit der Argo-CD-Funktion wurde erstellt
- Eine Verbindung, die in AWS CodeConnections deinem Git-Anbieter erstellt wurde
- Für die Nutzung der Verbindung durch Argo CD konfigurierte IAM-Berechtigungen

Um den Zugriff auf das CodeConnections Argo-CD-Repository zu konfigurieren

1. Stellen Sie eine Verbindung in der CodeConnections Konsole her:

a. Öffnen Sie die CodeConnections -Konsole.

b. Wählen Sie Create Connection (Verbindung erstellen) aus.

c. Wähle deinen Anbieter (GitHub, GitLab, oder Bitbucket) und folge dem Authentifizierungsablauf.

d. Notieren Sie sich den Verbindungs-ARN zur Verwendung in Ihrer Argo-CD-Konfiguration.

2. Stellen Sie sicher, dass die Argo-CD-Funktionsrolle über die Berechtigungen zur Verwendung der Verbindung mit einer ressourcenbasierten Richtlinie verfügt:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codeconnections:UseConnection",
        "codeconnections:GetConnection"
      ],
      "Resource": "arn:aws:codeconnections:region:account-id:connection/connection-
id"
    }
  ]
}
```

3. Konfigurieren Sie Argo CD so, dass beim Hinzufügen eines Repositorys auf die CodeConnections Ressource verwiesen wird, und verwenden Sie dabei den CodeConnections Ressourcenendpunkt als Repository-URL. Die Argo-CD-Funktion verwendet die Verbindung, um sich bei Ihrem Git-Anbieter zu authentifizieren, ohne dass langlebige Anmeldeinformationen erforderlich sind.

Überlegungen zur Verwendung mit Argo CD CodeConnections

Beachten Sie bei der Verwendung AWS CodeConnections mit der EKS-Funktion für Argo-CD Folgendes:

- Die CodeConnections Verbindung muss sich in derselben AWS Region wie Ihr EKS-Cluster befinden
- Die Argo-CD-Funktionsrolle muss über folgende Berechtigungen `codeconnections:UseConnection` verfügen `codeconnections:GetConnection`
- CodeConnections verwaltet den OAuth Ablauf und den Lebenszyklus der Anmeldeinformationen automatisch

Weitere Informationen zur Konfiguration des Repository-Zugriffs mit Argo CD finden Sie unter. the section called “Repositorys konfigurieren”

Sicherheitsereignissen in EKS mit Amazon Detective analysieren

Amazon Detective hilft Ihnen, die Ursache von Sicherheitserkenntnissen oder verdächtigen Aktivitäten zu analysieren, zu untersuchen und schnell zu identifizieren. Detective sammelt automatisch Protokolldaten von Ihren AWS Ressourcen. Es verwendet dann Machine Learning, statistische Analysen und die Diagrammtheorie, um Visualisierungen zu erstellen, mit denen Sie effektive Sicherheitsuntersuchungen schneller und effizienter durchführen können. Detective bietet vordefinierte Datenaggregationen, Übersichten und Kontexte, mit denen Sie Art und Ausmaß möglicher Sicherheitsprobleme schnell analysieren und feststellen können. Weitere Informationen finden Sie im Benutzerhandbuch von Amazon Detective.

Detective organisiert Kubernetes und AWS Daten zu Ergebnissen wie:

- Amazon EKS-Clusterdetails, einschließlich der IAM-Identität, mit der der Cluster erstellt wurde, und der Servicerolle des Clusters. Sie können die Aktivität AWS und die Kubernetes-API-Aktivität dieser IAM-Identitäten mit Detective untersuchen.
- Container-Details, wie das Image und der Sicherheitskontext. Sie können sich die Angaben auch für beendete Pods überprüfen.
- Kubernetes-API-Aktivität, einschließlich allgemeiner Trends bei der API-Aktivität und Details zu bestimmten API-Aufrufen. Sie können beispielsweise die Anzahl der erfolgreichen und fehlgeschlagenen Kubernetes-API-Aufrufe anzeigen, die in einem ausgewählten Zeitraum ausgegeben wurden. Darüber hinaus kann der Abschnitt über neu beobachtete API-Aufrufe hilfreich sein, um verdächtige Aktivitäten zu identifizieren.

Amazon EKS Audit Logs sind ein optionales Datenquellenpaket, das Ihrem Detective-Verhaltensdiagramm hinzugefügt werden kann. Sie können die verfügbaren optionalen Quellpakete und deren Status in Ihrem Konto einsehen. Weitere Informationen finden Sie unter Amazon-EKS-Auditprotokolle für Detective im Benutzerhandbuch von Amazon Detective.

Verwenden Sie Amazon Detective mit Amazon EKS

Bevor Sie die Ergebnisse überprüfen können, muss Detective für mindestens 48 Stunden in derselben AWS Region aktiviert sein, in der sich Ihr Cluster befindet. Weitere Informationen finden Sie unter Einrichten von Amazon Detective im Benutzerhandbuch von Amazon Detective.

1. Öffnen Sie die Detective-Konsole unter https://console.aws.amazon.com/detective/.
2. Wählen Sie Search (Suche) im linken Navigationsbereich.
3. Wählen Sie Choose type (Typ auswählen) und anschließend EKS cluster (EKS-Cluster) aus.
4. Geben Sie den Clusternamen oder ARN ein und wählen Sie dann Search (Suchen) aus.
5. Wählen Sie in den Suchergebnissen den Namen des Clusters aus, für den Sie Aktivitäten anzeigen möchten. Weitere Informationen darüber, was Sie sich ansehen können, finden Sie unter Allgemeine Kubernetes-API-Aktivität mit einem Amazon-EKS-Cluster im Benutzerhandbuch zu Amazon Detective.

Bedrohungen mit Amazon GuardDuty erkennen

Amazon GuardDuty ist ein Service zur Bedrohungserkennung, der Ihnen hilft, Ihre Konten, Container, Workloads und Daten in Ihrer AWS-Umgebung zu schützen. Mit Machine Learning (ML)-Modellen sowie Funktionen zur Erkennung von Anomalien und Bedrohungen überwacht GuardDuty kontinuierlich verschiedene Protokollquellen und Laufzeitaktivitäten, um potenzielle Sicherheitsrisiken und böswillige Aktivitäten in Ihrer Umgebung zu identifizieren und zu priorisieren.

Neben anderen Features bietet GuardDuty die folgenden zwei Features, die potenzielle Bedrohungen für Ihre EKS-Cluster erkennen: EKS-Schutz und Laufzeit-Überwachung.

> **Note**
>
> Neu: Amazon EKS Auto Mode ist in GuardDuty integriert.

EKS-Schutz

Dieses Feature bietet Schutz vor Bedrohungen, indem es die zugehörigen Kubernetes-Auditprotokolle überwacht und so zum Schutz Ihrer Amazon-EKS-Cluster beiträgt. Kubernetes-Auditprotokolle erfassen sequenzielle Aktionen innerhalb Ihres Clusters, einschließlich Aktivitäten von Benutzern, Anwendungen, welche die Kubernetes-API verwenden, und der Steuerebene. GuardDuty kann beispielsweise erkennen, dass APIs, die möglicherweise zur Manipulation von Ressourcen in einem Kubernetes-Cluster aufgerufen wurden, von einem nicht authentifizierten Benutzer aufgerufen wurden.

Wenn Sie EKS Protection aktivieren, kann GuardDuty ausschließlich zum Zwecke der kontinuierlichen Bedrohungserkennung auf Ihre Amazon-EKS-Auditprotokolle zugreifen.

Wenn GuardDuty eine potenzielle Bedrohung für Ihren Cluster identifiziert, generiert es einen entsprechenden Kubernetes-Audit-Protokoll-Befund eines bestimmten Typs. Weitere Informationen zu den in Kubernetes-Auditprotokollen verfügbaren Ergebnistypen finden Sie unter Befundtypen für Kubernetes-Auditprotokolle im Amazon GuardDuty-Benutzerhandbuch.

Weitere Informationen finden Sie unter EKS-Schutz im Amazon-GuardDuty-Benutzerhandbuch.

Laufzeitüberwachung

Dieses Feature überwacht und analysiert Ereignisse auf Betriebssystemebene, im Netzwerk und in Dateien, um Ihnen dabei zu helfen, potenzielle Bedrohungen in bestimmten AWS-Workloads in Ihrer Umgebung zu erkennen.

Wenn Sie die Laufzeitüberwachung aktivieren und den GuardDuty-Agenten in Ihren Amazon-EKS-Clustern installieren, beginnt GuardDuty mit der Überwachung der diesem Cluster zugeordneten Laufzeitereignisse. Beachten Sie, dass der GuardDuty-Agent und die Laufzeitüberwachung für Amazon EKS Hybrid Nodes nicht verfügbar sind. Daher ist die Laufzeitüberwachung für Laufzeitereignisse, die auf Ihren Hybridknoten auftreten, nicht verfügbar. Wenn GuardDuty eine potenzielle Bedrohung für Ihren Cluster identifiziert, generiert es einen entsprechenden Befund zur Laufzeitüberwachung. Eine Bedrohung kann beispielsweise dadurch entstehen, dass ein einzelner Container kompromittiert wird, auf dem eine anfällige Web-Anwendung ausgeführt wird. Diese Web-Anwendung verfügt möglicherweise über Zugriffsberechtigungen für die zugrunde liegenden Container und Workloads. In diesem Fall könnten falsch konfigurierte Anmeldedaten möglicherweise zu einem umfassenderen Zugriff auf das Konto und die darin gespeicherten Daten führen.

Um die Laufzeitüberwachung zu konfigurieren, installieren Sie den GuardDuty-Agenten als Amazon-EKS-Add-On auf Ihrem Cluster. Weitere Informationen zu Add-Ons finden Sie unter the section called “ AWS Add-Ons”.

Weitere Informationen finden Sie unter Laufzeitüberwachung im Amazon-GuardDuty-Benutzerhandbuch.

Starten Sie EKS-Cluster mit geringer Latenz in AWS Local Zones.

Eine AWS Local Zone ist die Erweiterung einer AWS-Region in geografischer Nähe zu Ihren Benutzern. Lokale Zonen haben ihre eigenen Verbindungen mit dem Internet und unterstützen AWS Direct Connect. Ressourcen, die in einer Local Zone erstellt wurden, können von lokalen

Benutzern mit sehr latenzarmen Verbindungen genutzt werden. Weitere Informationen finden Sie im Benutzerhandbuch zu AWS Local Zones und im Amazon-EC2-Benutzerhandbuch unter Local Zones.

Amazon EKS unterstützt bestimmte Ressourcen in Local Zones. Dies umfasst verwaltete Knotengruppen, selbstverwaltete Amazon-EC2-Knoten, Amazon-EBS-Volumes und Application Load Balancers (ALBs). Es wird empfohlen, Folgendes zu berücksichtigen, wenn Sie Local Zones als Teil Ihres Amazon-EKS-Clusters verwenden.

- Mit Amazon EKS können Sie keine Fargate-Knoten in Local Zones erstellen.
- Die von Amazon EKS verwaltete Kubernetes Steuerungsebene läuft immer imAWSRegion : Die von Amazon EKS verwaltete Kubernetes Steuerebene kann nicht in der Local Zone ausgeführt werden. Da Local Zones in Ihrer VPC als Subnetz angezeigt werden, sieht Kubernetes Ihre lokalen Zonenressourcen als Teil dieses Subnetzes.
- Der Amazon EKS Kubernetes-Cluster kommuniziert mit den Amazon EC2 Instances, die Sie imAWSRegion oder lokale Zone mit verwaltetem Amazon EKSElastic Network-Schnittstelleaus. Weitere Informationen zur Amazon EKS Network-Architektur finden Sie unterNetzwerk konfigurierenaus.
- Im Gegensatz zu regionalen Subnetzen kann Amazon EKS keine Netzwerkschnittstellen in Ihre Local-Zone-Subnetze platzieren. Das bedeutet, dass Sie beim Erstellen des Clusters keine Subnetze der lokalen Zone angeben dürfen. Es ist jedoch möglich, Worker-Knoten in verschiedenen Local Zones mit demselben Cluster zu verbinden.

Bewertung der Ausfallsicherheit von EKS-Clustern mit AWS Resilience Hub

AWS Resilience Hub bewertet die Ausfallsicherheit eines Amazon-EKS-Clusters durch Analyse seiner Infrastruktur. AWS Resilience Hub verwendet die rollenbasierte Konfiguration für Zugriffskontrolle (RBAC) von Kubernetes, um die in Ihrem Cluster bereitgestellten Kubernetes-Workloads zu bewerten. Weitere Informationen finden Sie unter Aktivieren des AWS-Resilience-Hub-Zugriffs in Ihren Amazon-EKS-Cluster im Benutzerhandbuch zu AWS Resilience Hub.

Anwendungsgeheimnisse mit AWS Secrets Manager verwalten

AWS Secrets Manager hilft Ihnen dabei, Anmeldeinformationen, API-Schlüssel und andere Geheimnisse während ihres gesamten Lebenszyklus zu verwalten, darauf zuzugreifen und sie zu rotieren. Mit Secrets Manager können Sie Geheimnisse sichern und verwalten, die für den Zugriff auf

Ressourcen in AWS, auf Drittanbieterdiensten und vor Ort verwendet werden. Weitere Informationen finden Sie im AWS Secrets Manager Manager-Benutzerhandbuch.

Wenn Sie die EKS-Funktion für Argo CD verwenden, bietet Secrets Manager eine sichere Möglichkeit, Git-Repository-Anmeldeinformationen zu speichern und abzurufen, ohne sensible Daten in Ihrer Argo-CD-Konfiguration und Ihren Ressourcen fest zu codieren. Diese Integration ist besonders nützlich für die Verwaltung von Zugriffstoken und SSH-Schlüsseln für private Repositorys, die von Argo CD verwendet werden, um Anwendungen aus Git-Repositorys zu synchronisieren.

Verwenden Sie AWS Secrets Manager mit Argo CD

Wenn Sie die EKS-Capability for Argo-CD verwenden, können Sie die Anmeldeinformationen für das Git-Repository in Secrets Manager speichern und Argo CD so konfigurieren, dass sie abgerufen werden. Dieser Ansatz ist sicherer als das direkte Speichern von Anmeldeinformationen in der Argo-CD-Konfiguration oder die Verwendung langlebiger persönlicher Zugriffstoken.

Voraussetzungen

- Ein Amazon EKS-Cluster mit aktivierter Argo-CD-Funktion
- In AWS Secrets Manager gespeicherte Git-Repository-Anmeldeinformationen
- Für Argo CD konfigurierte IAM-Berechtigungen für den Zugriff auf Secrets Manager

So konfigurieren Sie Argo CD für die Verwendung von Secrets Manager für Repository-Anmeldeinformationen

1. Speichern Sie Ihre Git-Anmeldeinformationen in Secrets Manager. Um beispielsweise ein GitHub persönliches Zugriffstoken zu speichern:

```
aws secretsmanager create-secret \
  --name argocd/github-token \
  --secret-string '{"username":"git","password":"ghp_xxxxxxxxxxxx"}'
```

2. Stellen Sie sicher, dass die Argo-CD-Capability-Rolle berechtigt ist, das Geheimnis abzurufen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
```

```
        "secretsmanager:GetSecretValue",
        "secretsmanager:DescribeSecret"
      ],
      "Resource": "arn:aws:secretsmanager:region:account-id:secret:argocd/github-
token*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt"
      ],
      "Resource": "arn:aws:kms:region:account-id:key/*",
      "Condition": {
        "StringLike": {
          "kms:EncryptionContext:SecretARN": "arn:aws:secretsmanager:region:account-
id:secret:argocd/*",
          "kms:ViaService": "secretsmanager.*.amazonaws.com"
        }
      }
    }
  ]
}
```

> **Note**
>
> Die KMS-Entschlüsselungsberechtigung ist erforderlich, da Secrets Manager alle Geheimnisse mit AWS KMS verschlüsselt. Die Bedingung beschränkt die Entschlüsselung nur auf Geheimnisse mit dem Präfix. `argocd/` Wenn Sie den AWS verwalteten Standardschlüssel für Secrets Manager verwenden, ist diese Berechtigung ausreichend. Aktualisieren Sie das `Resource` Feld für vom Kunden verwaltete KMS-Schlüssel mit Ihrem spezifischen Schlüssel-ARN.

3. Konfigurieren Sie Argo CD so, dass es die Anmeldeinformationen von Secrets Manager verwendet. Informationen zum Synchronisieren von Geheimnissen aus Secrets Manager mit Kubernetes-Geheimnissen, auf die Argo CD verweisen kann, finden Sie unter Secret Management in der Argo-CD-Dokumentation.
4. Erstellen Sie eine Argo-CD-Repository-Konfiguration, die auf den geheimen ARN verweist:

```
apiVersion: v1
kind: Secret
```

```
metadata:
  name: private-repo
  namespace: argocd
  labels:
    argocd.argoproj.io/secret-type: repository
stringData:
  type: git
  url: https://github.com/org/repo
  secretArn: arn:aws:secretsmanager:region:account-id:secret:argocd/github-token
```

Weitere Informationen zur Konfiguration des Repository-Zugriffs mit Argo CD finden Sie unter. the section called “Repositorys konfigurieren”

EKS-Sicherheitsdaten mit Security Lake zentralisieren und analysieren

Amazon Security Lake ist ein vollständig verwalteter Data-Lake-Service für Sicherheitsdaten, mit dem Sie Sicherheitsdaten aus verschiedenen Quellen, einschließlich Amazon EKS, zentralisieren können. Durch die Integration von Amazon EKS mit Security Lake erhalten Sie tiefere Einblicke in die Aktivitäten, die in Ihren Kubernetes-Ressourcen ausgeführt werden, und können die Sicherheitslage Ihrer Amazon-EKS-Cluster verbessern.

> **Note**
>
> Weitere Informationen zur Verwendung von Security Lake mit Amazon EKS und zur Einrichtung von Datenquellen finden Sie in der Dokumentation zu Amazon Security Lake.

Vorteile der Verwendung von Security Lake mit Amazon Amazon EKS

Zentralisierte Sicherheitsdaten — Security Lake sammelt und zentralisiert automatisch Sicherheitsdaten aus Ihren Amazon EKS-Clustern zusammen mit Daten von anderen AWS Diensten, SaaS-Anbietern, lokalen Quellen und Quellen von Drittanbietern. Dies bietet einen umfassenden Überblick über die Sicherheitslage Ihres gesamten Unternehmens.

Standardisiertes Datenformat – Security Lake konvertiert die erfassten Daten in das Format Open Cybersecurity Schema Framework (OCSF), ein standardisiertes Open-Source-Schema. Diese

Normalisierung ermöglicht eine einfachere Analyse und Integration mit anderen Sicherheitstools und -services.

Verbesserte Bedrohungserkennung – Durch die Analyse der zentralisierten Sicherheitsdaten, einschließlich der Protokolle der Amazon-EKS-Steuerebene, können Sie potenziell verdächtige Aktivitäten innerhalb Ihrer Amazon-EKS-Cluster effektiver erkennen. Dies hilft, Sicherheitsvorfälle schnell zu erkennen und darauf zu reagieren.

Vereinfachte Datenverwaltung – Security Lake verwaltet den Lebenszyklus Ihrer Sicherheitsdaten mit anpassbaren Aufbewahrungs- und Replikationseinstellungen. Dies vereinfacht die Datenverwaltungsaufgaben und stellt sicher, dass Sie die erforderlichen Daten für Compliance- und Auditzwecke aufbewahren.

Aktivierung von Security Lake für Amazon EKS

1. Aktivieren Sie die Amazon-EKS-Steuerebene-Protokollierung für Ihre EKS-Cluster. Ausführliche Anweisungen finden Sie unter Aktivierung und Deaktivierung von Steuerebene-Protokollen.
2. Fügen Sie Amazon-EKS-Auditprotokolle als Quelle in Security Lake hinzu. Security Lake beginnt dann mit der Erfassung detaillierter Informationen zu den Aktivitäten, die in den in Ihren EKS-Clustern ausgeführten Kubernetes-Ressourcen ausgeführt werden.
3. Konfigurieren Sie die Aufbewahrungs- und Replikationseinstellungen für Ihre Sicherheitsdaten in Security Lake basierend auf Ihren Anforderungen.
4. Verwenden Sie die in Security Lake gespeicherten normalisierten OCSF-Daten für die Reaktion auf Vorfälle, Sicherheitsanalysen und die Integration mit anderen AWS Diensten oder Tools von Drittanbietern. Beispielsweise können Sie mithilfe von Amazon Ingestion Sicherheitsinformationen aus Amazon Security Lake-Daten generieren. OpenSearch

Analyse von EKS-Protokollen in Security Lake

Security Lake normalisiert EKS-Protokollereignisse in das OCSF-Format, wodurch die Analyse und Korrelation der Daten mit anderen Sicherheitsereignissen vereinfacht wird. Sie können verschiedene Tools und Dienste wie Amazon Athena, Amazon oder Sicherheitsanalysetools von Drittanbietern verwenden QuickSight, um die normalisierten Daten abzufragen und zu visualisieren.

Weitere Informationen zur OCSF-Zuordnung für EKS-Protokollereignisse finden Sie in den https://github.com/ocsf/examples/tree/main/mappings/markdown/AWS/v1.1.0/EKS Audit Logs [Zuordnungsreferenz] im OCSF-Repository. GitHub

Sichere Cluster-übergreifende Konnektivität mit Amazon VPC Lattice ermöglichen

Amazon VPC Lattice ist ein vollständig verwalteter Anwendungsnetzwerkservice, der direkt in die AWS-Netzwerkinfrastruktur integriert ist und den Sie verwenden können, um Ihre Services über mehrere Konten und Virtual Private Clouds (VPCs) hinweg zu verbinden, zu schützen und zu überwachen. Mit Amazon EKS können Sie Amazon VPC Lattice mithilfe des AWS Gateway API Controller, einer Implementierung der Kubernetes Gateway-API, nutzen. Mit Amazon VPC Lattice können Sie auf einfache und konsistente Weise eine Cluster-übergreifende Konnektivität mit Standard-Kubernetes-Semantik einrichten. Informationen zum Einstieg in die Verwendung von Amazon VPC Lattice mit Amazon EKS finden Sie im AWS-Gateway-API-Controller-Benutzerhandbuch.

Verbinden Sie einen Kubernetes-Cluster mit einer Amazon-EKS–Managementkonsole mit Amazon EKS Connector.

Sie können den Amazon EKS Connector verwenden, um alle konformen Kubernetes-Cluster zu registrieren und mit AWS zu verbinden und in der Amazon-EKS-Konsole visualisieren. Nachdem ein Cluster verbunden wurde, können Sie den Status, die Konfiguration und die Workloads für diesen Cluster in der Amazon-EKS-Konsole anzeigen. Sie können dieses Feature verwenden, um verbundene Cluster in der Amazon-EKS-Konsole anzuzeigen, aber Sie können sie nicht verwalten. Der Amazon EKS Connector erfordert einen Agent, der ein Open-Source-Projekt auf Github ist. Weitere technische Inhalte, darunter Häufig gestellte Fragen und Fehlerbehebung, finden Sie unter the section called “Fehlerbehebung für EKS Connector”.

Der Amazon EKS Connector kann die folgenden Typen von Kubernetes-Clustern mit Amazon EKS verbinden.

- On-Premises-Kubernetes-Cluster
- Selbstverwaltete Cluster, die auf Amazon EC2 ausgeführt werden
- Verwaltete Cluster von anderen Cloud-Anbietern

Überlegungen zum Amazon EKS Connector

Bevor Sie Amazon EKS Connector verwenden, sollten Sie Folgendes verstehen:

- Sie müssen über Administratorrechte für den Kubernetes-Cluster verfügen, um den Cluster mit Amazon EKS zu verbinden.
- Im Kubernetes-Cluster müssen Linux-64-Bit-Worker-Knoten (x86) vorhanden sein, bevor eine Verbindung hergestellt werden kann. ARM-Worker-Knoten werden nicht unterstützt.
- In Ihrem Kubernetes-Cluster müssen Worker-Knoten vorhanden sein, die ausgehenden Zugriff auf die `ssm.`- und `ssmmessages.`-Systems Manager-Endpunkte haben. Weitere Informationen finden Sie unter Systems-Manager-Endpunkte in der Allgemeinen AWS-Referenz.
- Standardmäßig können Sie bis zu zehn Cluster in einer Region verbinden. Sie können eine Erhöhung über die Servicekontingentkonsole anfordern. Weitere Informationen finden Sie unter Beantragen einer Kontingenterhöhung.
- Für externe Kubernetes-Cluster werden nur die Amazon EKS `RegisterCluster`-, `ListClusters`-, `DescribeCluster`- und `DeregisterCluster`-APIs unterstützt.

- Sie müssen über die folgenden Berechtigungen verfügen, um einen Cluster zu registrieren:
  - eks:RegisterCluster
  - ssm:CreateActivation
  - ssm:DeleteActivation
  - iam: PassRole
- Sie müssen über die folgenden Berechtigungen verfügen, um die Registrierung eines Clusters aufzuheben:
  - eks:DeregisterCluster
  - ssm:DeleteActivation
  - ssm:DeregisterManagedInstance

Erforderliche IAM-Rollen für Amazon EKS Connector

Die Verwendung des Amazon EKS Connectors erfordert die folgenden zwei IAM-Rollen:

- Die serviceverknüpfte Rolle Amazon EKS Connector wird bei der ersten Registrierung eines Clusters erstellt.
- Sie müssen die IAM-Rolle des Amazon-EKS-Connector-Agents erstellen. Details dazu finden Sie unter the section called “Konnektor-IAM-Rolle”.

Um die Cluster- und Workload-Ansichtsberechtigung für IAM-Prinzipale zu aktivieren, müssen Sie die Clusterrollen `eks-connector` und Amazon EKS Connector auf Ihren Cluster anwenden. Befolgen Sie die Schritte unter Zugriff gewähren, um Kubernetes-Cluster-Ressourcen auf einer Amazon-EKS-Konsole anzuzeigen.

Verbinden Sie einen externen Kubernetes-Cluster mit der Amazon-EKS-Managementkonsole

Sie können einen externen Kubernetes-Cluster mit Amazon EKS verbinden, indem Sie mehrere Methoden im folgenden Prozess verwenden. Dieser Prozess umfasst zwei Schritte: die Registrierung des Clusters bei Amazon EKS und die Installation des `eks-connector`-Agents im Cluster.

> **Important**
>
> Der zweite Schritt muss innerhalb von 3 Tagen nach Abschluss des ersten Schritts erfolgen, bevor die Registrierung abläuft.

Überlegungen

Sie können YAML-Manifeste bei der Installation des Agenten verwenden. Alternativ können Sie Helm verwenden, wenn Sie den Cluster mit der Befehlszeilenschnittstelle AWS-Managementkonsole oder AWS registrieren. Sie können Helm jedoch nicht zum Installieren des Agenten verwenden, wenn Sie den Cluster mit `eksctl` registrieren.

Voraussetzungen

- Stellen Sie sicher, dass die Agent-Rolle von Amazon EKS Connector erstellt wurde. Befolgen Sie die Schritte unter Erstellen der Konnektor-Agent-Rolle für Amazon EKS.
- Sie müssen über die folgenden Berechtigungen verfügen, um einen Cluster zu registrieren:
  - `eks:RegisterCluster`
  - `ssm:CreateActivation`
  - `ssm:DeleteActivation`
  - `iam:PassRole`

Schritt 1: Registrieren des Clusters

Um einen Cluster beim Amazon EKS Connector zu registrieren, können Sie eines dieser Tools verwenden:

- the section called “AWS-CLI”
- the section called “AWS-Managementkonsole”
- the section called “`eksctl`”

AWS-CLI

1. Die AWS-CLI muss installiert sein. Informationen zur Installation oder Aktualisierung finden Sie unter Installieren der AWS-CLI.

2. Geben Sie für die Connector-Konfiguration Ihre Amazon-EKS-Connector-Agent-IAM-Rolle an. Weitere Informationen finden Sie unter the section called “Erforderliche IAM-Rollen für Amazon EKS Connector”.

```
aws eks register-cluster \
     --name my-first-registered-cluster \
     --connector-config roleArn=arn:aws:iam::111122223333:role/
AmazonEKSConnectorAgentRole,provider="OTHER" \
     --region aws-region
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "cluster": {
        "name": "my-first-registered-cluster",
        "arn": "arn:aws:eks:region:111122223333:cluster/my-first-registered-cluster",
        "createdAt": 1627669203.531,
        "ConnectorConfig": {
            "activationId": "xxxxxxxxACTIVATION_IDxxxxxxxx",
            "activationCode": "xxxxxxxxACTIVATION_CODExxxxxxxx",
            "activationExpiry": 1627672543.0,
            "provider": "OTHER",
            "roleArn": "arn:aws:iam::111122223333:role/AmazonEKSConnectorAgentRole"
        },
        "status": "CREATING"
    }
}
```

Sie verwenden die Werte `aws-region`, `activationId` und `activationCode` im nächsten Schritt.

AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie Add cluster (Cluster hinzufügen) und Register (Registrieren) aus, um die Konfigurationsseite aufzurufen.
3. Füllen Sie im Abschnitt Cluster konfigurieren die folgenden Felder aus:
   - Name – Ein eindeutiger Name für Ihren Cluster.

- Provider (Provider) – Hiermit zeigen Sie die Dropdown-Liste der Kubernetes-Clusteranbieter an. Wenn Sie den spezifischen Anbieter nicht kennen, wählen Sie Andere.
- Rolle EKS-Connector – Wählen Sie die Rolle aus, die für die Verbindung mit dem Cluster verwendet werden soll.

4. Wählen Sie Cluster registrieren aus.
5. Die Seite Cluster-Übersicht wird angezeigt. Wenn Sie das Helm-Chart verwenden möchten, kopieren Sie den Befehl `helm install` und fahren Sie mit dem nächsten Schritt fort. Wenn Sie das YAML-Manifest verwenden möchten, wählen Sie YAML-Datei herunterladen aus, um die Manifestdatei auf Ihr lokales Laufwerk herunterzuladen.

   > **Important**
   >
   > Dies ist Ihre einzige Möglichkeit, den Befehl `helm install` zu kopieren oder diese Datei herunterzuladen. Navigieren Sie nicht von dieser Seite weg, da auf den Link nicht zugegriffen werden kann und Sie den Cluster abmelden und die Schritte von vorne beginnen müssen.

   Der Befehl bzw. die Manifestdatei kann nur einmal für den registrierten Cluster verwendet werden. Wenn Sie Ressourcen aus dem Kubernetes-Cluster löschen, müssen Sie den Cluster erneut registrieren und eine neue Manifestdatei abrufen.

Fahren Sie mit dem nächsten Schritt fort, um die Manifestdatei auf Ihren Kubernetes-Cluster anzuwenden.

`eksctl`

1. `eksctl`-Version `0.68` oder höher muss installiert sein. Informationen zur Installation oder Aktualisierung finden Sie unter the section called “Cluster erstellen (eksctl)”.
2. Registrieren Sie den Cluster unter Angabe eines Namens, eines Anbieters und einer Region.

   ```
   eksctl register cluster --name my-cluster --provider my-provider --region region-code
   ```

   Beispielausgabe:

   ```
   2021-08-19 13:47:26 [#]  creating IAM role "eksctl-20210819194112186040"
   ```

```
2021-08-19 13:47:26 [#]  registered cluster "<name>" successfully
2021-08-19 13:47:26 [#]  wrote file eks-connector.yaml to <current directory>
2021-08-19 13:47:26 [#]  wrote file eks-connector-clusterrole.yaml to <current
 directory>
2021-08-19 13:47:26 [#]  wrote file eks-connector-console-dashboard-full-access-
group.yaml to <current directory>
2021-08-19 13:47:26 [!]  note: "eks-connector-clusterrole.yaml" and "eks-connector-
console-dashboard-full-access-group.yaml" give full EKS Console access to IAM
 identity "<aws-arn>", edit if required; read https://eksctl.io/usage/eks-connector
 for more info
2021-08-19 13:47:26 [#]  run `kubectl apply -f eks-connector.yaml,eks-connector-
clusterrole.yaml,eks-connector-console-dashboard-full-access-group.yaml` before
 expiry> to connect the cluster
```

Dadurch werden Dateien auf Ihrem lokalen Computer erstellt. Diese Dateien müssen innerhalb von 3 Tagen auf den externen Cluster angewendet werden, sonst läuft die Registrierung ab.

3. Wenden Sie in einem Terminal, das Zugriff auf den Cluster hat, die Datei `eks-connector-binding.yaml` an:

```
kubectl apply -f eks-connector-binding.yaml
```

Schritt 2: Installieren des **`eks-connector`**-Agents

Um den `eks-connector`-Agenten zu installieren, verwenden Sie eines der folgenden Tools:

- the section called “Helm”
- the section called “yaml”

Helm

> **Note**
>
> Wenn Sie den Cluster mit `eksctl` registriert haben, verwenden Sie die YAML-Manifest-Methode anstelle der Helm-Chart-Methode.

1. Wenn Sie im vorherigen Schritt die AWS-CLI verwendet haben, ersetzen Sie `ACTIVATION_CODE` und `ACTIVATION_ID` im folgenden Befehl durch die Werte `activationId` bzw.

`activationCode`. Ersetzen Sie die `aws-region` durch die AWS-Region, die Sie im vorherigen Schritt verwendet haben. Führen Sie dann den Befehl aus, um den `eks-connector`-Agent auf dem registrierenden Cluster zu installieren:

```
$ helm install eks-connector \
  --namespace eks-connector \
  oci://public.ecr.aws/eks-connector/eks-connector-chart \
  --set eks.activationCode=ACTIVATION_CODE \
  --set eks.activationId=ACTIVATION_ID \
  --set eks.agentRegion=aws-region
```

Wenn Sie im vorherigen Schritt die AWS-Managementkonsole verwendet haben, verwenden Sie den Befehl, den Sie aus dem vorherigen Schritt kopiert haben und in den diese Werte eingetragen sind.

2. Überprüfen Sie den Zustand der installierten `eks-connector`-Bereitstellung und warten Sie, bis der Status des registrierten Clusters in Amazon EKS `ACTIVE` lautet.

yaml

Schließen Sie die Verbindung ab, indem Sie die Amazon-EKS-Connector-Manifestdatei auf Ihren Kubernetes-Cluster anwenden. Dazu müssen Sie die zuvor beschriebenen Methoden verwenden. Wenn das Manifest nicht innerhalb von drei Tagen angewendet wird, läuft die Registrierung für Amazon EKS Connector ab. Wenn die Clusterverbindung abläuft, muss die Registrierung des Clusters aufgehoben werden, bevor der Cluster erneut verbunden wird.

1. Laden Sie die Amazon EKS Connector YAML-Datei herunter.

```
curl -O https://amazon-eks.s3.us-west-2.amazonaws.com/eks-connector/manifests/eks-connector/latest/eks-connector.yaml
```

2. Bearbeiten Sie die YAML-Datei von Amazon EKS Connector und ersetzen Sie alle Verweise auf `%AWS_REGION%`, `%EKS_ACTIVATION_ID%`, `%EKS_ACTIVATION_CODE%` durch die `aws-region`, `activationId` und `activationCode` aus der Ausgabe des vorherigen Schritts.

   Der folgende Beispielbefehl kann diese Werte ersetzen.

```
sed -i "s~%AWS_REGION%~$aws-region~g; s~%EKS_ACTIVATION_ID%~$EKS_ACTIVATION_ID~g; s~%EKS_ACTIVATION_CODE%~$(echo -n $EKS_ACTIVATION_CODE | base64)~g" eks-connector.yaml
```

> **Important**
>
> Stellen Sie sicher, dass Ihr Aktivierungscode das base64-Format hat.

3. In einem Terminal, das auf den Cluster zugreifen kann, können Sie jetzt die aktualisierte Manifestdatei mit dem folgenden Befehl anwenden:

```
kubectl apply -f eks-connector.yaml
```

4. Nachdem die YAML-Dateien des Amazon-EKS-Connector-Manifests und der Rollenbindung auf Ihren Kubernetes-Cluster angewendet wurden, bestätigen Sie, dass der Cluster jetzt verbunden ist.

```
aws eks describe-cluster \
     --name "my-first-registered-cluster" \
     --region AWS_REGION
```

   Die Ausgabe sollte `status=ACTIVE` enthalten.

5. (Optional) Fügen Sie Ihrem Cluster Tags hinzu. Weitere Informationen finden Sie unter the section called “Markieren Ihrer -Ressourcen”.

Nächste Schritte

Falls Sie Probleme mit diesen Schritten haben, finden Sie weitere Informationen unter the section called “Fehlerbehebung für EKS Connector”.

Informationen zum Gewähren des Zugriffs zusätzlicher IAM-Prinzipale in der Amazon-EKS-Konsole zum Anzeigen von Kubernetes-Ressourcen in einem verbundenen Cluster finden Sie unter the section called “Zugriff auf Cluster gewähren”.

Gewähren Sie Zugriff auf die Anzeige von Kubernetes-Cluster-Ressourcen in einer Amazon-EKS-Konsole.

Gewähren Sie zusätzlichen IAM-Prinzipalen Zugriff auf die Amazon-EKS-Konsole, um Informationen zu Kubernetes-Ressourcen anzuzeigen, die auf Ihrem verbundenen Cluster ausgeführt werden.

Voraussetzungen

Der IAM-Prinzipal die Sie für den Zugriff auf das AWS-Managementkonsole verwenden, muss die folgenden Anforderungen erfüllen:

- Er muss über die `eks:AccessKubernetesApi`-IAM-Berechtigung verfügen.
- Das Amazon-EKS-Connector-Service-Konto kann den IAM-Prinzipal im Cluster imitieren. Dadurch kann Amazon EKS Connector den IAM-Prinzipal einem Kubernetes-Benutzer zuordnen.

So erstellen Sie die Cluster-Rolle für Amazon EKS Connector und wenden sie an

1. Laden Sie die `eks-connector`-Clusterrollenvorlage herunter.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/eks-connector/manifests/eks-connector-console-roles/eks-connector-clusterrole.yaml
```

2. Bearbeiten Sie die YAML-Datei der Clusterrollen-Vorlage. Ersetzen Sie die Referenzen von `%IAM_ARN%` durch den Amazon-Ressourcennamen (ARN) Ihres IAM-Prinzipals.
3. Wenden Sie die Amazon-EKS-Connector-Clusterrolle YAML auf Ihren Kubernetes-Cluster an.

```
kubectl apply -f eks-connector-clusterrole.yaml
```

Damit ein IAM-Prinzipal Kubernetes-Ressourcen in der Amazon-EKS-Konsole anzeigen kann, muss der Prinzipal mit einem Kubernetes `role` oder `clusterrole` verknüpft sein und über die erforderlichen Berechtigungen zum Lesen der Ressourcen verfügen. Weitere Informationen finden Sie unter Using RBAC Authorization in der Kubernetes-Dokumentation.

Konfigurieren Sie einen IAM-Prinzipal für den Zugriff auf den verbundenen Cluster wie folgt

1. Sie können einer dieser Beispielmanifestdateien herunterladen, um eine `clusterrole` und `clusterrolebinding` oder eine `role` und `rolebinding` zu erstellen:

   Kubernetes-Ressourcen in allen Namespaces anzeigen

   - Die `eks-connector-console-dashboard-full-access-clusterrole`-Clusterrolle ermöglicht den Zugriff auf alle Namespaces und Ressourcen, die in der Konsole visualisiert werden können. Sie können den Namen des `role`, `clusterrole` und der entsprechenden Bindung ändern, bevor Sie ihn auf Ihren Cluster anwenden. Verwenden Sie den folgenden Befehl, um eine Beispieldatei herunterzuladen.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/eks-connector/manifests/
eks-connector-console-roles/eks-connector-console-dashboard-full-access-
group.yaml
```

Kubernetes-Ressourcen in einem bestimmten Namespace anzeigen

- Der Namespace in dieser Datei ist `default`. Wenn Sie also einen anderen Namespace angeben möchten, bearbeiten Sie die Datei, bevor Sie sie auf Ihren Cluster anwenden. Verwenden Sie den folgenden Befehl, um eine Beispieldatei herunterzuladen.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/eks-connector/manifests/
eks-connector-console-roles/eks-connector-console-dashboard-restricted-access-
group.yaml
```

2. Bearbeiten Sie die YAML-Datei mit vollem oder eingeschränktem Zugriff, um die Referenzen von `%IAM_ARN%` durch den Amazon-Ressourcennamen (ARN) Ihres IAM-Prinzipals zu ersetzen.
3. Wenden Sie die YAML-Dateien mit vollem oder eingeschränktem Zugriff auf Ihren Kubernetes-Cluster an. Ersetzen Sie den Wert der YAML-Datei durch Ihren eigenen.

```
kubectl apply -f eks-connector-console-dashboard-full-access-group.yaml
```

Informationen zum Anzeigen von Kubernetes-Ressourcen in Ihrem verbundenen Cluster finden Sie unter the section called “Zugriff auf Cluster-Ressourcen”. Daten für einige Ressourcentypen auf der Registerkarte Ressourcen sind nicht für verbundene Cluster verfügbar.

Abmelden eines Kubernetes-Clusters von der Amazon-EKS-Konsole

Wenn Sie mit der Verwendung eines verbundenen Clusters fertig sind, können Sie die Registrierung aufheben. Nach der Abmeldung ist der Cluster in der Amazon-EKS-Konsole nicht mehr sichtbar.

Sie müssen über die folgenden Berechtigungen verfügen, um die DeregisterCluster-API aufzurufen:

- `eks:DeregisterCluster`
- `ssm:DeleteActivation`
- `ssm:DeregisterManagedInstance`

Dieser Vorgang umfasst zwei Schritte: Abmeldung des Clusters von Amazon EKS und Deinstallation des Agents „eks-connector“ im Cluster.

Abmelden des Kubernetes-Clusters

Um einen Cluster aus Amazon EKS Connector abzumelden, können Sie eines der folgenden Tools verwenden:

- the section called “AWS-CLI”
- the section called “AWS-Managementkonsole”
- the section called “`eksctl`”

AWS-CLI

1. Die AWS-CLI muss installiert sein. Informationen zur Installation oder Aktualisierung finden Sie unter Installieren der AWS-CLI.
2. Stellen Sie sicher, dass die Agent-Rolle von Amazon EKS Connector erstellt wurde.
3. Den verbundenen Cluster deregistrieren.

```
aws eks deregister-cluster \
    --name my-cluster \
    --region region-code
```

AWS-Managementkonsole

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie Clusters (Cluster) aus.
3. Wählen Sie auf der Seite Cluster den verbundenen Cluster aus und wählen Sie Deregistrieren.
4. Bestätigen Sie, dass Sie den Cluster deregistrieren möchten.

`eksctl`

1. Installieren Sie die `eksctl`-Version `0.68` oder höher. Informationen zur Installation oder Aktualisierung finden Sie unter the section called “Cluster erstellen (eksctl)”.
2. Stellen Sie sicher, dass die Agent-Rolle von Amazon EKS Connector erstellt wurde.

3. Melden Sie den verbundenen Cluster ab:

```
eksctl deregister cluster --name my-cluster
```

Ressourcen in Ihrem Kubernetes-Cluster bereinigen

Verwenden Sie zur Deinstallation des `eks-connector`-Agenten eines der folgenden Tools:

- the section called “helm”
- the section called “yaml”

helm

Führen Sie zur Deinstallation des Agents den folgenden Befehl aus.

```
helm -n eks-connector uninstall eks-connector
```

yaml

1. Löschen Sie die Amazon-EKS-Connector-YAML-Datei aus Ihrem Kubernetes-Cluster.

```
kubectl delete -f eks-connector.yaml
```

2. Wenn Sie `clusterrole` oder `clusterrolebindings` für einen zusätzlichen IAM-Prinzipal für den Zugriff auf den Cluster erstellt haben, stellen Sie sicher, dass Sie ihn aus Ihrem Kubernetes-Cluster löschen.

Fehlerbehebung bei Problemen mit dem Amazon EKS Connector

Dieses Thema behandelt einige der häufigsten Fehler, die bei der Verwendung des Amazon EKS Connectors auftreten können, einschließlich Anleitungen zu deren Behebung und Umgehungen.

Grundlegende Fehlersuche

In diesem Abschnitt werden die Schritte zur Diagnose von Problemen mit dem Amazon EKS Connector beschrieben.

Prüfen Sie den Status von Amazon EKS Connector

Um den Status von Amazon EKS Connector zu überprüfen, geben Sie Folgendes ein:

```
kubectl get pods -n eks-connector
```

Überprüfen Sie die Protokolle des Amazon-EKS-Connectors

Der Amazon EKS Connector Pod besteht aus drei Containern. Führen Sie die folgenden Befehle aus, um vollständige Protokolle für alle diese Container abzurufen, damit Sie sie überprüfen können:

- `connector-init`

```
kubectl logs eks-connector-0 --container connector-init -n eks-connector
kubectl logs eks-connector-1 --container connector-init -n eks-connector
```

- `connector-proxy`

```
kubectl logs eks-connector-0 --container connector-proxy -n eks-connector
kubectl logs eks-connector-1 --container connector-proxy -n eks-connector
```

- `connector-agent`

```
kubectl exec eks-connector-0 --container connector-agent -n eks-connector -- cat /
var/log/amazon/ssm/amazon-ssm-agent.log
kubectl exec eks-connector-1 --container connector-agent -n eks-connector -- cat /
var/log/amazon/ssm/amazon-ssm-agent.log
```

Bringen Sie den effektiven Clusternamen in Erfahrung

Amazon-EKS-Cluster werden innerhalb eines einzigen AWS-Kontos und AWS-Region von `clusterName` identifiziert. Wenn Sie mehrere verbundene Cluster in Amazon EKS haben, können Sie bestätigen, bei welchem Amazon-EKS-Cluster der aktuelle Kubernetes-Cluster registriert ist. Geben Sie dafür Folgendes ein, um den `clusterName` des aktuellen Clusters herauszufinden.

```
kubectl exec eks-connector-0 --container connector-agent -n eks-connector \
  -- cat /var/log/amazon/ssm/amazon-ssm-agent.log | grep -m1 -oE "eks_c:[a-zA-Z0-9_-]+"
 | sed -E "s/^.*eks_c:([a-zA-Z0-9_-]+)_[a-zA-Z0-9]+.*$/\1/"
kubectl exec eks-connector-1 --container connector-agent -n eks-connector \
```

```
  -- cat /var/log/amazon/ssm/amazon-ssm-agent.log | grep -m1 -oE "eks_c:[a-zA-Z0-9_-]+"
| sed -E "s/^.*eks_c:([a-zA-Z0-9_-]+)_[a-zA-Z0-9]+.*$/\1/"
```

Verschiedene Befehle

Die folgenden Befehle sind nützlich, um Informationen abzurufen, die Sie zur Behebung von Problemen benötigen.

- Verwenden Sie den folgenden Befehl, um Images zu sammeln, die von Pods in Amazon EKS Connector verwendet werden.

```
kubectl get pods -n eks-connector -o jsonpath="{.items[*].spec.containers[*].image}"
 | tr -s '[[:space:]]' '\n'
```

- Verwenden Sie den folgenden Befehl, um die Knotennamen zu ermitteln, auf denen Amazon-EKS-Connector ausgeführt wird.

```
kubectl get pods -n eks-connector -o jsonpath="{.items[*].spec.nodeName}" | tr -s
 '[[:space:]]' '\n'
```

- Führen Sie den folgenden Befehl aus, um Ihre Kubernetes-Client- und Server-Versionen abzurufen.

```
kubectl version
```

- Führen Sie den folgenden Befehl aus, um Informationen zu Ihren Knoten abzurufen.

```
kubectl get nodes -o wide --show-labels
```

Helm-Ausgabe: 403 Forbidden

Wenn Sie beim Ausführen von Helm-Installationsbefehlen die folgende Fehlermeldung erhalten haben:

```
Error: INSTALLATION FAILED: unexpected status from HEAD request to https://
public.ecr.aws/v2/eks-connector/eks-connector-chart/manifests/0.0.6: 403 Forbidden
```

Sie können die folgende Zeile ausführen, um das Problem zu beheben:

```
docker logout public.ecr.aws
```

Konsolenfehler: Der Cluster steckt im Status „Ausstehend“ fest

Wenn der Cluster in der Amazon-EKS-Konsole im `Pending`-Status hängen bleibt, nachdem Sie ihn registriert haben, kann dies daran liegen, dass Amazon EKS Connector den Cluster noch nicht erfolgreich mit AWS verbunden hat. Für einen registrierten Cluster bedeutet der `Pending`-Status, dass die Verbindung nicht erfolgreich hergestellt wurde. Um dieses Problem zu lösen, stellen Sie sicher, dass Sie das Manifest auf den Kubernetes-Ziel-Cluster angewendet haben. Wenn Sie es auf den Cluster angewendet haben, der Cluster sich aber immer noch im `Pending`-Zustand befindet, ist `eks-connector`-StatefulSet möglicherweise fehlerhaft. Informationen zum Beheben dieses Problems finden Sie in diesem Thema unter the section called “Amazon-EKS-Connector-Pods stürzen fortlaufend ab”.

Konsolenfehler: Benutzer „system:serviceaccount:eks-connector:eks-connector“ kann keine Ressourcen-Benutzer in der API-Gruppe auf Cluster-Ebene imitieren.

Der Amazon EKS Connector nutzt die Kubernetes-Benutzer-Identitätsübernahme, um im Namen von IAM-Prinzipalen aus dem AWS-Managementkonsole zu agieren. Jedem IAM-Prinzipal, der auf die Kubernetes-API über das AWS `eks-connector`-Servicekonto zugreift, muss die Berechtigung gewährt werden, die Identität des entsprechenden Kubernetes-Benutzers mit einem IAM-ARN als Kubernetes-Benutzernamen anzunehmen. In den folgenden Beispielen wird der IAM-ARN einem Kubernetes-Benutzer zugeordnet.

- IAM-Benutzer *`John`* vom AWS-Konto *`111122223333`* wird einem Kubernetes-Benutzer zugeordnet. Bewährte Methoden für IAM empfehlen, dass Sie Rollen statt Benutzern Berechtigungen gewähren.

  ```
  arn:aws:iam::111122223333:user/john
  ```

- IAM-Rollen-*`Administrator`* von AWS-Konto *`111122223333`* ist einem Kubernetes-Benutzer zugeordnet:

  ```
  arn:aws:iam::111122223333:role/admin
  ```

Das Ergebnis ist ein ARN der IAM-Rolle anstelle des ARN der AWS-STS-Sitzung.

Anweisungen zum Konfigurieren der `ClusterRole` und `ClusterRoleBinding`, um dem `eks-connector`-Servicekonto die Berechtigung zu erteilen, sich als zugeordneter Benutzer auszugeben, finden Sie unter the section called “Zugriff auf Cluster gewähren”. Stellen Sie sicher, dass `%IAM_ARN%` in der Vorlage durch den IAM-ARN des AWS-Managementkonsole-IAM-Prinzipals ersetzt wird.

Konsolenfehler: […] ist verboten: Benutzer […] kann Ressource […] nicht in der API-Gruppe im Cluster-Bereich auflisten

Betrachten Sie das folgende Problem. Der Amazon EKS Connector hat sich erfolgreich als den anfordernden AWS-Managementkonsole-IAM-Prinzipal im Kubernetes-Ziel-Cluster ausgegeben. Der imitierte Prinzipal verfügt jedoch nicht über die RBAC-Berechtigung für Kubernetes-API-Operationen.

Um dieses Problem zu beheben, gibt es zwei Methoden, um zusätzlichen Benutzern Berechtigungen zu erteilen. Wenn Sie „eks-connector“ zuvor über Helm-Chart installiert haben, können Sie Benutzern ganz einfach Zugriff gewähren, indem Sie den folgenden Befehl ausführen. Ersetzen Sie `userARN1` und `userARN2` durch eine Liste der ARNs der IAM-Rollen, um Zugriff auf die -Ressourcen zu gewähren:

```
helm upgrade eks-connector oci://public.ecr.aws/eks-connector/eks-connector-chart \
    --reuse-values \
    --set 'authentication.allowedUserARNs={userARN1,userARN2}'
```

Alternativ können Sie als Cluster-Administrator einzelnen Kubernetes-Benutzern die entsprechenden RBAC-Berechtigungen zuweisen. Weitere Informationen und Beispiele finden Sie unter the section called “Zugriff auf Cluster gewähren”.

Konsolenfehler: Amazon EKS kann nicht mit dem API-Server Ihres Kubernetes-Clusters kommunizieren. Der Cluster muss sich im ACTIVE-Status befinden, um eine erfolgreiche Verbindung zu erreichen. Try again in few minutes. (Versuchen Sie es in ein paar Minuten erneut.)

Wenn der Amazon-EKS-Service nicht mit Amazon EKS Connector im Ziel-Cluster kommunizieren kann, kann dies einen der folgenden Gründe haben:

- Der Amazon-EKS-Connector im Ziel-Cluster ist fehlerhaft.

- Schlechte Konnektivität oder eine unterbrochene Verbindung zwischen dem Zielcluster und der AWS-Region.

Um dieses Problem zu beheben, überprüfen Sie die Amazon-EKS-Connector-Protokolle. Wenn kein Fehler für Amazon EKS Connector angezeigt wird, versuchen Sie die Verbindung nach einigen Minuten erneut. Sollten Sie regelmäßig hohe Latenzzeiten oder eine instabile Verbindung zum Ziel-Cluster feststellen, empfehlen wir Ihnen, den Cluster in einer AWS-Region neu zu registrieren, die sich näher an Ihrem Standort befindet.

Amazon-EKS-Connector-Pods stürzen fortlaufend ab

Es gibt viele Gründe, die dazu führen können, dass ein Amazon-EKS-Connector-Pod in den `CrashLoopBackOff`-Status übergeht. Dieses Problem betrifft wahrscheinlich den `connector-init`-Container. Überprüfen Sie den Status des Amazon-EKS-Connector-Pods.

```
kubectl get pods -n eks-connector
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME              READY   STATUS                  RESTARTS   AGE
eks-connector-0   0/2     Init:CrashLoopBackOff   1          7s
```

Wenn Ihre Ausgabe der vorherigen Ausgabe ähnlich ist, lesen Sie the section called “Überprüfen Sie die Protokolle des Amazon-EKS-Connectors”, um das Problem zu beheben.

eks-connector konnte nicht initiiert werden: InvalidActivation

Wenn Sie den Amazon-EKS-Connector zum ersten Mal starten, registriert er ein `activationId` und `activationCode` bei Amazon Web Services. Die Registrierung schlägt möglicherweise fehl, was dazu führen kann, dass der `connector-init`-Container mit einem Fehler ähnlich dem folgenden abstürzt.

```
F1116 20:30:47.261469       1 init.go:43] failed to initiate eks-connector:
 InvalidActivation:
```

Berücksichtigen Sie die folgenden Ursachen und empfohlenen Korrekturen, um dieses Problem zu beheben:

- Die Registrierung ist möglicherweise fehlgeschlagen, da `activationId` und `activationCode` nicht in Ihrer Manifest-Datei enthalten sind. Stellen Sie in diesem Fall sicher, dass es sich um die richtigen Werte handelt, die von der `RegisterCluster`-API-Operation zurückgegeben wurden, und dass sich `activationCode` in der Manifest-Datei befindet. Der `activationCode` wird zu Kubernetes-Geheimnissen hinzugefügt, daher muss er `base64`-codiert sein. Weitere Informationen finden Sie unter the section called “Schritt 1: Registrieren des Clusters”.
- Die Registrierung ist möglicherweise fehlgeschlagen, da Ihre Aktivierung abgelaufen ist. Dies liegt daran, dass Sie den Amazon-EKS-Connector aus Sicherheitsgründen innerhalb von drei Tagen nach der Registrierung des Clusters aktivieren müssen. Um dieses Problem zu beheben, stellen Sie sicher, dass das Manifest für Amazon EKS Connector vor Ablauf des Ablaufdatums und der Ablaufzeit auf den Ziel-Kubernetes-Cluster angewendet wird. Um das Ablaufdatum der Aktivierung zu bestätigen, rufen Sie die `DescribeCluster`-API-Operation auf.

  ```
  aws eks describe-cluster --name my-cluster
  ```

  In der folgenden Beispielantwort wird das Ablaufdatum und die Uhrzeit als `2021-11-12T22:28:51.101000-08:00` aufgezeichnet.

  ```
  {
      "cluster": {
          "name": "my-cluster",
          "arn": "arn:aws:eks:region:111122223333:cluster/my-cluster",
          "createdAt": "2021-11-09T22:28:51.449000-08:00",
          "status": "FAILED",
          "tags": {
          },
          "connectorConfig": {
              "activationId": "00000000-0000-0000-0000-000000000000",
              "activationExpiry": "2021-11-12T22:28:51.101000-08:00",
              "provider": "OTHER",
              "roleArn": "arn:aws:iam::111122223333:role/my-connector-role"
          }
      }
  }
  ```

  Wenn das `activationExpiry` erreicht wurde, melden Sie den Cluster ab und registrieren Sie ihn erneut. Dadurch wird eine neue Aktivierung generiert.

Im Cluster-Knoten fehlt die ausgehende Konnektivität

Um ordnungsgemäß zu funktionieren, benötigt der Amazon-EKS-Connector eine ausgehende Konnektivität zu mehreren AWS-Endpunkten. Sie können einen privaten Cluster ohne ausgehende Konnektivität nicht mit einer Ziel-AWS-Region verbinden. Um dieses Problem zu lösen, müssen Sie die erforderliche ausgehende Konnektivität hinzufügen. Weitere Informationen zu den Anforderungen für Konnektoren finden Sie unter the section called “Überlegungen zum Amazon EKS Connector”.

Amazon-EKS-Connectors-Pods befinden sich im **`ImagePullBackOff`**-Status

Wenn Sie den `get pods`-Befehl ausführen und sich Pods im `ImagePullBackOff`-Zustand befinden, können diese nicht ordnungsgemäß funktionieren. Wenn sich die Amazon-EKS-Connector-Pods im `ImagePullBackOff`-Status befinden, können sie nicht ordnungsgemäß funktionieren. Überprüfen Sie den Status Ihrer Amazon-EKS-Connector-Pods.

```
kubectl get pods -n eks-connector
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME              READY   STATUS                  RESTARTS   AGE
eks-connector-0   0/2     Init:ImagePullBackOff   0          4s
```

Die standardmäßige Amazon-EKS-Connector-Manifest-Datei verweist auf Images aus der öffentlichen Amazon-ECR-Galerie. Es ist möglich, dass der Ziel-Kubernetes-Cluster keine Images aus der Amazon ECR Public Gallery abrufen kann. Beheben Sie entweder das Image-Abrufproblem der öffentlichen Amazon-ECR-Galerie oder erwägen Sie eine Spiegelung der Images in der privaten Container-Registry Ihrer Wahl.

Häufig gestellte Fragen zum AWS-Konnektor

F: Wie funktioniert die zugrunde liegende Technologie hinter dem Amazon-EKS-Connector?

A: Amazon EKS Connector basiert auf dem AWS Systems Manager (Systems Manager)-Agenten. Amazon EKS Connector wird als `StatefulSet` auf Ihrem Kubernetes-Cluster ausgeführt. Er stellt eine Verbindung her und ermöglicht die Kommunikation zwischen dem API-Server Ihres Clusters und Amazon Web Services. Dadurch werden Clusterdaten in der Amazon EKS-Konsole angezeigt, bis

Sie den Cluster von AWS trennen. Der Systems-Manager-Agent ist ein Open-Source-Projekt. Weitere Informationen zu diesem Projekt finden Sie auf der GitHub-Projektseite.

F: Ich habe einen On-Premises-Kubernetes-Cluster, den ich verbinden möchte. Muss ich Firewall-Ports öffnen, um ihn zu verbinden?

A: Nein, Sie müssen keine Firewall-Ports öffnen. Der Kubernetes-Cluster erfordert nur eine ausgehende Verbindung zu AWS-Regionen. AWS-Services greifen niemals auf Ressourcen in Ihrem On-Premises-Netzwerk zu. Der Amazon EKS Connector läuft auf Ihrem Cluster und leitet die Verbindung mit AWS ein. Wenn die Clusterregistrierung abgeschlossen ist, gibt AWS Befehle nur an den Amazon EKS Connector aus, nachdem Sie eine Aktion von der Amazon EKS-Konsole aus gestartet haben, die Informationen vom Kubernetes-API-Server auf Ihrem Cluster benötigt.

F: Welche Daten werden vom Amazon EKS Connector über meinen Cluster an AWS gesendet?

A: Amazon EKS Connector sendet technische Informationen, die erforderlich sind, damit Ihr Cluster in AWS registriert wird. Er sendet auch Cluster- und Workload-Metadaten für die Funktionen der Amazon EKS-Konsole, die Kunden anfordern. Der Amazon-EKS-Connector erfasst oder sendet diese Daten nur, wenn Sie eine Aktion von der Amazon-EKS-Konsole oder der Amazon-EKS-API aus starten, die das Senden der Daten an AWS erfordert. Neben der Kubernetes-Versionsnummer speichert AWS standardmäßig keine Daten. Er speichert Daten nur, wenn Sie dies autorisieren.

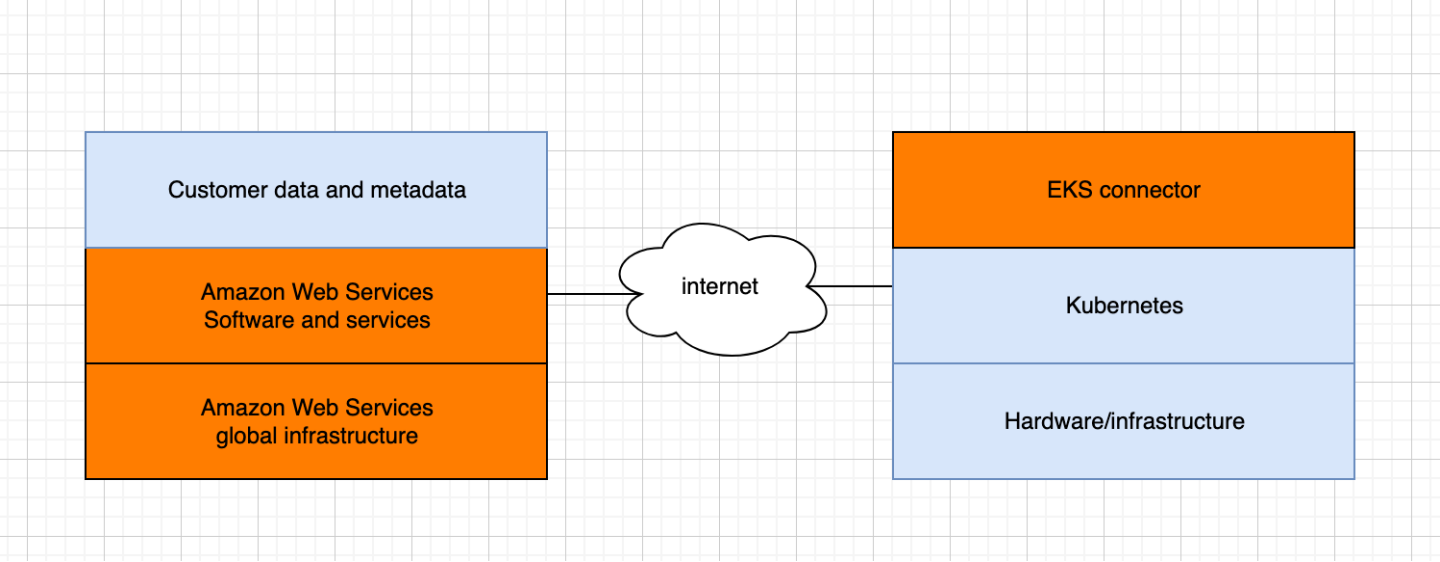
F: Kann ich einen Cluster außerhalb einer AWS-Region verbinden?

A: Ja, Sie können einen Cluster von jedem Standort aus mit Amazon EKS verbinden. Darüber hinaus kann sich Ihr Amazon-EKS-Service in jeder AWS-öffentlichen kommerziellen AWS-Region befinden. Dies funktioniert mit einer gültigen Netzwerkverbindung von Ihrem Cluster zur AWS-Zielregion. Wir empfehlen Ihnen, eine AWS-Region auszuwählen, die Ihrem Cluster-Standort am nächsten ist, um die Leistungsoptimierung der Benutzeroberfläche zu gewährleisten. Wenn Sie beispielsweise einen Cluster in Tokio ausführen, verbinden Sie Ihren Cluster mit der AWS-Region in Tokio (d. h. der `ap-northeast-1` AWS-Region), um eine geringe Latenz zu erzielen. Sie können einen Cluster von jedem Standort aus mit Amazon EKS in jeder öffentlichen kommerziellen AWS-Region verbinden, mit Ausnahme von China oder der GovCloud-AWS-Regionen.

Grundlegendes zur Sicherheit in Amazon EKS Connector

Der Amazon EKS Connector ist eine Open-Source-Komponente, die auf Ihrem Kubernetes-Cluster ausgeführt wird. Dieser Cluster kann sich außerhalb der AWS-Umgebung befinden. Dies erfordert

zusätzliche Überlegungen zur Sicherheitsverantwortung. Das folgende Diagramm verdeutlicht diese Konfiguration. Orange steht für Pflichten von AWS, Blau für Pflichten des Kunden:



In diesem Thema werden die Unterschiede im Verantwortungsmodell beschrieben, wenn sich der verbundene Cluster außerhalb von AWS befindet.

Pflichten von AWS

- Wartung, Erstellung und Bereitstellung von Amazon EKS Connector, eine Open-Source-Komponente, die auf dem Kubernetes-Cluster eines Kunden ausgeführt und mit AWS kommuniziert.
- Aufrechterhaltung der Kommunikationssicherheit auf Transport- und Anwendungsebene zwischen dem verbundenen Kubernetes-Cluster und den AWS-Services.

Pflichten des Kunden

- Kubernetes-Cluster-spezifische Sicherheit, insbesondere in folgenden Bereichen:
  - Kubernetes-Geheimnisse müssen ordnungsgemäß verschlüsselt und geschützt sein.
  - Sperren des Zugriffs auf den `eks-connector`-Namespace.
- Konfigurieren der RBAC-Berechtigungen (rollenbasierte Zugriffskontrolle) zur Verwaltung des Benutzerzugriffs von IAM-Prinzipal aus AWS. Detaillierte Anweisungen finden Sie unter the section called “Zugriff auf Cluster gewähren”.
- Amazon EKS Connector installieren und aktualisieren.

- Wartung der Hardware, Software und Infrastruktur, die den verbundenen Kubernetes-Cluster unterstützt.
- Sichern ihrer AWS-Konten (z. B. durch Verwendung von sicheren Anmeldeinformationen des Root-Benutzers).

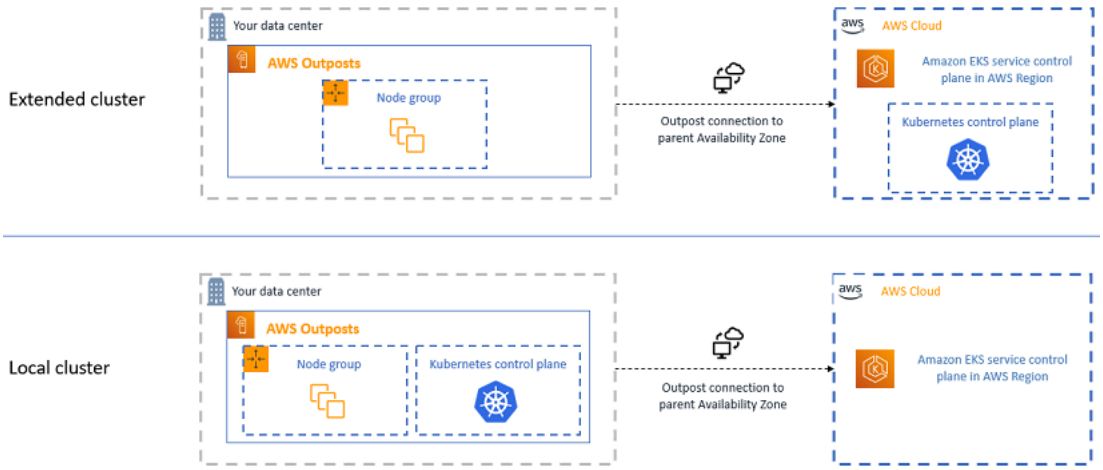
Bereitstellung von Amazon EKS On-Premises mit AWS Outposts

Sie können Amazon EKS verwenden, um On-Premises-Kubernetes-Anwendungen in AWS Outposts auszuführen. Sie können Amazon EKS auf Outposts folgendermaßen bereitstellen:

- Erweiterte Cluster – Führen Sie die Kubernetes-Steuerebene in einer AWS-Region und Knoten in Ihrem Outpost aus.
- Lokale Cluster – Führen Sie die Kubernetes-Steuerebene und Knoten in Ihrem Outpost aus.

Für beide Bereitstellungsoptionen wird die Kubernetes-Steuerebene vollständig von AWS verwaltet. Sie können dieselben Amazon-EKS-APIs, -Tools und -Konsolen verwenden, die Sie in der Cloud zum Erstellen und Ausführen von Amazon EKS auf Outposts verwenden.

Das folgende Diagramm zeigt diese Optionen für die Bereitstellung.



Wann die einzelnen Bereitstellungsoptionen verwendet werden sollten

Sowohl lokale als auch erweiterte Cluster sind allgemeine Bereitstellungsoptionen und können für eine Reihe von Anwendungen verwendet werden.

Mit lokalen Clustern können Sie den gesamten Amazon-EKS-Cluster lokal auf Outposts ausführen. Mit dieser Option können Sie das Risiko von Ausfallzeiten bei Anwendungen verringern, die durch vorübergehende Netzwerkunterbrechungen in der Cloud entstehen können. Diese

Netzwerkunterbrechungen können durch Glasfaserausfälle oder Wetterereignisse verursacht werden. Da der gesamte Amazon-EKS-Cluster lokal auf Outposts ausgeführt wird, bleiben Anwendungen verfügbar. Sie können Cluster-Vorgänge während Netzwerkunterbrechungen mit der Cloud durchführen. Weitere Informationen finden Sie unter the section called “Auf Unterbrechungen vorbereiten”. Wenn Sie Bedenken hinsichtlich der Qualität der Netzwerkverbindung zwischen Ihren Outposts und dem übergeordneten AWS-Region haben und eine hohe Verfügbarkeit durch Netzwerkunterbrechungen benötigen, verwenden Sie die Bereitstellungsoption des lokalen Clusters.

Mit erweiterten Clustern können Sie Kapazität in Ihrem Outpost sparen, da die Kubernetes-Steuerebene im übergeordneten AWS-Region ausgeführt wird. Diese Option ist geeignet, wenn Sie in eine zuverlässige, redundante Netzwerkverbindung von Ihrem Outpost zum AWS-Region investieren können. Die Qualität der Netzwerkverbindung ist für diese Option entscheidend. Die Art und Weise, wie Kubernetes Netzwerkunterbrechungen zwischen der -Steuerebene und den Knoten handhabt, kann zu Ausfallzeiten der Anwendung führen. Weitere Informationen zum Verhalten von Kubernetes finden Sie unter Planung, Vorkaufsrecht und Bereinigung in der Kubernetes-Dokumentation.

Vergleich der Optionen für die Bereitstellung

Die folgende Tabelle vergleicht die Unterschiede zwischen den beiden Optionen.

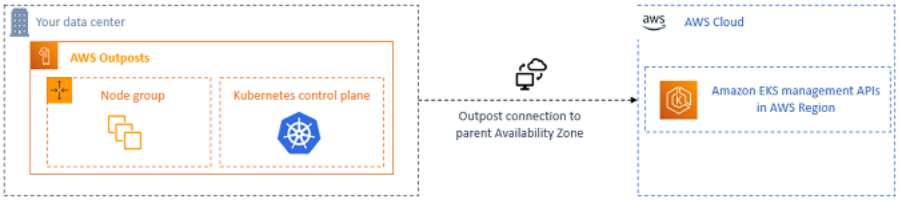
Funktion	Erweiterter Cluster	Lokaler Cluster
Ort der Kubernetes-Steuere bene	AWS-Region	Outpost
Konto der Kubernetes-Steuere bene	AWS-Konto	Ihr Konto
Regionale Verfügbarkeit	siehe Service-Endpunkte	USA Ost (Ohio), USA Ost (Nord-Virginia),USA West (Nordkalifornien), USA West (Oregon), Asien-Paz ifik (Seoul), Asien-Pazifik (Singapur), Asien-Pazifik (Sydney), Asien-Pazifik (Tokio), Kanada (Zentral), Europa (Frankfurt), Europa (Irland), Europa (London),

Funktion	Erweiterter Cluster	Lokaler Cluster
		Naher Osten (Bahrain), Naher Osten (Bahrain) und Südamerika (São Paulo)
Kubernetes-Nebenversionen	eks/latest/userguide/kubernetes-versions.html[Supported Amazon EKS versions, type="documentation"].	eks/latest/userguide/kubernetes-versions.html[Supported Amazon EKS versions, type="documentation"].
Plattformversionen	Siehe EKS-Plattformversionen	Siehe the section called “EKS-Plattformversionen”
Outpost-Formfaktoren	Outpost-Racks	Outpost-Racks
Benutzeroberflächen	AWS-Managementkonsole, AWS-CLI, Amazon-EKS-API, `eksctl`, AWS CloudFormation und Terraform	AWS-Managementkonsole, AWS-CLI, Amazon-EKS-API, `eksctl`, AWS CloudFormation und Terraform
Verwaltete Richtlinien	AmazonEKSClusterPolicy und the section called “AWS verwaltete Richtlinie: Amazon EKSService RolePolicy”	AmazonEKSLocalOutpostClusterPolicy und the section called “AWS verwaltete Richtlinie: Amazon EKSLocal OutpostServiceRolePolicy”
Cluster-VPC und Subnetze	Siehe the section called “VPC- und Subnetz-Anforderungen”	Siehe the section called “Erstellen von VPC und Subnetzen”
Cluster-Endpunktzugriff	Öffentlich oder privat oder beides	Nur privat
Kubernetes-API-Server-Authentifizierung	AWS Identity and Access Management (IAM) und OIDC	IAM und `x.509`-Zertifikate
Knotentypen	Nur selbstverwaltet	Nur selbstverwaltet

Funktion	Erweiterter Cluster	Lokaler Cluster
Knoten-Berechnungstypen	Amazon EC2 On-Demand	Amazon EC2 On-Demand
Knoten-Speichertypen	Amazon EBS gp2 und lokale NVMe-SSD	Amazon EBS gp2 und lokale NVMe-SSD
Amazon EKS-optimierte AMIs	Amazon Linux, Windows und Bottlerocket	Nur Amazon Linux
IP-Versionen	Nur `IPv4`	Nur `IPv4`
Add-Ons	Amazon-EKS-Add-Ons oder selbstverwaltete Add-Ons	Nur Selbstverwaltete Add-Ons
Standard-Container-Netzwerkschnittstelle	Amazon-VPC-CNI-Plug-In für Kubernetes	Amazon-VPC-CNI-Plug-In für Kubernetes
Kubernetes-Steuerebene-Protokolle	Amazon CloudWatch Logs	Amazon CloudWatch Logs
Load Balancing	Verwenden Sie den AWS Load Balancer Controller nur zum Bereitstellen von Application Load Balancern (keine Network Load Balancer)	Verwenden Sie den AWS Load Balancer Controller nur zum Bereitstellen von Application Load Balancern (keine Network Load Balancer)
Secrets-Umschlagverschlüsselung	Siehe the section called “Geheime Verschlüsselung aktivieren”	Nicht unterstützt
IAM-Rollen für Servicekonten	Siehe the section called “Anmeldeinformationen mit IRSA”	Nicht unterstützt
Fehlersuche	Siehe Fehlersuche	Siehe the section called “Fehlersuche bei Clustern”

Erstellung lokaler Amazon-EKS-Cluster in AWS Outposts für hohe Verfügbarkeit

Sie können lokale Cluster verwenden, um Ihren gesamten Amazon-EKS-Cluster lokal in AWS Outposts auszuführen. Auf diese Weise wird das Risiko von Anwendungsausfällen, die sich aus vorübergehenden Unterbrechungen der Netzwerkverbindung zur Cloud ergeben können, minimiert. Diese Verbindungsunterbrechungen können durch Glasfaserunterbrechungen oder Wetterereignisse verursacht werden. Weil das gesamte Kubernetes-Cluster lokal in Outposts ausgeführt wird, bleiben Anwendungen verfügbar. Sie können Cluster-Vorgänge während Netzwerkunterbrechungen mit der Cloud durchführen. Weitere Informationen finden Sie unter the section called “Auf Unterbrechungen vorbereiten”. Das folgende Diagramm zeigt eine lokale Cluster-Bereitstellung.



Lokale Cluster sind allgemein für die Verwendung mit Outpost-Racks verfügbar.

Unterstützte AWS-Regionen

Sie können lokale Cluster in den folgenden AWS-Regionen erstellen: USA Ost (Ohio), USA Ost (Nord-Virginia),USA West (Nordkalifornien), USA West (Oregon), Asien-Pazifik (Seoul), Asien-Pazifik (Singapur), Asien-Pazifik (Sydney), Asien-Pazifik (Tokio), Kanada (Zentral), Europa (Frankfurt), Europa (Irland), Europa (London), Naher Osten (Bahrain), Naher Osten (Bahrain) und Südamerika (São Paulo). Weitere Informationen zu unterstützten Funktionen finden Sie unter the section called “Vergleich der Optionen für die Bereitstellung”.

Stellen Sie einen Amazon EKS-Cluster auf AWS Outposts bereit

Dieses Thema bietet einen Überblick darüber, was beim Ausführen eines lokalen Clusters auf einem Outpost zu beachten ist. Das Thema enthält auch Anweisungen zum Bereitstellen eines lokalen Clusters auf einem Outpost.

> **Important**
>
> - Diese Überlegungen werden in der zugehörigen Amazon-EKS-Dokumentation nicht wiederholt. Wenn andere Themen der Amazon-EKS-Dokumentation mit den hier aufgeführten Überlegungen in Konflikt stehen, befolgen Sie die Überlegungen hier.
> - Diese Überlegungen sind freibleibend und können sich häufig ändern. Wir empfehlen Ihnen daher, dieses Thema regelmäßig zu überprüfen.
> - Viele der Überlegungen unterscheiden sich von den Überlegungen zur Erstellung eines Clusters in der AWS Cloud.

- Lokale Cluster unterstützen nur Outpost-Racks. Ein einzelner lokaler Cluster kann über mehrere physische Outpost-Racks laufen, die aus einem einzigen logischen Outpost bestehen. Ein einzelner lokaler Cluster kann nicht über mehrere logische Outposts hinweg ausgeführt werden. Jeder logische Outpost hat einen einzigen Outpost-ARN.
- Lokale Cluster führen und verwalten die Kubernetes-Steuerebenen in Ihrem Konto in dem Outpost. Sie können keine Workloads auf den Instances der Kubernete-Steuerebene ausführen oder die Komponenten der Kubernetes -Steuerebene ändern. Diese Knoten werden vom Amazon-EKS-Service verwaltet. Änderungen an derKubernetes-Steuerebene bleiben nicht durch automatische Amazon-EKS-Verwaltungsaktionen, wie etwa Patching, erhalten.
- Lokale Cluster unterstützen selbstverwaltete Add-Ons und selbstverwaltete Amazon-Linux-Knotengruppen. Das Amazon-VPC-CNI-Plugin für Kubernetes, kube-proxy und CoreDNS-Add-Ons wird automatisch auf lokalen Clustern installiert.
- Lokale Cluster erfordern die Verwendung von Amazon EBS auf Outposts. In Ihrem Outpost muss Amazon EBS für die Aufbewahrung der Kubernetes-Steuerebene verfügbar sein. Outposts unterstützen nur Amazon EBS `gp2`-Volumes.
- Amazon-EBS-gestützte Kubernetes `PersistentVolumes` werden mit dem Amazon-EBS-CSI-Treiber unterstützt.
- Die Instances der Steuerebene lokaler Cluster werden in einer gestapelten hochverfügbaren Topologie eingerichtet. Zwei der drei Steuerebenen-Instances müssen jederzeit funktionsfähig sein, um das Quorum aufrechtzuerhalten. Wenn das Quorum verloren geht, wenden Sie sich an den AWS Support, da einige serviceseitige Aktionen erforderlich sind, um die neuen verwalteten Instanzen zu aktivieren.

Voraussetzungen

- Vertrautheit mit den Outposts-Bereitstellungsoptionen, Auswahl von Instance-Typen und Platzierungsgruppen für Amazon EKS-Cluster auf AWS Outposts auf der Grundlage von Kapazitätsüberlegungen sowie VPC-Anforderungen und -Überlegungen
- Ein vorhandener Outpost. Weitere Informationen finden Sie unter Was sind AWS Outposts.
- Das `kubectl`-Befehlszeilen-Tool ist auf Ihrem Computer oder AWS CloudShell installiert. Die Version kann mit der Kubernetes-Version Ihres Clusters identisch sein oder bis zu einer Nebenversion älter oder neuer sein. Wenn Ihre Clusterversion beispielsweise `1.29` ist, können Sie `kubectl`-Version `1.28`, `1.29`, oder `1.30` damit verwenden. Informationen zum Installieren oder Aktualisieren von `kubectl` finden Sie unter the section called “kubectl und eksctl einrichten”.
- Version `2.12.3` oder höher oder Version `1.27.160` oder höher der auf Ihrem Gerät installierten und konfigurierten AWS Befehlszeilenschnittstelle (AWS CLI) oder AWS CloudShell. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Paketmanager wie `yum`apt-get, oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der AWS CLI. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Die AWS CLI-Version, in der installiert ist, AWS CloudShell kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zur Aktualisierung finden Sie im AWS CloudShell Benutzerhandbuch unter AWS CLI in Ihrem Home-Verzeichnis installieren.
- Ein IAM-Prinzipal (Benutzer oder Rolle) mit Berechtigungen zum Erstellen (`create`) und Beschreiben (`describe`) eines Amazon-EKS-Clusters. Weitere Informationen erhalten Sie unter the section called “Kubernetes-Cluster auf einem Outpost erstellen” und the section called “Auflisten oder Beschreiben aller Cluster”.

Wenn ein lokaler Amazon-EKS-Cluster erstellt wird, wird der IAM-Prinzipal, der den Cluster erstellt, dauerhaft hinzugefügt. Der Prinzipal wird ausdrücklich als Administrator in die Kubernetes-RBAC-Autorisierungstabelle aufgenommen. Diese Entität hat `system:masters`-Berechtigungen. Die Identität dieser Entität ist in Ihrer Cluster-Konfiguration nicht sichtbar. Daher ist es wichtig, die Entität zu notieren, die den Cluster erstellt hat, und sicherzustellen, dass Sie diese keinesfalls löschen. Anfänglich kann nur der Prinzipal, der den Server erstellt hat, Aufrufe an den Kubernetes-API-Server mit `kubectl` tätigen. Wenn Sie die Konsole zum Erstellen des Clusters verwenden, stellen Sie sicher, dass sich dieselben IAM-Anmeldeinformationen in der AWS SDK-Anmeldeinformationskette befinden, wenn Sie `kubectl` Befehle auf Ihrem Cluster ausführen. Nachdem Ihr Cluster erstellt wurde, können Sie anderen IAM-Prinzipalen Zugriff auf Ihren Cluster gewähren.

Einen lokalen Amazon-EKS-Cluster erstellen

Sie können einen lokalen Cluster mit den folgenden Tools erstellen, die auf dieser Seite beschrieben werden:

- the section called “eksctl”
- the section called “AWS-Managementkonsole”

Sie können auch die AWS CLI, die Amazon EKS-API, die AWS CloudFormationoder Terraform verwenden AWS SDKs, um Cluster auf Outposts zu erstellen.

`eksctl`

So erstellen Sie einen Cluster mit **`eksctl`**

1. Installieren Sie Version `0.215.0` oder höher des `eksctl` Befehlszeilentools auf Ihrem Gerät oder. AWS CloudShell Informationen zum Installieren und Aktualisieren von `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.
2. Kopieren Sie den folgenden Inhalt auf Ihr Gerät. Ersetzen Sie die folgenden Werte und führen Sie dann den geänderten Befehl aus, um die `outpost-control-plane.yaml`-Datei zu erstellen:
   - *`region-code`*Ersetzen Sie es durch die unterstützte AWS Region, in der Sie Ihren Cluster erstellen möchten.
   - Ersetzen Sie *`my-cluster`* durch Ihren Cluster-Namen. Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein.
   - Ersetzen Sie *`vpc-ExampleID1`* und *`subnet-ExampleID1`* durch die IDs Ihrer vorhandenen VPC und Ihres Subnetzes. Die VPC und das Subnetz müssen die Anforderungen unter Erstellen einer VPC und Subnetze für Amazon EKS-Cluster auf Outposts erfüllen. AWS
   - Ersetzen Sie es durch die *`uniqueid`* ID Ihres Outposts.
   - Ersetzen Sie *`m5.large`* durch einen Instance-Typ, der auf Ihrem Outpost verfügbar ist. Bevor Sie einen Instance-Typ auswählen, lesen Sie the section called “Überlegungen zur Kapazität”. Drei Steuerebenen-Instances werden bereitgestellt. Sie können diese Nummer nicht ändern.

```
cat >outpost-control-plane.yaml <<EOF
```

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code
  version: "1.33"

vpc:
  clusterEndpoints:
    privateAccess: true
  id: "vpc-vpc-ExampleID1"
  subnets:
    private:
      outpost-subnet-1:
        id: "subnet-subnet-ExampleID1"

outpost:
  controlPlaneOutpostARN: arn:aws: outposts:region-code:111122223333:outpost/op-
uniqueid
  controlPlaneInstanceType: m5.large
EOF
```

Eine vollständige Liste aller verfügbaren Optionen und Standardwerte finden Sie unter AWS - Outposts-Support und Konfigurationsdateischema in der `eksctl`-Dokumentation.

3. Erstellen Sie den Cluster mit der Konfigurationsdatei, die Sie im vorherigen Schritt erstellt haben. `eksctl` erstellt eine VPC und ein Subnetz auf Ihrem Outpost, in dem Sie den Cluster bereitstellen können.

```
eksctl create cluster -f outpost-control-plane.yaml
```

Die Clusterbereitstellung dauert mehrere Minuten. Während der Cluster erstellt wird, werden mehrere Ausgabezeilen angezeigt. Die letzte Ausgabezeile ähnelt der folgenden Beispielzeile.

```
[#]  EKS cluster "my-cluster" in "region-code" region is ready
```

> **Tip**
>
> Sie können die meisten Optionen, die beim Erstellen eines Clusters mit `eksctl` angegeben werden können, über den Befehl `eksctl create cluster --help`

anzeigen. Verwenden Sie eine `config`-Datei, um alle verfügbaren Optionen anzuzeigen. Weitere Informationen finden Sie unter Verwenden von Config-Dateien und im Config-Datei-Schema in der `eksctl`-Dokumentation. Auf GitHub finden Sie Beispiele für Config-Dateien.

Der `eksctl`-Befehl hat automatisch einen Zugriffseintrag für den IAM-Prinzipal (Benutzer oder Rolle) erstellt, der den Cluster erstellt hat, und dem IAM-Prinzipal Administratorberechtigungen für Kubernete -Objekte im Cluster gewährt. Wenn der Cluster-Ersteller keinen Administratorzugriff auf Kubernetes-Objekte im Cluster haben soll, fügen Sie der vorherigen Konfigurationsdatei den folgenden Text hinzu: `bootstrapClusterCreatorAdminPermissions: false` (auf der gleichen Ebene wie `metadata`, `vpc` und `outpost`). Wenn Sie die Option hinzugefügt haben, müssen Sie nach der Cluster-Erstellung einen Zugriffseintrag für mindestens einen IAM-Prinzipal erstellen. Andernfalls kann kein IAM-Prinzipal auf Kubernetes-Objekte im Cluster zugreifen.

AWS-Managementkonsole

Um Ihren Cluster mit dem zu erstellen AWS-Managementkonsole

1. Sie benötigen eine vorhandene VPC und ein Subnetz, die den Anforderungen von Amazon EKS entsprechen. Weitere Informationen finden Sie unter the section called “Erstellen von VPC und Subnetzen”.
2. Wenn Sie bereits eine lokale Cluster-IAM-Rolle haben oder Ihren Cluster mit `eksctl` erstellen, können Sie diesen Schritt überspringen. Standardmäßig erstellt `eksctl` eine Rolle für Sie.
   a. Führen Sie den folgenden Befehl aus, um eine JSON-Datei für eine IAM-Vertrauensrichtlinie zu erstellen.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
```

```
}
```

b. Erstellen Sie die Amazon-EKS-Cluster-IAM-Rolle. Um eine IAM-Rolle zu erstellen, muss dem IAM-Prinzipal, der die Rolle erstellt, die folgende `iam:CreateRole`-Aktion (Berechtigung) zugewiesen werden.

```
aws iam create-role --role-name myAmazonEKSLocalClusterRole --assume-role-policy-
document file://"eks-local-cluster-role-trust-policy.json"
```

c. Hängen Sie die von Amazon EKS verwaltete Richtlinie mit dem Namen Amazon EKSLocal OutpostClusterPolicy an die Rolle an. Um eine IAM-Richtlinie an einen IAM-Prinzipal anzuhängen, muss der Prinzipal, der die Richtlinie anhängt, eine der folgenden IAM-Aktionen (Berechtigungen) zugewiesen werden: `iam:AttachUserPolicy` oder `iam:AttachRolePolicy`.

```
aws iam attach-role-policy --policy-arn arn:aws: iam::aws:policy/
AmazonEKSLocalOutpostClusterPolicy --role-name myAmazonEKSLocalClusterRole
```

3. Öffnen Sie die Amazon-EKS-Konsole.
4. Vergewissern Sie sich, dass Sie oben auf dem Konsolenbildschirm eine unterstützte AWS Region ausgewählt haben.
5. Wählen Sie Cluster hinzufügen und dann Erstellen aus.
6. Füllen Sie auf der Seite Configure cluster (Cluster konfigurieren) die folgenden Felder aus oder wählen Sie sie aus:
   - Standort der Kubernetes-Steuerebene — Wählen Sie AWS Outposts.
   - Outpost-ID – Wählen Sie die ID des Outposts, auf dem Sie Ihre Steuerebene erstellen möchten.
   - Instance Type (Instance-Typ) – Wählen Sie einen Instance-Typ aus. Es werden nur die in Ihrem Outpost verfügbaren Instance-Typen angezeigt. In der Dropdown-Liste beschreibt jeder Instance-Typ, für wie viele Knoten der Instance-Typ empfohlen wird. Bevor Sie einen Instance-Typ auswählen, lesen Sie the section called “Überlegungen zur Kapazität”. Alle Replikate werden mit demselben Instance-Typ bereitgestellt. Sie können den Instance-Typ nicht mehr ändern, nachdem der Cluster erstellt wurde. Drei Steuerebenen-Instances werden bereitgestellt. Sie können diese Nummer nicht ändern.
   - Name – Ein Name für Ihren Cluster. Es muss in Ihrem Konto einzigartig sein. AWS Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des

AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein.

- Kubernetes-Version – Wählen Sie die Kubernetes-Version aus, die Sie für Ihren Cluster verwenden möchten. Wir empfehlen, die frühere Version auszuwählen, es sei denn, Sie müssen eine ältere Version verwenden.
- Cluster-Servicerolle — Wählen Sie die Amazon EKS-Cluster-IAM-Rolle aus, die Sie in einem vorherigen Schritt erstellt haben, damit die Kubernetes-Steuerebene Ressourcen verwalten kann. AWS
- Kubernetes-Cluster-Administratorzugriff: Wenn der IAM-Prinzipal (Rolle oder Benutzer), der den Cluster erstellt, Administratorzugriff auf die Kubernetes-Objekte im Cluster haben soll, übernehmen Sie die Standardeinstellung (zulassen). Amazon EKS erstellt einen Zugriffseintrag für den IAM-Prinzipal und erteilt Cluster-Administratorberechtigungen für den Zugriffseintrag. Weitere Informationen zu Zugriffseinträgen finden Sie unter the section called “Zugriffseinträge”.

  Wenn nicht der Prinzipal, der den Cluster erstellt, sondern ein anderer IAM-Prinzipal Administratorzugriff auf Kubernetes-Cluster-Objekte haben soll, wählen Sie die Verweigerungsoption aus. Nach der Cluster-Erstellung kann jeder IAM-Prinzipal, der über IAM-Berechtigungen zum Erstellen von Zugriffseinträgen verfügt, Zugriffseinträge für beliebige IAM-Prinzipale hinzufügen, die Zugriff auf Kubernetes-Cluster-Objekte benötigen. Weitere Informationen zu den erforderlichen IAM-Berechtigungen finden Sie in der Service-Authorization-Referenz unter Von Amazon Elastic Kubernetes Service definierte Aktionen. Wenn Sie die Verweigerungsoption auswählen und keine Zugriffseinträge erstellen, können keine IAM-Prinzipale auf die Kubernetes-Objekte im Cluster zugreifen.
- Tags – (Optional) Fügen Sie Ihrem Cluster beliebige Tags hinzu. Weitere Informationen finden Sie unter the section called “Markieren Ihrer -Ressourcen”. Wenn Sie mit dieser Seite fertig sind, wählen Sie Weiter aus.

7. Wählen Sie auf der Seite Specify networking (Netzwerk angeben) die Werte für die folgenden Felder aus:
   - VPC – Wählen Sie eine vorhandene VPC aus. Die VPC muss über eine ausreichende Anzahl von IP-Adressen für den Cluster, alle Knoten und andere Kubernetes-Ressourcen, die Sie erstellen möchten, verfügen. Ihr VPC muss die Anforderungen in den VPC-Anforderungen und -Überlegungen erfüllen.
   - Subnetze – Standardmäßig sind alle im vorherigen Feld angegebenen Subnetze in der VPC vorausgewählt. Die von Ihnen ausgewählten Subnetze müssen die in den Subnetz-Anforderungen und -Überlegungen beschriebenen Anforderungen erfüllen.

- Security groups (Sicherheitsgruppen) – (Optional) Geben Sie eine oder mehrere Sicherheitsgruppen an, die Amazon EKS den erstellten Netzwerkschnittstellen zuordnen soll. Amazon EKS erstellt automatisch eine Sicherheitsgruppe, die die Kommunikation zwischen Ihrem Cluster und Ihrer VPC ermöglicht. Amazon EKS verknüpft diese Sicherheitsgruppe und alle, die Sie wählen, mit den erstellten Netzwerkschnittstellen. Weitere Informationen zu der Cluster-Sicherheitsgruppe, die Amazon EKS erstellt, finden Sie unter the section called “Anforderungen an Sicherheitsgruppen”. Sie können die Regeln in der von Amazon EKS erstellten Cluster-Sicherheitsgruppe ändern. Wenn Sie Ihre eigenen Sicherheitsgruppen hinzufügen möchten, können Sie die ausgewählten nach der Cluster-Erstellung nicht ändern. Damit On-Premises-Hosts mit dem Cluster-Endpunkt kommunizieren können, müssen Sie eingehenden Datenverkehr von der Cluster-Sicherheitsgruppe zulassen. Bei Clustern, die nicht über eine eingehende und ausgehende Internetverbindung verfügen (auch als private Cluster bezeichnet), müssen Sie eine der folgenden Maßnahmen ergreifen:
  - Fügen Sie die Sicherheitsgruppe hinzu, die mit den erforderlichen VPC-Endpunkten verbunden ist. Weitere Informationen zu den erforderlichen Endpunkten finden Sie unter Subnetzzugriff the section called “Verwendung von -Schnittstellen-VPC-Endpunkten” auf Dienste. AWS
  - Ändern Sie die Sicherheitsgruppe, die Amazon EKS erstellt hat, um Datenverkehr von der Sicherheitsgruppe zuzulassen, die den VPC-Endpunkten zugeordnet ist. Wenn Sie mit dieser Seite fertig sind, wählen Sie Weiter aus.

8. Auf der Seite Beobachtbarkeit konfigurieren können Sie optional auswählen, welche Metriken und Optionen zur Steuerebenen-Protokollierung Sie aktivieren möchten. Standardmäßig sind alle Protokollierungstypen deaktiviert.
   - Weiteren Informationen zur Prometheus-Metrik-Option finden Sie unter the section called “Schritt 1: Prometheus-Metriken aktivieren”.
   - Weitere Informationen zu den Optionen für die Steuerebenen-Protokollierung finden Sie unter the section called “Protokolle der Steuerebene”. Wenn Sie mit dieser Seite fertig sind, wählen Sie Weiter aus.
9. Überprüfen Sie auf der Seite Review and create (Überprüfen und erstellen) die Informationen, die Sie auf den vorherigen Seiten eingegeben oder ausgewählt haben. Wenn Sie Änderungen vornehmen müssen, wählen Sie Edit (Bearbeiten). Wenn Sie zufrieden sind, klicken Sie auf Erstellen. Das Feld Status zeigt CREATING (WIRD ERSTELLT) an, während der Cluster bereitgestellt wird.

   Die Clusterbereitstellung dauert mehrere Minuten.

Anzeigen Ihres lokalen Amazon-EKS-Clusters

1. Nachdem Ihr Cluster erstellt wurde, können Sie die erstellten Amazon EC2 Control Plane-Instances anzeigen.

   ```
   aws ec2 describe-instances --query 'Reservations[*].Instances[*].{Name:Tags[?
   Key==`Name`]|[0].Value}' | grep my-cluster-control-plane
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   "Name": "my-cluster-control-plane-id1"
   "Name": "my-cluster-control-plane-id2"
   "Name": "my-cluster-control-plane-id3"
   ```

   Jede Instance ist mit `node-role.eks-local.amazonaws.com/control-plane` gekennzeichnet, sodass auf den Instances der Steuerebene keine Workloads geplant werden. Weitere Informationen zu Taints finden Sie unter Taints und Toleranzen in der Kubernetes-Dokumentation. Amazon EKS überwacht kontinuierlich den Status lokaler Cluster. Wir führen automatische Verwaltungsaktionen durch, z. B. Sicherheits-Patches und das Reparieren fehlerhafter Instances. Wenn lokale Cluster von der Cloud getrennt werden, führen wir Aktionen durch, um sicherzustellen, dass der Cluster bei der erneuten Verbindung wieder in einen fehlerfreien Zustand versetzt wird.
2. Wenn Sie Ihren Cluster mit `eksctl` erstellt haben, können Sie diesen Schritt überspringen. `eksctl` schließt diesen Schritt für Sie ab. Aktivieren Sie `kubectl`, um mit Ihrem Cluster zu kommunizieren, indem Sie einen neuen Kontext zur Datei `kubectl config` hinzufügen. Anweisungen zum Erstellen und Aktualisieren der Datei finden Sie unter the section called “Zugriff auf den Cluster mit kubectl”.

   ```
   aws eks update-kubeconfig --region region-code --name my-cluster
   ```

   Eine Beispielausgabe sieht wie folgt aus.

   ```
   Added new context arn:aws: eks:region-code:111122223333:cluster/my-cluster to /home/
   username/.kube/config
   ```
3. Um eine Verbindung zum Kubernetes-API-Server Ihres lokalen Clusters herzustellen, müssen Sie Zugriff auf das lokale Gateway für das Subnetz haben oder eine Verbindung innerhalb der VPC herstellen. Weitere Informationen zum Verbinden eines Outpost-Racks mit Ihrem lokalen Netzwerk

finden Sie unter So funktionieren lokale Gateways für Racks im AWS Outposts-Benutzerhandbuch. Wenn Sie direktes VPC-Routing verwenden und das Outpost-Subnetz eine Route zu Ihrem lokalen Gateway hat, werden die privaten IP-Adressen der -Instances der Kubernetes-Steuerebene automatisch über Ihr lokales Netzwerk übertragen. Der Kubernetes-API-Server-Endpunkt des lokalen Clusters wird in Amazon Route 53 (Route 53) gehostet. Der API-Service-Endpunkt kann von öffentlichen DNS-Servern in die privaten IP-Adressen der Kubernetes API-Server aufgelöst werden.

Die Instances der Kubernetes-Steuerebenen lokaler Cluster sind mit statischen Elastic-Network-Schnittstellen mit festen privaten IP-Adressen konfiguriert, die sich während des Cluster-Lebenszyklus nicht ändern. Rechner, die mit dem Kubernetes-API-Server interagieren, verfügen möglicherweise nicht über eine Verbindung zu Route 53, wenn die Netzwerkverbindung unterbrochen ist. In diesem Fall empfehlen wir, `/etc/hosts` mit den statischen privaten IP-Adressen für den weiteren Betrieb zu konfigurieren. Wir empfehlen außerdem, lokale DNS-Server einzurichten und diese mit Ihrem Outpost zu verbinden. Weitere Informationen finden Sie in der AWS -Outposts-Dokumentation. Führen Sie den folgenden Befehl aus, um zu bestätigen, dass die Kommunikation mit Ihrem Cluster hergestellt wurde.

```
kubectl get svc
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   28h
```

4. (Optional) Testen Sie die Authentifizierung für Ihren lokalen Cluster, wenn dieser nicht mit der Cloud verbunden ist. AWS Detaillierte Anweisungen finden Sie unter the section called “Auf Unterbrechungen vorbereiten”.

Interne Ressourcen

Amazon EKS erstellt die folgenden Ressourcen in Ihrem Cluster. Die Ressourcen sind für den internen Gebrauch von Amazon EKS bestimmt. Damit Ihr Cluster ordnungsgemäß funktioniert, sollten Sie diese Ressourcen nicht bearbeiten oder ändern.

- Die folgenden Spiegel-Pods:
  - `aws-iam-authenticator-`*`node-hostname`*
  - `eks-certificates-controller-`*`node-hostname`*

  - `etcd-`*`node-hostname`*
  - `kube-apiserver-`*`node-hostname`*
  - `kube-controller-manager-`*`node-hostname`*
  - `kube-scheduler-`*`node-hostname`*
- Die folgenden selbstverwalteten Add-Ons:
  - `kube-system/coredns`
  - `kube-system/ kube-proxy` (wird erst erstellt, wenn Sie Ihren ersten Knoten hinzugefügt haben)
  - `kube-system/aws-node` (wird erst erstellt, wenn Sie Ihren ersten Knoten hinzugefügt haben). Lokale Cluster verwenden das Amazon VPC-CNI-Plugin für das Kubernetes-Plugin für die Cluster-Vernetzung. Ändern Sie nicht die Konfiguration für Steuerebene-Instances (Pods mit dem Namen `aws-node-controlplane-*`). Es gibt Konfigurationsvariablen, die Sie verwenden können, um den Standardwert zu ändern, wenn das Plugin neue Netzwerkschnittstellen erstellt. Weitere Informationen finden Sie in der Dokumentation zu GitHub.
- Die folgenden Services:
  - `default/kubernetes`
  - `kube-system/kube-dns`
- Eine `PodSecurityPolicy` namens `eks.system`
- Eine `ClusterRole` namens `eks:system:podsecuritypolicy`
- Eine `ClusterRoleBinding` namens `eks:system`
- Zusätzlich zur Cluster-Sicherheitsgruppe erstellt Amazon EKS in Ihrem AWS Konto eine Sicherheitsgruppe mit dem Name`eks-local-internal-do-not-use-or-edit-`*`cluster-name`*`-`*`uniqueid`* . Diese Sicherheitsgruppe ermöglicht den freien Datenverkehr zwischen Kubernetes-Komponenten, die in Instances der Steuerebene ausgeführt werden.

Empfohlene nächste Schritte:

- Erteilen Sie dem IAM-Prinzipal, der den Cluster erstellt hat, die erforderlichen Berechtigungen zum Anzeigen von Kubernetes-Ressourcen in AWS-Managementkonsole
- Gewähren Sie IAM-Entitäten Zugriff auf Ihren Cluster. Wenn Sie möchten, dass die Entitäten Kubernetes-Ressourcen in der Amazon-EKS-Konsole anzeigen können, gewähren Sie ihnen die Erforderlichen Berechtigungen.
- Konfigurieren Sie die Protokollierung für Ihren Cluster

- Machen Sie sich mit dem vertraut, was während Netzwerktrennungen passiert.
- Fügen Sie Knoten zu Ihrem Cluster hinzu
- Erwägen Sie, einen Backup-Plan für Ihr `etcd` zu erstellen. Amazon EKS unterstützt keine automatisierte Sicherung und Wiederherstellung von `etcd` für lokale Cluster. Weitere Informationen finden Sie unter Sicherung eines etcd-Clusters in der Kubernetes-Dokumentation. Die beiden Hauptoptionen sind die Verwendung von `etcdctl` zur Automatisierung der Erstellung von Snapshots oder die Verwendung der Amazon-EBS-Speicher-Volume-Sicherung.

Erfahren Sie mehr über die Plattformversionen von Kubernetes und Amazon EKS für Outposts AWS

Lokale Clusterplattformversionen stellen die Funktionen des Amazon EKS-Clusters auf AWS Outposts dar. Die Versionen umfassen die Komponenten, die in der Kubernetes-Steuerebene ausgeführt werden, wobei die Kubernetes-API-Server-Flags aktiviert sind. Sie enthalten auch die aktuelle Kubernetes-Patch-Version. Jede Kubernetes-Minor-Version hat eine oder mehrere zugehörige -Plattformversionen. Die Plattformversionen für verschiedene Kubernetes-Nebenversionen sind unabhängig. Die Plattformversionen für lokale Cluster und Amazon-EKS-Cluster in der Cloud sind unabhängig.

Wenn eine neue Kubernetes-Nebenversion für lokale Cluster verfügbar ist, z. B. `1.31`, beginnt die anfängliche Plattformversion für diese Kubernetes-Nebenversion bei `eks-local-outposts.1`. Allerdings veröffentlicht Amazon EKS regelmäßig neue Plattformversionen, um neue Einstellungen für die Kubernetes-Steuerebene und Sicherheitsfixes bereitzustellen.

Wenn neue lokale Cluster-Plattformversionen für eine Minor-Version verfügbar werden:

- Die Versionsnummer der -Plattform wird erhöht (`eks-local-outposts.n+1`).
- Amazon EKS aktualisiert automatisch alle vorhandenen lokalen Cluster auf die neueste Plattformversion für ihre entsprechende Kubernetes-Nebenversion. Automatische Aktualisierungen bestehender Plattformversionen werden schrittweise durchgeführt. Der Einführungsprozess umfasst den schrittweisen Austausch der verwalteten Kubernetes-Steuerebenen-Instances, die auf dem Outpost ausgeführt werden, bis alle 3 Instances durch neue ersetzt sind.
- Der Austauschprozess der Kubernetes-Steuerebenen-Instance wird gestoppt, wenn die Gefahr einer Service-Unterbrechung besteht. Amazon EKS versucht nur dann, eine Instance zu ersetzen, wenn die beiden anderen Kubernetes-Steuerebenen-Instances fehlerfrei sind und alle Bereitschaftsbedingungen als Cluster-Knoten erfüllen.

- Die Einführung einer neuen Plattformversion dauert in der Regel weniger als 30 Minuten. Wenn ein Cluster über einen längeren Zeitraum im `UPDATING` Status verbleibt, wenden Sie sich an den Support the section called “Fehlersuche bei Clustern” und wenden Sie sich an den AWS Support. Beenden Sie Kubernetes-Steuerungsinstanzen niemals manuell, es sei denn, Sie werden vom Support dazu aufgefordert. AWS
- Amazon EKS veröffentlicht möglicherweise ein neues Knoten-AMI mit einer entsprechenden Patch-Version. Alle Patch-Versionen sind zwischen der Kubernetes-Steuerebene und dem Knoten AMIs für eine einzelne Kubernetes-Nebenversion kompatibel.

Neue Plattformversionen führen keine kritischen Änderungen ein. Sie führen nicht zu Service-Unterbrechungen.

Lokale Cluster werden immer mit der neuesten verfügbaren Plattformversion (`eks-local-outposts.n`) für die angegebene Kubernetes-Version erstellt.

Die aktuellen und kürzlichen -Plattformversionen sind in den folgenden Tabellen beschrieben.

Um Benachrichtigungen über alle Änderungen an den Quelldateien dieser spezifischen Dokumentationsseite zu erhalten, können Sie die folgende URL mit einem RSS-Reader abonnieren:

```
https://github.com/awsdocs/amazon-eks-user-guide/commits/mainline/latest/ug/outposts/eks-outposts-platform-versions.adoc.atom
```

Kubernetes-Version `1.31`

Die folgenden Zugangs-Controller sind für alle `1.31`-Plattformversionen aktiviert: `CertificateApproval`, `CertificateSigning`, `CertificateSubjectRestriction`, `ClusterTrustBundleAttest`, `DefaultIngressClass`, `DefaultStorageClass`, `DefaultTolerationSeconds`, `ExtendedResourceToleration`, `LimitRanger`, `MutatingAdmissionWebhook`, `NamespaceLifecycle`, `NodeRestriction`, `PersistentVolumeClaimResize`, `PodSecurity`, `Priority`, `ResourceQuota`, `RuntimeClass`, `ServiceAccount`, `StorageObjectInUseProtection`, `TaintNodesByCondition`, `ValidatingAdmissionPolicy` und `ValidatingAdmissionWebhook`.

Kubernetes-Version	Amazon-EKS-Plattformversion	Versionshinweise	Datum der Veröffentlichung
1.31.14	eks-local-outposts.8	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. Kube-Proxy wurde aktualisiert auf. v1.31.14 AWS IAM Authenticator wurde aktualisiert auf. v0.7.8 Amazon-VPC-CNI-Plugin für Kubernetes aktualisiert zu v1.20.4. Bottlerocket-Version aktualisiert auf v1.52.0.	23. Dezember 2025
1.31.12	eks-local-outposts.5	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. Kube-Proxy wurde aktualisiert auf. v1.31.10 AWS IAM Authenticator wurde aktualisiert auf. v0.7.4 Amazon-VPC-CNI-Plugin für Kubernetes aktualisiert zu v1.20.2. Bottlerocket-Version aktualisiert auf v1.47.0.	3. Oktober 2025
1.31.9	eks-local-outposts.4	Neue Plattformversion mit Sicherheitskorrekt	9. August 2025

Kubernetes-Version	Amazon-EKS-Plattformversion	Versionshinweise	Datum der Veröffentlichung
		uren und Verbesserungen. Kube-Proxy wurde aktualisiert auf. `v1.31.9` AWS IAM Authenticator wurde aktualisiert auf. `v0.7.2` Das Amazon VPC CNI-Plugin für Kubernetes wurde auf die `v1.20.0` Bottlerocket-Version aktualisiert auf. `v1.43.0`	
`1.31.7`	`eks-local-outposts.3`	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. Kube-Proxy wurde aktualisiert auf. `v1.31.9` AWS IAM Authenticator wurde aktualisiert auf. `v0.7.1` Amazon-VPC-CNI-Plugin für Kubernetes aktualisiert zu `v1.19.5`. Bottlerocket-Version aktualisiert auf `v1.40.0`.	19. Juni 2025

Kubernetes-Version	Amazon-EKS-Plattformversion	Versionshinweise	Datum der Veröffentlichung
1.31.6	eks-local-outposts.2	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen. Bottlerocket-Version aktualisiert auf v1.36.0.	24. April 2025
1.31.6	eks-local-outposts.1	Erste Veröffentlichung von Kubernetes-Version v1.31 für lokale Amazon-EKS-Cluster in Outposts.	9. April 2025

Kubernetes-Version **1.30**

Die folgenden Zugangs-Controller sind für alle 1.30-Plattformversionen aktiviert: CertificateApproval, CertificateSigning, CertificateSubjectRestriction, ClusterTrustBundleAttest, DefaultIngressClass, DefaultStorageClass, DefaultTolerationSeconds, ExtendedResourceToleration, LimitRanger, MutatingAdmissionWebhook, NamespaceLifecycle, NodeRestriction, PersistentVolumeClaimResize, PodSecurity, Priority, ResourceQuota, RuntimeClass, ServiceAccount, StorageObjectInUseProtection, TaintNodesByCondition, ValidatingAdmissionPolicy und ValidatingAdmissionWebhook.

Kubernetes-Version	Amazon-EKS-Plattformversion	Versionshinweise	Datum der Veröffentlichung
1.30.14	eks-local-outposts.10	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. AWS IAM Authenticator wurde	23. Dezember 2025

Kubernetes-Version	Amazon-EKS-Plattformversion	Versionshinweise	Datum der Veröffentlichung
		aktualisiert auf. `v0.7.8` Amazon-VPC-CNI-Plugin für Kubernetes aktualisiert zu `v1.20.4`. Bottlerocket-Version aktualisiert auf `v1.52.0`.	
`1.30.14`	`eks-local-outposts.7`	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. Kube-Proxy wurde aktualisiert auf. `v1.30.14` AWS IAM Authenticator wurde aktualisiert auf. `v0.7.4` Amazon-VPC-CNI-Plugin für Kubernetes aktualisiert zu `v1.20.2`. Bottlerocket-Version aktualisiert auf `v1.47.0`.	3. Oktober 2025

Kubernetes-Version	Amazon-EKS-Plattformversion	Versionshinweise	Datum der Veröffentlichung
1.30.13	eks-local-outposts.6	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. Kube-Proxy wurde aktualisiert auf. v1.30.13 AWS IAM Authenticator wurde aktualisiert auf. v0.7.2 Amazon-VPC-CNI-Plugin für Kubernetes aktualisiert zu v1.20.0. Bottlerocket-Version aktualisiert auf v1.43.0.	09. August 2025
1.30.11	eks-local-outposts.5	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. Kube-Proxy wurde aktualisiert auf. v1.30.11 AWS IAM Authenticator wurde aktualisiert auf. v0.7.1 Das Amazon VPC CNI-Plugin für Kubernetes wurde auf die v1.19.5 Bottlerocket-Version aktualisiert auf. v1.40.0	19. Juni 2025

Kubernetes-Version	Amazon-EKS-Plattformversion	Versionshinweise	Datum der Veröffentlichung
1.30.10	eks-local-outposts.4	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen. Bottlerocket-Version aktualisiert auf v1.36.0.	24. April 2025
1.30.10	eks-local-outposts.3	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. Kube-Proxy wurde aktualisiert auf. v1.30.10 AWS IAM Authenticator wurde aktualisiert auf. v0.6.29 Amazon-VPC-CNI-Plugin für Kubernetes aktualisiert zu v1.19.2. CoreDNS wurde aktualisiert auf. v1.11.4 AWS Cloud Controller Manager wurde aktualisiert auf. v1.30.8 Bottlerocket-Version aktualisiert auf v1.34.0.	27. März 2025

Kubernetes-Version	Amazon-EKS-Plattformversion	Versionshinweise	Datum der Veröffentlichung
1.30.7	eks-local-outposts.2	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. Kube-Proxy wurde aktualisiert auf. v1.30.7 AWS IAM Authenticator wurde aktualisiert auf. v0.6.28 Amazon-VPC-CNI-Plugin für Kubernetes aktualisiert zu v1.19.0. Die Bottlerocket-Version wurde aktualisiert auf. v1.29.0	10. Januar 2025
1.30.5	eks-local-outposts.1	Erste Veröffentlichung von Kubernetes-Version v1.30 für lokale Amazon-EKS-Cluster in Outposts.	13. November 2024

Kubernetes-Version **1.29**

Die folgenden Zugangs-Controller sind für alle 1.29-Plattformversionen aktiviert: CertificateApproval, CertificateSigning, CertificateSubjectRestriction, ClusterTrustBundleAttest, DefaultIngressClass, DefaultStorageClass, DefaultTolerationSeconds, ExtendedResourceToleration, LimitRanger, MutatingAdmissionWebhook, NamespaceLifecycle, NodeRestriction, PersistentVolumeClaimResize, PodSecurity, Priority, ResourceQuota, RuntimeClass, ServiceAccount, StorageObjectInUseProtection, TaintNodesByCondition, ValidatingAdmissionPolicy und ValidatingAdmissionWebhook.

Kubernetes-Version	Amazon-EKS-Plattformversion	Versionshinweise	Datum der Veröffentlichung
1.29.15	eks-local-outposts.13	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. AWS IAM Authenticator wurde aktualisiert auf. v0.7.8 Amazon-VPC-CNI-Plugin für Kubernetes aktualisiert zu v1.20.4. Bottlerocket-Version aktualisiert auf v1.52.0.	23. Dezember 2025
v1.29.15	eks-local-outposts.10	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. AWS IAM Authenticator wurde aktualisiert auf. v0.7.4 Amazon-VPC-CNI-Plugin für Kubernetes aktualisiert zu v1.20.2. Bottlerocket-Version aktualisiert auf v1.47.0.	3. Oktober 2025
v1.29.15	eks-local-outposts.9	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. AWS IAM Authenticator wurde aktualisiert auf.	9. August 2025

Kubernetes-Version	Amazon-EKS-Plattformversion	Versionshinweise	Datum der Veröffentlichung
		v0.7.2 Amazon-VPC-CNI-Plugin für Kubernetes aktualisiert zu v1.20.0. Bottlerocket-Version aktualisiert auf v1.43.0.	
v1.29.15	eks-local-outposts.8	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. Kube-Proxy wurde aktualisiert auf. v1.29.15 AWS IAM Authenticator wurde aktualisiert auf. v0.7.1 Amazon-VPC-CNI-Plugin für Kubernetes aktualisiert zu v1.19.5. Bottlerocket-Version aktualisiert auf v1.40.0.	19. Juni 2025
v1.29.14	eks-local-outposts.7	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen. Bottlerocket-Version aktualisiert auf v1.36.0.	24. März 2025

Kubernetes-Version	Amazon-EKS-Plattformversion	Versionshinweise	Datum der Veröffentlichung
v1.29.14	eks-local-outposts.6	Neue Plattformversion mit Sicherheitskorrekturen und -verbesserungen. kube-proxy auf v1.29.14 aktualisiert. Amazon-VPC-CNI-Plugin für Kubernetes aktualisiert zu v1.19.2. CoreDNS wurde aktualisiert auf. v1.11.4 AWS Cloud Controller Manager wurde aktualisiert auf. v1.29.8 Bottlerocket-Version aktualisiert auf v1.34.0.	27. März 2025
v1.29.11	eks-local-outposts.5	Neue Plattformversion mit Sicherheitskorrekturen und -verbesserungen. kube-proxy auf v1.29.11 aktualisiert. Amazon-VPC-CNI-Plugin für Kubernetes aktualisiert zu v1.19.0. Das CoreDNS-Image wurde aktualisiert auf. v1.11.3 Die Bottlerocket-Version wurde aktualisiert auf. v1.29.0	10. Januar 2025

Kubernetes-Version	Amazon-EKS-Plattformversion	Versionshinweise	Datum der Veröffentlichung
1.29.9	eks-local-outposts.4	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. Kube-Proxy wurde aktualisiert auf. v1.29.9 AWS IAM Authenticator wurde aktualisiert auf. v0.6.26 Die Bottlerocket-Version wurde aktualisiert auf. v1.26.0	8. November 2024
1.29.6	eks-local-outposts.3	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen. Die Bottlerocket-Version wurde aktualisiert auf. v1.22.0	22. Oktober 2024
1.29.6	eks-local-outposts.2	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen. Die Bottlerocket-Version wurde aktualisiert auf. v1.21.0	27. August 2024
1.29.6	eks-local-outposts.1	Erste Veröffentlichung von Kubernetes-Version v1.29 für lokale Amazon-EKS-Cluster in Outposts.	20. August 2024

Erstellung einer VPC und von Subnetzen für Amazon-EKS-Cluster in AWS Outposts

Wenn Sie einen lokalen Cluster erstellen, geben Sie eine VPC und mindestens ein privates Subnetz an, das auf Outposts ausgeführt wird. Dieses Thema bietet einen Überblick über die VPC- und Subnetzanforderungen und -überlegungen für Ihren lokalen Cluster.

VPC-Anforderungen und -Überlegungen

Wenn Sie einen lokalen Cluster erstellen, muss die von Ihnen angegebene VPC die folgenden Anforderungen und Überlegungen erfüllen:

- Stellen Sie sicher, dass die VPC über ausreichend IP-Adressen für den lokalen Cluster, alle Knoten und andere Kubernetes-Ressourcen verfügt, die Sie erstellen möchten. Wenn die VPC, die Sie verwenden möchten, nicht über genügend IP-Adressen verfügt, erhöhen Sie die Anzahl der verfügbaren IP-Adressen. Das ist möglich, indem Sie zusätzliche CIDR-Blöcke (Classless Inter-Domain Routing) mit Ihrer VPC verbinden. Sie können entweder vor oder nach der Erstellung Ihres Clusters private (RFC 1918) und öffentliche (nicht RFC 1918) CIDR-Blöcke mit Ihrer VPC verbinden. Es kann bis zu 5 Stunden dauern, bis ein CIDR-Block, den Sie einem VPC zugeordnet haben, von einem Cluster erkannt wird.
- Der VPC dürfen keine IP-Präfixe oder IPv6-CIDR-Blöcke zugewiesen werden. Aufgrund dieser Einschränkungen sind die Informationen, die unter Weitere IP-Adressen mit Präfixen und the section called “IPv6” für Amazon-EKS-Knoten zuweisen behandelt werden, für Ihre VPC nicht anwendbar.
- Die VPC verfügt über einen DNS-Hostnamen und die DNS-Auflösung ist aktiviert. Ohne diese Funktionen kann der lokale Cluster nicht erstellt werden, und Sie müssen die Funktionen aktivieren und Ihren Cluster neu erstellen. Weitere Informationen finden Sie unter DNS-Attribute für Ihre VPC im Amazon-VPC-Benutzerhandbuch.
- Um über Ihr lokales Netzwerk auf Ihren lokalen Cluster zuzugreifen, muss die VPC mit der lokalen Gateway-Routing-Tabelle Ihres Outpost verknüpft sein. Weitere Informationen finden Sie unter VPC-Zuordnungen im AWS-Outposts-Benutzerhandbuch.

Subnetz-Anforderungen und -Überlegungen

Geben Sie beim Erstellen des Clusters mindestens ein privates Subnetz an. Wenn Sie mehr als ein Subnetz angeben, werden die Kubernetes-Steuerebene-Instances gleichmäßig über die Subnetze

hinweg verteilt. Wenn mehr als ein Subnetz angegeben wird, müssen die Subnetze auf demselben Outpost vorhanden sein. Darüber hinaus müssen die Subnetze auch über die richtigen Routen und Sicherheitsgruppen-Berechtigungen verfügen, um miteinander kommunizieren zu können. Wenn Sie einen lokalen Cluster erstellen, müssen die von Ihnen angegebenen Subnetze die folgenden Anforderungen erfüllen:

- Die Subnetze befinden sich alle auf demselben logischen Outpost.
- Die Subnetze verfügen zusammen über mindestens drei verfügbare IP-Adressen für die Kubernetes-Instances der Steuerebene. Wenn drei Subnetze angegeben werden, muss jedes Subnetz über mindestens eine verfügbare IP-Adresse verfügen. Wenn zwei Subnetze angegeben werden, muss jedes Subnetz über mindestens zwei verfügbare IP-Adressen verfügen. Wenn ein Subnetz angegeben wird, muss das Subnetz über mindestens drei verfügbare IP-Adressen verfügen.
- Die Subnetze verfügen über eine Weiterleitung zum lokalen Gateway des Outpost-Racks, um über Ihr lokales Netzwerk auf den Kubernetes-API-Server zugreifen zu können. Wenn Subnetze über keine Weiterleitung zum lokalen Gateway des Outpost-Racks verfügen, müssen Sie von innerhalb der VPC mit Ihrem Kubernetes-API-Server kommunizieren.
- Die Subnetze müssen eine IP-Adressen-basierte Benennung aufweisen. Die ressourcenbasierte Benennung von Amazon EC2 wird nicht durch Amazon EKS unterstützt.

Subnetz-Zugang zu AWS-Services

Die privaten Subnetze des lokalen Clusters in Outposts müssen mit regionalen AWS-Services kommunizieren können. Dazu können Sie ein NAT-Gateway für den ausgehenden Internetzugang verwenden oder, wenn Sie den gesamten Datenverkehr innerhalb Ihrer VPC privat halten möchten, Schnittstellen-VPC-Endpunkte verwenden.

Verwenden eines NAT-Gateways

Die privaten Subnetze des lokalen Clusters in Outposts müssen über eine zugeordnete Routing-Tabelle mit einer Weiterleitung zu einem NAT-Gateway verfügen, das sich in einem öffentlichen Subnetz in der übergeordneten Availability Zone des Outposts befindet. Das öffentliche Subnetz muss über eine Route zu einem Internet-Gateway verfügen. Das NAT-Gateway ermöglicht einen ausgehenden Zugriff auf das Internet und verhindert unerwünschte eingehende Verbindungen aus dem Internet zu Instances im Outpost.

Verwendung von -Schnittstellen-VPC-Endpunkten

Wenn die privaten Subnetze des lokalen Clusters in Outposts keine ausgehende Internetverbindung haben oder wenn Sie den gesamten Datenverkehr innerhalb Ihrer VPC privat halten möchten, müssen Sie die folgenden Schnittstellen-VPC-Endpunkte und den Gateway-Endpunkt in einem regionalen Subnetz erstellen, bevor Sie Ihren Cluster erstellen.

Endpunkt	Endpunkttyp
com.amazonaws. *region-code* .ssm	Schnittstelle
com.amazonaws. *region-code* .ssmmessages	Schnittstelle
com.amazonaws. *region-code* .ec2messages	Schnittstelle
com.amazonaws. *region-code* .ec2	Schnittstelle
com.amazonaws. *region-code* .secretsmanager	Schnittstelle
com.amazonaws. *region-code* .logs	Schnittstelle
com.amazonaws. *region-code* .sts	Schnittstelle
com.amazonaws. *region-code* .ecr.api	Schnittstelle
com.amazonaws. *region-code* .ecr.dkr	Schnittstelle
com.amazonaws. *region-code* .s3	Gateway

Die Endpunkte müssen die folgenden Anforderungen erfüllen:

- Sie müssen in einem privaten Subnetz erstellt werden, das sich in der übergeordneten Availability Zone Ihres Outposts befindet.
- Private DNS-Namen müssen aktiviert sein.
- Sie müssen über eine angefügte Sicherheitsgruppe verfügen, die eingehenden HTTPS-Datenverkehr aus dem CIDR-Bereich des privaten Outpost-Subnetzes zulässt.

Für die Erstellung von Endpunkten fallen Gebühren an. Weitere Informationen erhalten Sie unter AWS PrivateLink-Preise. Wenn Ihre Pods Zugriff auf andere AWS-Services benötigen, müssen Sie zusätzliche Endpunkte erstellen. Eine umfassende Liste der Endpunkte finden Sie unter AWS-Services, die in AWS PrivateLink integriert sind.

Erstellen einer VPC

Sie können eine VPC erstellen, welche die oben genannten Anforderungen erfüllt, indem Sie eine der folgenden AWS-CloudFormation-Vorlagen verwenden:

- Vorlage 1 – Diese Vorlage erstellt eine VPC mit einem privaten Subnetz in dem Outpost und einem öffentlichen Subnetz in der AWS-Region. Das private Subnetz verfügt über eine Weiterleitung zum Internet über ein NAT-Gateway, das sich im öffentlichen Subnetz in der AWS-Region befindet. Diese Vorlage kann verwendet werden, um einen lokalen Cluster in einem Subnetz mit ausgehendem Internetzugriff zu erstellen.
- Vorlage 2 – Diese Vorlage erstellt eine VPC mit einem privaten Subnetz in dem Outpost und der Mindestanzahl von VPC-Endpunkten, die erforderlich sind, um einen lokalen Cluster in einem Subnetz zu erstellen, das über keinen eingehenden oder ausgehenden Internetzugriff verfügt (auch als privates Subnetz bezeichnet).

Vorbereitung lokaler Amazon-EKS-Cluster in AWS Outposts für Netzwerkunterbrechungen

Wenn Ihr lokales Netzwerk die Verbindung zur AWS Cloud verloren hat, können Sie Ihren lokalen Amazon-EKS-Cluster weiterhin in einem Outpost verwenden. In diesem Thema wird beschrieben, wie Sie Ihren lokalen Cluster auf Netzwerkunterbrechungen vorbereiten können, und verwandte Überlegungen.

- Lokale Cluster gewährleisten Stabilität und einen unterbrechungsfreien Betrieb bei vorübergehenden, ungeplanten Netzwerkausfällen. AWS Outposts bleibt ein vollständig verbundenes Angebot, das als Erweiterung der AWS Cloud in Ihrem Rechenzentrum fungiert. Im Falle einer Netzwerkunterbrechung zwischen Ihrem Outpost und der AWS Cloud empfehlen wir, die Verbindung wiederherzustellen. Eine Anleitung dazu finden Sie in der Checkliste zur Fehlerbehebung im AWS-Rack-Netzwerk im AWS-Outposts-Benutzerhandbuch. Weitere Informationen zum Beheben von Problemen mit lokalen Clustern finden Sie unter the section called “Fehlersuche bei Clustern”.

- Outposts geben eine `ConnectedStatus`-Metrik aus, mit der Sie den Konnektivitätsstatus Ihres Outposts überwachen können. Weitere Informationen finden Sie unter Outposts-Metriken im AWS-Benutzerhandbuch.
- Lokale Cluster verwenden IAM als standardmäßigen Authentifizierungsmechanismus mit dem AWS Authenticator Identity and Access Management für Kubernetes. IAM ist während Netzwerkunterbrechungen nicht verfügbar. Daher unterstützen lokale Cluster einen alternativen Authentifizierungsmechanismus mithilfe von `x.509`-Zertifikaten, die Sie verwenden können, um eine Verbindung zu Ihrem Cluster herzustellen, wenn die Netzwerkverbindung unterbrochen ist. Informationen zum Abrufen und Verwenden eines `x.509`-Zertifikats für Ihren Cluster finden Sie unter the section called “Authentifizierung bei Ihrem lokalen Cluster während einer Netzwerkunterbrechung”.
- Wenn Sie bei Netzwerkunterbrechungen nicht auf Route 53 zugreifen können, sollten Sie lokale DNS-Server in Ihrer On-Premises-Umgebung verwenden. Die Kubernetes-Instances der Steuerebene verwenden statische IP-Adressen. Sie können die Hosts, die Sie für die Verbindung mit Ihrem Cluster verwenden, mit dem Hostnamen und den IP-Adressen des Endpunkts als Alternative zur Verwendung lokaler DNS-Server konfigurieren. Weitere Informationen finden Sie unter DNS im AWS-Outposts-Benutzerhandbuch.
- Wenn Sie bei Netzwerkunterbrechungen mit einem Anstieg des Anwendungsdatenverkehrs rechnen, können Sie in Ihrem Cluster bei Verbindung mit der Cloud freie Rechenkapazität bereitstellen. Amazon-EC2-Instances sind im Preis von AWS Outposts enthalten. Der Betrieb von Ersatz-Instances hat also keinen Einfluss auf Ihre AWS-Nutzungskosten.
- Während Netzwerkunterbrechungen, die das Erstellen, Aktualisieren und Skalieren von Workloads ermöglichen, müssen die Container-Images Ihrer Anwendung über das lokale Netzwerk zugänglich sein und Ihr Cluster muss über genügend Kapazität verfügen. Lokale Cluster hosten keine Container-Registry für Sie. Wenn die Pods zuvor auf diesen Knoten ausgeführt wurden, werden Container-Images auf den Knoten zwischengespeichert. Wenn Sie die Container-Images Ihrer Anwendung in der Regel aus Amazon ECR in der Cloud beziehen, sollten Sie erwägen, einen lokalen Cache oder eine lokale Registry auszuführen. Ein lokaler Cache oder eine lokale Registry ist hilfreich, wenn Sie während einer Netzwerkunterbrechung Workload-Ressourcen erstellen, aktualisieren und skalieren müssen.
- Lokale Cluster verwenden Amazon EBS als Standardspeicherklasse für persistente Volumes und den Amazon-EBS-CSI-Treiber, um den Lebenszyklus persistenter Amazon-EBS-Volumes zu verwalten. Während Netzwerkunterbrechungen können Pods, die von Amazon EBS gesichert werden, nicht erstellt, aktualisiert oder skaliert werden. Dies liegt daran, dass diese Vorgänge Aufrufe an die Amazon-EBS-API in der Cloud erfordern. Wenn Sie statusbehaftete

Workloads auf lokalen Clustern bereitstellen und während Netzwerktrennungen Vorgänge zum Erstellen, Aktualisieren oder Skalieren benötigen, sollten Sie die Verwendung eines alternativen Speichermechanismus in Betracht ziehen.

- Amazon-EBS-Snapshots können nicht erstellt oder gelöscht werden, wenn AWS Outposts kein Zugriff auf die relevanten AWS-APIs in der Region (z. B. die APIs für Amazon EBS oder Amazon S3) möglich ist.
- Bei der Integration von ALB (Ingress) mit AWS Certificate Manager (ACM) werden Zertifikate übertragen und im Speicher der ALB-Rechen-Instance von AWS Outposts gespeichert. Die aktuelle TLS-Terminierung wird im Falle einer Trennung von der AWS-Region weiter betrieben. Mutationsvorgänge in diesem Kontext schlagen fehl (z. B. neue Eingangsdefinitionen, neue API-Vorgänge für ACM-basierte Zertifikate, ALB-Rechenskalierung oder Zertifikatsrotation). Weitere Informationen finden Sie unter Problembehandlung bei der Erneuerung verwalteter Zertifikate im Benutzerhandbuch zum AWS-Zertifikatsmanager.
- Die Protokolle der Amazon-EKS-Steuerebene werden während Netzwerkunterbrechungen lokal auf Kubernetes-Instances der Steuerebene zwischengespeichert. Nach der erneuten Verbindung werden die Protokolle an CloudWatch Logs in der übergeordneten AWS-Region gesendet. Sie können Prometheus-, Grafana- oder Amazon-EKS-Partnerlösungen verwenden, um den Cluster lokal mithilfe des Metrik-Endpunkts des Kubernetes-API-Servers oder mithilfe von Fluent Bit für Protokolle zu überwachen.
- Wenn Sie AWS Load Balancer Controller in Outposts für den Anwendungsdatenverkehr verwenden, erhalten vorhandene Pods, denen das AWS Load Balancer Controller vorangestellt ist, während der Netzwerkunterbrechung weiterhin Datenverkehr. Neue Pods, die bei Netzwerkunterbrechungen erstellt wurden, empfangen keinen Datenverkehr, bis der Outpost wieder mit der AWS Cloud verbunden ist. Erwägen Sie, die Replikatanzahl für Ihre Anwendungen festzulegen, während Sie mit der AWS Cloud verbunden sind, um Ihre Skalierungsanforderungen während Netzwerktrennungen zu erfüllen.
- Das Amazon-VPC-CNI-Plugin für Kubernetes ist standardmäßig auf den sekundären IP-Modus eingestellt. Es ist mit `WARM_ENI_TARGET = 1` konfiguriert, wodurch das Plugin „eine vollständige elastische Netzwerkschnittstelle“ mit verfügbaren IP-Adressen bereithalten kann. Erwägen Sie, `WARM_ENI_TARGET`-, `WARM_IP_TARGET`- und `MINIMUM_IP_TARGET`-Werte entsprechend Ihren Skalierungsanforderungen während eines getrennten Zustands zu ändern. Weitere Informationen finden Sie in der Datei readme für das Plugin auf GitHub. Eine Liste der maximalen Anzahl von Pods, die von jedem Instance-Typ unterstützt werden, finden Sie in der Datei eni-max-pods.txt auf GitHub.

Authentifizierung bei Ihrem lokalen Cluster während einer Netzwerkunterbrechung

AWS Identity and Access Management (IAM) ist bei Netzwerkunterbrechungen nicht verfügbar. Sie können sich nicht bei Ihrem lokalen Cluster mit IAM-Anmeldeinformationen authentifizieren, während keine Verbindung besteht. Sie können jedoch über Ihr lokales Netzwerk mithilfe von `x509`-Zertifikaten eine Verbindung zu Ihrem Cluster herstellen, wenn die Verbindung getrennt ist. Sie müssen ein Client `X509`-Zertifikat herunterladen und speichern, das Sie während der Verbindungstrennung verwenden können. In diesem Thema erfahren Sie, wie Sie das Zertifikat erstellen und verwenden, um sich bei Ihrem Cluster zu authentifizieren, wenn dieser sich in einem nicht verbundenen Status befindet.

1. Erstellen einer Zertifikatssignierungsanforderung

   a. Generieren einer Zertifikatssignierungsanforderung.

   ```
   openssl req -new -newkey rsa:4096 -nodes -days 365 \
       -keyout admin.key -out admin.csr -subj "/CN=admin"
   ```

   b. Erstellen Sie eine Anfrage zur Zertifikatssignierung in Kubernetes.

   ```
   BASE64_CSR=$(cat admin.csr | base64 -w 0)
   cat << EOF > admin-csr.yaml
   apiVersion: certificates.k8s.io/v1
   kind: CertificateSigningRequest
   metadata:
     name: admin-csr
   spec:
     signerName: kubernetes.io/kube-apiserver-client
     request: ${BASE64_CSR}
     usages:
     - client auth
   EOF
   ```

2. Erstellen einer Zertifikatssignierungsanforderung mit `kubectl`.

   ```
   kubectl create -f admin-csr.yaml
   ```

3. Überprüfen Sie den Status der Zertifikatssignierungsanforderung.

   ```
   kubectl get csr admin-csr
   ```

   Eine Beispielausgabe sieht wie folgt aus.

```
NAME       AGE   REQUESTOR                      CONDITION
admin-csr  11m   kubernetes-admin               Pending
```

Kubernetes hat die Anfrage zur Zertifikatsignierung erstellt.

4. Genehmigen Sie die Zertifikatssignieranforderung.

```
kubectl certificate approve admin-csr
```

5. Überprüfen Sie erneut den Status der Anfrage zum Signieren des Zertifikats auf Genehmigung.

```
kubectl get csr admin-csr
```

Eine Beispielausgabe sieht wie folgt aus.

```
NAME       AGE   REQUESTOR                      CONDITION
admin-csr  11m   kubernetes-admin               Approved
```

6. Rufen Sie das Zertifikat ab und überprüfen Sie es.

a. Rufen Sie das Zertifikat ab.

```
kubectl get csr admin-csr -o jsonpath='{.status.certificate}' | base64 --decode >
 admin.crt
```

b. Überprüfen Sie das Zertifikat.

```
cat admin.crt
```

7. Erstellen Sie eine Cluster-Rollenbindung für einen `admin`-Benutzer.

```
kubectl create clusterrolebinding admin --clusterrole=cluster-admin \
    --user=admin --group=system:masters
```

8. Generieren Sie eine kubeconfig mit Benutzerbereich für einen getrennten Status.

Sie können eine `kubeconfig`-Datei mit den heruntergeladenen `admin`-Zertifikaten generieren. Ersetzen Sie *`my-cluster`* und *`apiserver-endpoint`* in den folgenden Befehlen.

```
aws eks describe-cluster --name my-cluster \
    --query "cluster.certificateAuthority" \
```

```
    --output text | base64 --decode > ca.crt
```

```
kubectl config --kubeconfig admin.kubeconfig set-cluster my-cluster \
    --certificate-authority=ca.crt --server apiserver-endpoint --embed-certs
```

```
kubectl config --kubeconfig admin.kubeconfig set-credentials admin \
    --client-certificate=admin.crt --client-key=admin.key --embed-certs
```

```
kubectl config --kubeconfig admin.kubeconfig set-context admin@my-cluster \
    --cluster my-cluster --user admin
```

```
kubectl config --kubeconfig admin.kubeconfig use-context admin@my-cluster
```

9. Ziegen Sie Ihre `kubeconfig`-Datei an.

```
kubectl get nodes --kubeconfig admin.kubeconfig
```

10. Wenn Sie bereits über Services in Produktion auf Ihrem Outpost verfügen, überspringen Sie diesen Schritt. Wenn Amazon EKS der einzige Service ist, der in Ihrem Outpost ausgeführt wird, und der Outpost sich nicht in der Produktion befindet, können Sie eine Netzwerkunterbrechung simulieren. Bevor Sie mit Ihrem lokalen Cluster in Produktion gehen, simulieren Sie einen Verbindungsabbruch, um sicherzustellen, dass Sie auf Ihren Cluster zugreifen können, wenn er sich in einem getrennten Status befindet.

    a. Wenden Sie Firewall-Regeln auf die Netzwerkgeräte an, die Ihren Outpost mit der AWS-Region verbinden. Dadurch wird der Service-Link des Outposts getrennt. Sie können keine neuen Instances erstellen. Aktuell ausgeführte Instances verlieren die Konnektivität zur AWS-Region und zum Internet.

    b. Sie können die Verbindung zu Ihrem lokalen Cluster testen, während die Verbindung getrennt ist, indem Sie das `x509`-Zertifikat verwenden. Stellen Sie sicher, dass Sie Ihr `kubeconfig` in das `admin.kubeconfig` ändern, das Sie in einem vorherigen Schritt erstellt haben. Ersetzen Sie *`my-cluster`* durch den Namen Ihres lokalen Clusters.

    ```
    kubectl config use-context admin@my-cluster --kubeconfig admin.kubeconfig
    ```

    Wenn Sie Probleme mit Ihren lokalen Clustern feststellen, während diese sich im getrennten Status befinden, empfehlen wir Ihnen, ein Support-Ticket zu erstellen.

Wählen Sie Instance-Typen und Platzierungsgruppen für Amazon-EKS-Cluster in AWS Outposts basierend auf Kapazitätsüberlegungen aus.

Dieses Thema enthält Anleitungen zur Auswahl des Instance-Typs der Kubernetes-Steuerebene und (optional) zur Verwendung von Platzierungsgruppen, um Hochverfügbarkeitsanforderungen für Ihren lokalen Amazon-EKS-Cluster in einem Outpost zu erfüllen.

Bevor Sie einen Instance-Typ (z. B. `m5`, `c5`, oder `r5`) auswählen, den Sie für die Kubernetes-Steuerebene Ihres lokalen Clusters in Outposts verwenden möchten, überprüfen Sie die Instance-Typen, die in Ihrer Outpost-Konfiguration verfügbar sind. Nachdem Sie die verfügbaren Instance-Typen identifiziert haben, wählen Sie die Instance-Größe (z. B. `large`, `xlarge`, oder `2xlarge`) basierend auf der Anzahl der Knoten, die Ihr Workload benötigt. Die folgende Tabelle enthält Empfehlungen für die Auswahl einer Instance-Größe.

> **Note**
>
> Die Instance-Größen müssen auf Ihren Outposts eingestellt werden. Stellen Sie sicher, dass Sie über genügend Kapazität für drei Instances der auf Ihren Outposts verfügbaren Größe für die Lebensdauer Ihres lokalen Clusters verfügen. Eine Liste der verfügbaren Amazon-EC2-Instance-Typen finden Sie in den Abschnitten „Rechenleistung“ und „Speicher“ unter AWS-Outposts-Rack-Features.

Anzahl der Knoten	Instance-Größe der Kubernetes-Steuerebene
1–20	`large`
21–100	`xlarge`
101–250	`2xlarge`
251–500	`4xlarge`

Der Speicher für die Kubernetes-Steuerebene erfordert 246 GB Amazon EBS-Speicher für jeden lokalen Cluster, um die erforderlichen IOPS von `etcd` zu erfüllen. Bei der Erstellung des lokalen Clusters werden die Amazon-EBS-Volumes automatisch für Sie bereitgestellt.

Platzierung der Steuerebene

Wenn Sie keine Platzierungsgruppe mit der `OutpostConfig.ControlPlanePlacement.GroupName`-Eigenschaft angeben, erhalten die für Ihre Kubernetes-Steuerebene bereitgestellten Amazon-EC2-Instances keine spezifische Durchsetzung der Hardware-Platzierung über die zugrunde liegende Kapazität, die in Ihrem Outpost verfügbar ist.

Sie können Platzierungsgruppen verwenden, um die Hochverfügbarkeitsanforderungen für Ihren lokalen Amazon EKS-Cluster auf einem Outpost zu erfüllen. Indem Sie bei der Cluster-Erstellung eine Platzierungsgruppe angeben, beeinflussen Sie die Platzierung der Instances der Kubernetes-Steuerebene. Die Instances sind auf unabhängige zugrundeliegende Hardware (Racks oder Hosts) verteilt, wodurch die Auswirkungen korrelierter Instances bei Hardwareausfällen minimiert werden.

Die Art der Verteilung, die Sie konfigurieren können, hängt von der Anzahl der Outpost-Racks in Ihrer Bereitstellung ab.

- Bereitstellungen mit einem oder zwei physischen Racks in einem einzelnen logischen Outpost – Sie müssen über mindestens drei Hosts verfügen, die mit dem Instance-Typ konfiguriert sind, den Sie für Ihre Instances der Kubernetes-Steuerebene auswählen. Eine Spread-Placement-Gruppe, die eine Verteilung auf Host-Ebene verwendet, stellt sicher, dass alle Instances der Kubernetes-Steuerebene auf verschiedenen Hosts innerhalb der zugrunde liegenden Racks ausgeführt werden, die in Ihrer Outpost-Bereitstellung verfügbar sind.
- Bereitstellungen mit drei oder mehr physischen Racks in einem einzigen logischen Outpost – Sie müssen mindestens drei Hosts mit dem Instance-Typ konfiguriert haben, den Sie für Ihre Instances der Kubernetes-Steuerebene auswählen. Eine Spread-Placement-Gruppe, die eine Verteilung auf Rack-Ebene verwendet, stellt sicher, dass alle Instances der Kubernetes-Steuerebene auf unterschiedlichen Racks in Ihrer Outpost-Bereitstellung ausgeführt werden. Sie können alternativ die auf Host-Ebene verteilte Platzierungsgruppe verwenden, wie in der vorherigen Option beschrieben.

Sie sind für die Erstellung der gewünschten Platzierungsgruppe verantwortlich. Sie geben die Platzierungsgruppe an, wenn Sie die `CreateCluster`-API aufrufen. Weitere Informationen zu Platzierungsgruppen und deren Erstellung finden Sie unter Platzierungsgruppen im Amazon-EC2-Benutzerhandbuch.

- Wenn eine Platzierungsgruppe angegeben wird, muss verfügbare Slot-Kapazität auf Ihrem Outpost vorhanden sein, um erfolgreich einen lokalen Amazon-EKS-Cluster zu erstellen. Die

Kapazität variiert je nachdem, ob Sie den Host- oder Rack-Spread-Typ verwenden. Wenn nicht genügend Kapazität vorhanden ist, verbleibt der Cluster im `Creating`-Zustand. Sie können das `Insufficient Capacity Error` auf dem Zustandsfeld der DescribeCluster-API-Reaktion überprüfen. Sie müssen Kapazität freigeben, damit der Erstellungsprozess fortgesetzt werden kann.

- Während der Plattform- und Versionsaktualisierungen des Amazon-EKS-Clusters werden die Instances der Kubernetes-Steuerebene aus Ihrem Cluster durch neue Instances ersetzt, die eine fortlaufende Aktualisierungsstrategie verwenden. Während dieses Ersetzungsprozesses wird jede Instance der Steuerebene beendet, wodurch ihr jeweiliger Slot freigegeben wird. Eine neue aktualisierte Instance wird an ihrer Stelle bereitgestellt. Die aktualisierte Instance wird möglicherweise in den freigegebenen Slot platziert. Wenn der Slot von einer anderen unabhängigen Instance verbraucht wird und keine Kapazität mehr vorhanden ist, die die erforderliche verteilte Topologieanforderung erfüllt, verbleibt der Cluster im `Updating`-Zustand. Sie können das entsprechende `Insufficient Capacity Error` im Zustandsfeld der DescribeCluster-API-Reaktion anzeigen. Sie müssen Kapazitäten freigeben, damit der Aktualisierungsprozess fortschreiten und vorherige Hochverfügbarkeitsniveaus wiederherstellen kann.
- Sie können in jeder Region maximal 500 Platzierungsgruppen pro Konto in jeder AWS-Region erstellen. Weitere Informationen finden Sie unter Allgemeine Regeln und Einschränkungen im Amazon-EC2-Benutzerhandbuch.

Fehlerbehebung bei lokalen Amazon EKS-Clustern auf AWS Outposts

In diesem Thema werden einige häufige Fehler behandelt, die bei der Verwendung lokaler Cluster auftreten können, und wie Sie diese beheben können. Lokale Cluster ähneln Amazon-EKS-Clustern in der Cloud, es gibt jedoch einige Unterschiede in der Art und Weise, wie sie von Amazon EKS verwaltet werden.

> **Important**
>
> Beenden Sie niemals eine verwaltete lokale `Kubernetes` EKS-Cluster-Steuerebeneninstanz, die auf Outpost läuft, es sei denn, Sie werden ausdrücklich vom Support dazu aufgefordert. AWS Das Beenden dieser Instances stellt ein Risiko für die Verfügbarkeit des lokalen Cluster-Services dar, einschließlich des Verlusts des lokalen Clusters, falls mehrere Instances gleichzeitig beendet werden. Instanzen der lokalen `Kubernetes` EKS-Cluster-Steuerebene

werden durch das Tag auf der Instanzkonsole identifiziert. `eks-local:controlplane-name` EC2

API-Verhalten

Lokale Cluster werden über die Amazon-EKS-API erstellt, aber asynchron ausgeführt. Dies bedeutet, dass Anforderungen an die Amazon-EKS-API für lokale Cluster sofort zurückgegeben werden. Diese Anforderungen können jedoch erfolgreich sein, aufgrund von Eingabevalidierungsfehlern schnell scheitern oder fehlschlagen und beschreibende Validierungsfehler aufweisen. Dieses Verhalten ähnelt dem der Kubernetes-API.

Lokale Cluster wechseln nicht in einen `FAILED`-Status. Amazon EKS versucht, den Cluster-Status kontinuierlich mit dem vom Benutzer angeforderten gewünschten Status abzugleichen. Infolgedessen kann ein lokaler Cluster für längere Zeit im `CREATING`-Status verbleiben, bis das zugrunde liegende Problem behoben ist.

Beschreibung des Bereichs Cluster-Integrität

Probleme mit lokalen Clustern können mit dem Amazon AWS EKS-CLI-Befehl describe-cluster erkannt werden. Probleme mit lokalen Clustern werden durch das `cluster.health`-Feld der Antwort des `describe-cluster`-Befehls angezeigt. Die in diesem Feld enthaltene Nachricht enthält einen Fehlercode, eine beschreibende Meldung und eine zugehörige Ressource. IDs Diese Informationen sind nur über die Amazon EKS-API und AWS CLI verfügbar. Im folgenden Beispiel ersetzen Sie es *`my-cluster`* durch den Namen Ihres lokalen Clusters.

```
aws eks describe-cluster --name my-cluster --query 'cluster.health'
```

Eine Beispielausgabe sieht wie folgt aus.

```
{
    "issues": [
        {
            "code": "ConfigurationConflict",
            "message": "The instance type 'm5.large' is not supported in Outpost 'my-
outpost-arn'.",
            "resourceIds": [
                "my-cluster-arn"
            ]
        }
    ]
```

```
}
```

Wenn das Problem nicht behoben werden kann, müssen Sie möglicherweise den lokalen Cluster löschen und einen neuen erstellen. Versuchen Sie beispielsweise, einen Cluster mit einem Instance-Typ bereitzustellen, der in Ihrem Outpost nicht verfügbar ist. Die folgende Tabelle enthält allgemeine Fehler im Zusammenhang mit der Integrität.

Fehlerszenario	Code	Fehlermeldung	ResourceIds
Die bereitgestellten Subnetze konnten nicht gefunden werden.	`ResourceNotFound`	`The subnet ID subnet-id does not exist`	Das gesamte bereitgestellte Subnetz IDs
Bereitgestellte Subnetze gehören nicht zur selben VPC.	`ConfigurationConflict`	`Subnets specified must belong to the same VPC`	Alles bereitgestellte Subnetz IDs
Einige bereitgestellte Subnetze gehören nicht zum angegebenen Outpost.	`ConfigurationConflict`	`Subnet subnet-id expected to be in outpost-arn , but is in other-outpost-arn`	Problematische Subnetz-ID
Einige bereitgestellte Subnetze gehören zu keinem Outpost.	`ConfigurationConflict`	`Subnet subnet-id is not part of any Outpost`	Problematische Subnetz-ID
Einige bereitgestellte Subnetze verfügen nicht über genügend freie Adressen, um elastische Netzwerkschnittstellen für Instances der Steuerebene zu erstellen.	`ResourceLimitExceeded`	`The specified subnet does not have enough free addresses to satisfy the request.`	Problematische Subnetz-ID

Fehlerszenario	Code	Fehlermeldung	ResourceIds
Der angegebene Instance-Typ der Steuerebene wird von Ihrem Outpost nicht unterstützt.	`ConfigurationConflict`	`The instance type type is not supported in Outpost outpost-arn`	Cluster-ARN
Sie haben eine EC2 Amazon-Instance auf Kontrollebene beendet oder `run-instance` waren erfolgreich, aber der beobachtete Status hat sich geändert`Terminated` . Dies kann für einen bestimmten Zeitraum passieren, nachdem Ihr Outpost wieder eine Verbindung hergestellt hat und interne Amazon EBS-Fehler dazu führen, dass ein EC2 interner Amazon-Workflow fehlschlägt.	`InternalFailure`	`EC2 instance state "Terminated" is unexpected`	Cluster-ARN

Fehlerszenario	Code	Fehlermeldung	ResourceIds
Sie verfügen über unzureichende Kapazität auf Ihrem Outpost. Dies kann auch passieren, wenn ein Cluster erstellt wird und ein Outpost von der Region getrennt wird. AWS	`ResourceLimitExceeded`	`There is not enough capacity on the Outpost to launch or start the instance.`	Cluster-ARN
Ihr Konto hat das Kontingent Ihrer Sicherheitsgruppe überschritten.	`ResourceLimitExceeded`	Von der Amazon EC2 API zurückgegebene Fehlermeldung	Ziel-VPC-ID
Ihr Konto hat Ihr Kontingent für die elastische Netzwerkschnittstelle überschritten.	`ResourceLimitExceeded`	Von der Amazon EC2 API zurückgegebene Fehlermeldung	Ziel-Subnetz-ID
Instanzen der Kontrollebene waren über AWS Systems Manager nicht erreichbar. Informationen zur Lösung finden Sie unter the section called “Instanzen der Kontrollebene sind über AWS Systems Manager nicht erreichbar”.	`ClusterUnreachable`	Amazon-EKS-Steuerebene-Instances sind nicht über SSM erreichbar. Bitte überprüfen Sie Ihre SSM- und Netzwerkkonfiguration und lesen Sie die Dokumentation zur Fehlerbehebung bei EKS on Outposts.	EC2 Amazon-Instanz IDs

Fehlerszenario	Code	Fehlermeldung	ResourceIds
Beim Abrufen von Details für eine verwaltete Sicherheitsgruppe oder eine elastische Netzwerkschnittstelle ist ein Fehler aufgetreten.	Basiert auf dem EC2 Amazon-Client-Fehlercode.	Von der Amazon EC2 API zurückgegebene Fehlermeldung	Alle verwalteten Sicherheitsgruppen IDs
Beim Autorisieren oder Widerrufen von Eingangsregeln für Sicherheitsgruppen ist ein Fehler aufgetreten. Dies gilt sowohl für die Sicherheitsgruppen des Clusters als auch der Steuerebene.	Basiert auf dem EC2 Amazon-Client-Fehlercode.	Von der Amazon EC2 API zurückgegebene Fehlermeldung	Problematische Sicherheitsgruppen-ID
Beim Löschen einer elastischen Netzwerkschnittstelle für eine Instance der Steuerebene ist ein Fehler aufgetreten.	Basiert auf dem EC2 Amazon-Client-Fehlercode.	Von der Amazon EC2 API zurückgegebene Fehlermeldung	Problematische ID der Elastic-Network-Schnittstelle

In der folgenden Tabelle sind Fehler anderer AWS Dienste aufgeführt, die im Feld „Integrität“ der `describe-cluster` Antwort aufgeführt sind.

EC2 Amazon-Fehlercode	Code für Cluster-Integritätsprobleme	Description
`AuthFailure`	`AccessDenied`	Dieser Fehler kann aus einer Vielzahl von Gründen auftreten. Der häufigste

EC2 Amazon-Fehlercode	Code für Cluster-Integrität sprobleme	Description
		Grund ist, dass Sie versehentlich ein Tag entfernt haben, das der Service verwendet, um die Richtlinie für die mit dem Service verknüpfte Rolle von der Steuerebene herabzustufen. In diesem Fall kann Amazon EKS diese AWS Ressourcen nicht mehr verwalten und überwachen.
`UnauthorizedOperation`	`AccessDenied`	Dieser Fehler kann aus einer Vielzahl von Gründen auftreten. Der häufigste Grund ist, dass Sie versehentlich ein Tag entfernt haben, das der Service verwendet, um die Richtlinie für die mit dem Service verknüpfte Rolle von der Steuerebene herabzustufen. In diesem Fall kann Amazon EKS diese AWS Ressourcen nicht mehr verwalten und überwachen.
`InvalidSubnetID.NotFound`	`ResourceNotFound`	Dieser Fehler tritt auf, wenn die Subnetz-ID für die Eingangsregeln einer Sicherheitsgruppe nicht gefunden werden kann.

EC2 Amazon-Fehlercode	Code für Cluster-Integrität sprobleme	Description
`InvalidPermission.NotFound`	`ResourceNotFound`	Dieser Fehler tritt auf, wenn die Berechtigungen für die Eingangsregeln einer Sicherheitsgruppe nicht korrekt sind.
`InvalidGroup.NotFound`	`ResourceNotFound`	Dieser Fehler tritt auf, wenn die Gruppe der Eingangsregeln einer Sicherheitsgruppe nicht gefunden werden kann.
`InvalidNetworkInterfaceID.NotFound`	`ResourceNotFound`	Dieser Fehler tritt auf, wenn die Netzwerkschnittstellen-ID für die Eingangsregeln einer Sicherheitsgruppe nicht gefunden werden kann.
`InsufficientFreeAddressesInSubnet`	`ResourceLimitExceeded`	Dieser Fehler tritt auf, wenn das Kontingent der Subnetzressourcen überschritten wird.
`InsufficientCapacityOnOutpost`	`ResourceLimitExceeded`	Dieser Fehler tritt auf, wenn das Kapazitätskontingent des Außenpostens überschritten wird.
`NetworkInterfaceLimitExceeded`	`ResourceLimitExceeded`	Dieser Fehler tritt auf, wenn das Kontingent der elastischen Netzwerkschnittstelle überschritten wird.
`SecurityGroupLimitExceeded`	`ResourceLimitExceeded`	Dieser Fehler tritt auf, wenn das Kontingent der Sicherheitsgruppe überschritten wird.

EC2 Amazon-Fehlercode	Code für Cluster-Integrität sprobleme	Description
`VcpuLimitExceeded`	`ResourceLimitExceeded`	Dies wird beobachtet, wenn eine EC2 Amazon-Instance in einem neuen Konto erstellt wird. Die Fehlermeldung könnte ähnlich wie die folgende lauten: „`You have requested more vCPU capacity than your current vCPU limit of 32 allows for the instance bucket that the specified instance type belongs to. Please visit http://aws.amazon.com/contact-us/ec2-request to request an adjustment to this limit."`
`InvalidParameterValue`	`ConfigurationConflict`	Amazon EC2 gibt diesen Fehlercode zurück, wenn der angegebene Instance-Typ im Outpost nicht unterstützt wird.
Alle anderen Fehler	`InternalFailure`	Keine

Cluster können nicht erstellt oder geändert werden

Lokale Cluster erfordern andere Berechtigungen und Richtlinien als Amazon-EKS-Cluster, die in der Cloud gehostet werden. Wenn ein Cluster nicht erstellt werden kann und ein `InvalidPermissions` Fehler auftritt, überprüfen Sie, ob der Cluster-Rolle, die Sie verwenden, die von Amazon EKSLocal OutpostClusterPolicy verwaltete Richtlinie zugeordnet ist. Alle anderen API-Aufrufe erfordern dieselben Berechtigungen wie Amazon-EKS-Cluster in der Cloud.

Cluster bleibt im **`CREATING`**-Zustand hängen

Die Zeit, die zum Erstellen eines lokalen Clusters benötigt wird, hängt von mehreren Faktoren ab. Zu diesen Faktoren gehören Ihre Netzwerkkonfiguration, die Outpost-Konfiguration und die Konfiguration des Clusters. Im Allgemeinen wird ein lokaler Cluster erstellt und wechselt innerhalb von 15–20 Minuten in den `ACTIVE`-Status. Wenn ein lokaler Cluster im `CREATING`-Status bleibt, können Sie `describe-cluster`aufrufen, um Informationen über die Ursache im `cluster.health`-Ausgabefeld abzurufen.

Die am häufigsten auftretenden Probleme sind:

- Ihr Cluster kann von der AWS Region aus, in der sich Systems Manager befindet, keine Verbindung zur Kontrollebeneninstanz herstellen. Sie können dies überprüfen, indem Sie `aws ssm start-session --target` *`instance-id`* von einem Bastion-Host in der Region aufrufen. Wenn dieser Befehl nicht funktioniert, überprüfen Sie, ob Systems Manager auf der Instance der Steuerebene ausgeführt wird. Eine andere Möglichkeit besteht darin, den Cluster zu löschen und ihn dann neu zu erstellen.
- Die Steuerebenen-Instances können aufgrund von KMS-Schlüsselberechtigungen für EBS-Volumes nicht erstellt werden. Bei benutzerverwalteten KMS-Schlüsseln für verschlüsselte EBS-Volumes werden die Steuerebenen-Instances beendet, wenn auf den Schlüssel nicht zugegriffen werden kann. Wenn die Instanzen beendet werden, wechseln Sie entweder zu einem AWS verwalteten KMS-Schlüssel oder stellen Sie sicher, dass Ihre Richtlinie für benutzerverwaltete Schlüssel der Clusterrolle die erforderlichen Berechtigungen gewährt.
- Instances auf der Steuerebene von Systems Manager haben möglicherweise keinen Zugriff auf das Internet. Überprüfen Sie, ob das Subnetz, das Sie beim Erstellen des Clusters angegeben haben, über ein NAT-Gateway und eine VPC mit einem Internet-Gateway verfügt. Verwenden Sie VPC Reachability Analyzer, um zu überprüfen, ob die Instance der Steuerebene das Internet-Gateway erreichen kann. Weitere Informationen finden Sie unter Erste Schritte mit VPC Reachability Analyzer.
- Der von Ihnen angegebene Rollen-ARN enthält keine Richtlinien. Prüfen Sie, ob die AWS verwaltete Richtlinie: Amazon aus der Rolle entfernt EKSLocal OutpostClusterPolicy wurde. Dies kann auch passieren, wenn ein AWS CloudFormation Stack falsch konfiguriert ist.
- Alle bereitgestellten Subnetze müssen demselben Outpost zugeordnet sein und sich gegenseitig erreichen. Wenn beim Erstellen eines Clusters mehrere Subnetze angegeben werden, versucht Amazon EKS, die Instances der Steuerebene auf mehrere Subnetze zu verteilen.

- Amazon-EKS-verwaltete Sicherheitsgruppen werden auf die Elastic-Network-Schnittstelle angewendet. Andere Konfigurationselemente wie NACL-Firewall-Regeln könnten jedoch mit den Regeln für die elastische Netzwerkschnittstelle in Konflikt geraten.

VPC- und Subnetz-DNS-Konfiguration ist falsch konfiguriert oder fehlt

Lesen Sie den Artikel VPC und Subnetze für Amazon EKS-Cluster auf AWS Outposts erstellen.

Cluster bleibt im **`UPDATING`**-Zustand hängen

Amazon EKS aktualisiert automatisch alle vorhandenen lokalen Cluster auf die neuesten Plattformversionen für die entsprechende Kubernetes- Nebenversion. Weitere Informationen zu Plattformversionen finden Sie unter the section called “EKS-Plattformversionen”.

Während einer automatischen Einführung der Plattformversion ändert sich der Cluster-Status zu `UPDATING`. Der Aktualisierungsprozess besteht aus dem Austausch aller Kubernetes- Steuerebenen-Instances durch neue Instances, welche die neuesten Sicherheits-Patches und Fehlerkorrekturen enthalten, die für die jeweilige Kubernetes- Nebenversion veröffentlicht wurden. Im Allgemeinen dauert ein Aktualisierungsprozess der lokalen Cluster-Plattform weniger als 30 Minuten, und der Cluster kehrt anschließend in den `ACTIVE`-Status zurück Wenn ein lokaler Cluster über einen längeren Zeitraum in diesem `UPDATING`-Status verbleibt, können Sie `describe-cluster` aufrufen, um im `cluster.health`-Ausgabefeld nach Informationen zur Ursache zu suchen.

Amazon EKS stellt sicher, dass mindestens 2 von 3 Kubernetes-Steuerebenen-Instances fehlerfrei und betriebsbereit sind, um die lokale Cluster-Verfügbarkeit aufrechtzuerhalten und Service-Unterbrechungen zu vermeiden. Wenn ein lokaler Cluster im `UPDATING`-Status hängen bleibt, liegt dies normalerweise daran, dass ein Infrastruktur- oder Konfigurationsproblem vorliegt, das die Gewährleistung der Mindestverfügbarkeit von zwei Instances verhindert, falls der Prozess fortgesetzt wird. Daher wird der Aktualisierungsvorgang angehalten, um den lokalen Cluster-Service vor Unterbrechungen zu schützen.

Es ist wichtig, einen lokalen Cluster, der in einem `UPDATING`-Status hängen geblieben ist, zu überprüfen und die Ursache zu beheben. So kann der Aktualisierungsprozess abgeschlossen und der lokale Cluster mit der hohen Verfügbarkeit von drei Kubernetes-Steuerebenen-Instances zurück in `ACTIVE` wiederhergestellt werden.

Beenden Sie keine verwalteten lokalen `Kubernetes` EKS-Clusterinstanzen auf Outposts, sofern Sie nicht ausdrücklich vom AWS Support dazu aufgefordert werden. Dies ist besonders wichtig für lokale Cluster, die in einem bestimmten `UPDATING`-Status hängen geblieben sind, da eine hohe

Wahrscheinlichkeit besteht, dass ein anderer Steuerebenen-Knoten nicht vollständig fehlerfrei ist. Das Beenden der falschen Instances kann zu einer Service-Unterbrechung und dem Risiko eines Datenverlusts im lokalen Cluster führen.

Die am häufigsten auftretenden Probleme sind:

- Eine oder mehrere Steuerebenen-Instances können keine Verbindung zum System Manager herstellen, da sich die Netzwerkkonfiguration seit der Erstellung des lokalen Clusters geändert hat. Sie können dies überprüfen, indem Sie `aws ssm start-session --target instance-id` von einem Bastion-Host in der Region aufrufen. Wenn dieser Befehl nicht funktioniert, überprüfen Sie, ob Systems Manager auf der Instance der Steuerebene ausgeführt wird.
- Neue Steuerebenen-Instances können aufgrund von KMS-Schlüsselberechtigungen für EBS-Volumes nicht erstellt werden. Bei benutzerverwalteten KMS-Schlüsseln für verschlüsselte EBS-Volumes werden die Steuerebenen-Instances beendet, wenn auf den Schlüssel nicht zugegriffen werden kann. Wenn die Instanzen beendet werden, wechseln Sie entweder zu einem AWS verwalteten KMS-Schlüssel oder stellen Sie sicher, dass Ihre Richtlinie für benutzerverwaltete Schlüssel der Clusterrolle die erforderlichen Berechtigungen gewährt.
- Instances auf der Steuerebene von Systems Manager haben möglicherweise den Zugriff auf das Internet verloren. Überprüfen Sie, ob das Subnetz, das Sie beim Erstellen des Clusters angegeben haben, über ein NAT-Gateway und eine VPC mit einem Internet-Gateway verfügt. Verwenden Sie VPC Reachability Analyzer, um zu überprüfen, ob die Instance der Steuerebene das Internet-Gateway erreichen kann. Weitere Informationen finden Sie unter Erste Schritte mit VPC Reachability Analyzer. Wenn Ihre privaten Netzwerke über keine ausgehende Internetverbindung verfügen, stellen Sie sicher, dass alle erforderlichen VPC-Endpunkte und Gateway-Endpunkte weiterhin im regionalen Subnetz Ihres Clusters vorhanden sind (siehe the section called “Subnetz-Zugang zu AWS-Services”).
- Der von Ihnen angegebene Rollen-ARN enthält keine Richtlinien. Prüfen Sie, ob die AWS verwaltete Richtlinie: Amazon nicht aus der Rolle entfernt EKSLocal OutpostClusterPolicy wurde.
- Bei einer der neuen Kubernetes-Steuerebenen-Instances ist möglicherweise ein unerwarteter Bootstrapping-Fehler aufgetreten. Reichen Sie ein Ticket beim AWS Support Center ein, um weitere Anweisungen zur Fehlerbehebung und Protokollerfassung in diesem Ausnahmefall zu erhalten.

Knoten können keinem Cluster beitreten

- AMI-Probleme:

  - Sie verwenden ein inkompatibles AMI. Nur Amazon EKS-optimiertes Amazon Linux 2 AMIs wird unterstützt (`amazon-linux-2`,`amazon-linux-2-gpu`,`amazon-linux-2-arm64`). Wenn Sie versuchen, AL2 023 Knoten mit EKS LocalClusters auf AWS Outposts zu verbinden, können die Knoten dem Cluster nicht beitreten. Weitere Informationen finden Sie unter Amazon Linux-Knoten auf AWS Outposts erstellen.
  - Wenn Sie zum Erstellen Ihrer Knoten eine AWS CloudFormation Vorlage verwendet haben, stellen Sie sicher, dass sie kein nicht unterstütztes AMI verwendet.
- Fehlt der AWS IAM-Authenticator `ConfigMap` — Falls er fehlt, müssen Sie ihn erstellen. Weitere Informationen finden Sie unter the section called “Anwenden von `aws-auth`ConfigMap auf Ihren Cluster”.
- Die falsche Sicherheitsgruppe wird verwendet – Stellen Sie sicher, dass Sie `eks-cluster-sg-`*`cluster-name`*`-`*`uniqueid`* für die Sicherheitsgruppe Ihrer Worker-Knoten verwenden. Die ausgewählte Sicherheitsgruppe wird geändert AWS CloudFormation , sodass bei jeder Verwendung des Stacks eine neue Sicherheitsgruppe zulässig ist.
- Unerwartete VPC-Schritte für private Links – Falsche CA-Daten (`--b64-cluster-ca`) oder API-Endpunkt (`--apiserver-endpoint`) sind bestanden.

Sammeln von Protokollen

Wenn ein Outpost von der AWS Region getrennt wird, der er zugeordnet ist, funktioniert der Kubernetes-Cluster wahrscheinlich weiterhin normal. Wenn der Cluster jedoch nicht ordnungsgemäß funktioniert, folgen Sie den Schritten zur Fehlerbehebung unter Lokale Amazon EKS-Cluster auf AWS Outposts für Netzwerkunterbrechungen vorbereiten. Wenn Sie auf andere Probleme stoßen, wenden Sie sich an den AWS Support. AWS Der Support kann Sie beim Herunterladen und Ausführen eines Tools zur Protokollerfassung unterstützen. Auf diese Weise können Sie Protokolle von Ihren Kubernetes-Cluster-Steuerungsebeneninstanzen sammeln und sie zur weiteren Untersuchung an den AWS Support senden.

Instanzen der Kontrollebene sind über AWS Systems Manager nicht erreichbar

Wenn die Amazon EKS Control Plane Instances nicht über AWS Systems Manager (Systems Manager) erreichbar sind, zeigt Amazon EKS den folgenden Fehler für Ihren Cluster an.

```
Amazon EKS control plane instances are not reachable through SSM. Please verify your
 SSM and network configuration, and reference the EKS on Outposts troubleshooting
 documentation.
```

Um dieses Problem zu beheben, stellen Sie sicher, dass Ihre VPC und Subnetze die Anforderungen unter Erstellen einer VPC und Subnetze für Amazon EKS-Cluster auf AWS Outposts erfüllen und dass Sie die Schritte unter Setup Session Manager im Systems Manager Manager-Benutzerhandbuch ausgeführt haben. AWS

Amazon Linux-Knoten auf AWS Outposts erstellen

> **Important**
>
> Amazon EKS Local Clusters on Outposts unterstützt nur Knoten, die aus dem folgenden Amazon EKS-optimierten Amazon Linux 2023 erstellt wurden: AMIs
>
> - Standard Amazon Linux 2023 (`amazon-linux-2023/x86_64/standard`)
> - Beschleunigtes Nvidia Amazon Linux 2023 (`amazon-linux-2023/x86_64/nvidia`)
> - Beschleunigtes Neuron Amazon Linux 2023 () `amazon-linux-2023/x86_64/neuron`
>
> AWS Der Support für EKS AL2 — optimiert und AL2 beschleunigt — AMIs wurde mit Wirkung zum 26. November 2025 eingestellt. Sie können EKS zwar auch AL2 AMIs nach dem end-of-support (EOS-) Datum (26. November 2025) weiter verwenden, aber EKS wird nach diesem Datum keine neuen Kubernetes-Versionen oder Updates mehr veröffentlichen AL2 AMIs, einschließlich kleinerer Releases, Patches und Bugfixes. Weitere Informationen zu veralteten Versionen finden Sie hier. AL2

Dieses Thema beschreibt Hinweise zum Starten von Auto-Scaling-Gruppen von Amazon-Linux-Knoten auf einem Outpost, die mit Ihrem Amazon-EKS-Cluster registriert sind. Der Cluster kann sich in der AWS Cloud oder in einem Outpost befinden.

- Ein vorhandener Outpost. Weitere Informationen finden Sie unter Was sind AWS Outposts.
- Ein vorhandener Amazon-EKS-Cluster. Informationen zur Bereitstellung eines Clusters in der AWS Cloud finden Sie unterthe section called “Erstellen eines Clusters”. Informationen zum Bereitstellen eines Clusters in einem Outpost finden Sie unter the section called “Lokale Cluster ausführen”.
- Angenommen, Sie erstellen Ihre Knoten in einem Cluster in der AWS Cloud und Sie haben Subnetze in der AWS Region, in der Sie AWS Outposts, AWS Wavelength oder AWS Local Zones aktiviert haben. Diese Subnetze dürfen dann bei der Erstellung Ihres Clusters nicht angegeben

worden sein. Wenn Sie Ihre Knoten in einem Cluster in einem Outpost erstellen, müssen Sie bei der Erstellung Ihres Clusters ein Outpost-Subnetz angegeben haben.

- (Empfohlen für Cluster in der AWS Cloud) Das Amazon VPC CNI-Plug-In für Kubernetes ist mit einer eigenen IAM-Rolle konfiguriert, der die erforderliche IAM-Richtlinie zugeordnet ist. Weitere Informationen finden Sie unter the section called “Konfiguration für IRSA”. Lokale Cluster unterstützen keine IAM-Rollen für Service-Konten.

Sie können eine selbstverwaltete Amazon Linux-Knotengruppe mit `eksctl` oder AWS-Managementkonsole (mit einer AWS CloudFormation Vorlage) erstellen. Sie können auch Terraform verwenden.

Sie können eine selbstverwaltete Knotengruppe für einen lokalen Cluster mit den folgenden Tools erstellen, die auf dieser Seite beschrieben sind:

- the section called “`eksctl`”
- the section called “AWS-Managementkonsole”

⚠ Important

- Die selbstverwaltete Knotengruppe umfasst EC2 Amazon-Instances in Ihrem Konto. Diese Instances werden nicht automatisch aktualisiert, wenn Sie oder Amazon EKS die Version der Steuerebene in Ihrem Namen aktualisieren. Für eine selbstverwaltete Knotengruppe gibt es in der Konsole keinen Hinweis darauf, dass sie aktualisiert werden muss. Sie können die auf einem Knoten installierte `kubelet`-Version anzeigen, indem Sie den Knoten in der Liste Knoten auf der Registerkarte Übersicht Ihres Clusters auswählen, um zu bestimmen, welche Knoten aktualisiert werden müssen. Sie müssen die Knoten manuell aktualisieren. Weitere Informationen finden Sie unter the section called “Update-Methoden”.
- Die von kubelet auf Ihren selbstverwalteten Knoten verwendeten Zertifikate werden mit einer Gültigkeitsdauer von einem Jahr ausgestellt. Standardmäßig ist die Zertifikatsrotation nicht aktiviert (siehe: https://kubernetes). io/docs/reference/config-api/kubelet-config.v1beta1/# kubelet-config-k 8 s-io-v 1beta1-KubeletConfiguration), das heißt, wenn Sie einen selbstverwalteten Knoten haben, der länger als ein Jahr läuft, kann er sich nicht mehr bei der Kubernetes-API authentifizieren.
- Als bewährte Methode empfehlen wir Kunden, ihre selbstverwalteten Knotengruppen regelmäßig zu aktualisieren, um Sicherheitspatches von den neuesten für Amazon EKS

optimierten AMI zu erhalten CVEs . Die Aktualisierung der AMI, die in selbstverwalteten Knotengruppen verwendet wird, löst auch die Neuerstellung von Knoten aus und stellt sicher, dass sie nicht aufgrund abgelaufener Kubelet-Zertifikate in Konflikt geraten.

- Alternativ können Sie auch die Rotation von Client-Zertifikaten aktivieren (siehe: https://kubernetes). io/docs/tasks/tls/certificate-rotation/) beim Erstellen der selbstverwalteten Knotengruppen, um sicherzustellen, dass Kubelet-Zertifikate erneuert werden, wenn das aktuelle Zertifikat bald abläuft.

`eksctl`

Um selbstverwaltete Linux-Knoten zu starten mit **`eksctl`**

1. Installieren Sie die Version `0.215.0` oder höher des auf Ihrem Gerät installierten `eksctl` Befehlszeilentools oder AWS CloudShell. Informationen zum Installieren und Aktualisieren von `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.
2. Wenn sich Ihr Cluster in der AWS Cloud befindet und die von AmazonEKS_CNI_Policy verwaltete IAM-Richtlinie mit Ihrer Amazon EKS-Knoten-IAM-Rolle verknüpft ist, empfehlen wir, sie stattdessen einer IAM-Rolle zuzuweisen, die Sie dem Kubernetes-Dienstkonto zuordnen. `aws-node` Weitere Informationen finden Sie unter the section called “Konfiguration für IRSA”. Wenn sich Ihr Cluster in Ihrem Outpost befindet, muss die Richtlinie an Ihre Knotenrolle angehängt sein.
3. Der folgende Befehl erstellt eine Knotengruppe in einem bestehenden Cluster. Der Cluster muss mit `eksctl` erstellt worden sein. Ersetzen Sie *`al-nodes`* durch einen Namen für Ihre Knotengruppe. Der Name der Knotengruppe darf nicht länger als 63 Zeichen sein. Er muss mit einem Buchstaben oder einer Ziffer beginnen, kann danach aber auch Bindestriche und Unterstriche enthalten. Ersetzen Sie *`my-cluster`* mit dem Namen Ihres Clusters. Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der Region und des Kontos, in dem Sie den Cluster erstellen, eindeutig sein. AWS AWS Wenn Ihr Cluster auf einem Outpost existiert, ersetzen Sie *`id`* mit der ID eines Outpost-Subnetzes. Wenn Ihr Cluster in der AWS Cloud vorhanden ist, *`id`* ersetzen Sie ihn durch die ID eines Subnetzes, das Sie bei der Erstellung Ihres Clusters nicht angegeben haben. Ersetzen Sie die verbleibenden Beispielwerte durch Ihre eigenen Werte. Die Knoten werden standardmäßig mit derselben Kubernetes-Version wie die Steuerungsebene erstellt.

   Ersetzen Sie *`instance-type`* durch einen Instance-Typ, der auf Ihrem Outpost verfügbar ist.

*my-key*Ersetzen Sie es durch den Namen Ihres EC2 Amazon-Schlüsselpaars oder öffentlichen Schlüssels. Dieser Schlüssel wird für den SSH-Zugriff zu Ihren Knoten verwendet, nachdem diese gestartet wurden. Wenn Sie noch kein EC2 Amazon-Schlüsselpaar haben, können Sie eines in der erstellen AWS-Managementkonsole. Weitere Informationen finden Sie unter EC2 Amazon-Schlüsselpaare im EC2 Amazon-Benutzerhandbuch.

Erstellen Sie Ihre Knoten-Gruppe mit dem folgenden Befehl.

```
eksctl create nodegroup --cluster my-cluster --name al-nodes --node-type instance-
type \
    --nodes 3 --nodes-min 1 --nodes-max 4 --managed=false \
    --node-volume-type gp2 --subnet-ids subnet-id \
    --node-ami-family AmazonLinux2023
```

Wenn Ihr Cluster in der AWS Cloud bereitgestellt ist:

- Die von Ihnen bereitgestellte Knotengruppe kann `IPv4`-Adressen an Pods aus einem anderen CIDR-Block als dem der Instance zuweisen. Weitere Informationen finden Sie unter the section called “Benutzerdefinierte Netzwerke”.
- Die von Ihnen bereitgestellte Knotengruppe benötigt keinen ausgehenden Internetzugriff. Weitere Informationen finden Sie unter the section called “Private Cluster”.

Eine vollständige Liste aller verfügbaren Optionen und Standardwerte finden Sie unter Support für AWS Outposts in der `eksctl`-Dokumentation.

- Wenn Knoten dem Cluster nicht beitreten können, finden Sie weitere Informationen unter the section called “Knoten können nicht mit dem Cluster verknüpft werden” Beheben von Problemen mit Amazon EKS-Clustern und -Knoten und unter Problembehandlung the section called “Knoten können keinem Cluster beitreten” bei lokalen Amazon EKS-Clustern auf AWS Outposts.
- Eine Beispielausgabe sieht wie folgt aus. Mehrere Zeilen werden ausgegeben, während die Knoten erstellt werden. Die letzte Ausgabezeile ähnelt der folgenden Beispielzeile.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

4. (Optional) ??? Stellen Sie eine Beispielanwendung bereit, um Ihren Cluster und Ihre Linux-Worker-Knoten zu testen.

AWS-Managementkonsole

Schritt 1: Selbstverwaltete Linux-Knoten mit AWS-Managementkonsole starten

1. Laden Sie die neueste Version der AWS CloudFormation Vorlage herunter.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2025-11-24/
amazon-eks-outpost-nodegroup.yaml
```

2. Öffnen Sie die AWS CloudFormation -Konsole.
3. Wählen Sie Create stack (Stack erstellen) und dann With new resources (standard) (Mit neuen Ressourcen [Standard]) aus.
4. Wählen Sie für Specify template (Vorlage festlegen) Upload a template file (Vorlagendatei hochladen) aus und wählen Sie dann Choose file (Datei wählen). Wählen Sie die `amazon-eks-nodegroup.yaml`-Datei aus, die Sie in einem vorherigen Schritt heruntergeladen haben, und wählen Sie dann Next (Weiter) aus.
5. Geben Sie auf der Seite Specify stack details (Stack-Details angeben) die folgenden Parameter ein und klicken Sie dann auf Next (Weiter):
   - Stack-Name: Wählen Sie einen Stack-Namen für Ihren AWS CloudFormation Stack. Sie können ihn beispielsweise *`al-nodes`* nennen. Der Name darf nur alphanumerische Zeichen (wobei die Groß- und Kleinschreibung beachtet werden muss) und Bindestriche enthalten. Es muss mit einem alphanumerischen Zeichen beginnen und darf nicht länger als 100 Zeichen sein. Der Name muss innerhalb der AWS Region und des AWS Kontos, in dem Sie den Cluster erstellen, eindeutig sein.
   - ApiServerEndpoint: Geben Sie den Kubernetes API-Server-Endpunkt ein, der in der EKS-Konsole oder über DescribeCluster die API sichtbar ist.
   - ClusterName: Geben Sie den Namen Ihres Clusters ein. Wenn dieser Name nicht mit Ihrem Cluster-Namen übereinstimmt, können Ihre Knoten dem Cluster nicht beitreten.
   - ClusterId: Geben Sie die ID ein, die dem Cluster vom EKS-Dienst zugewiesen wurde. Sichtbar über DescribeCluster API. Wenn diese ID nicht mit Ihrer Cluster-ID übereinstimmt, können Ihre Knoten dem Cluster nicht beitreten.
   - CertificateAuthority: Geben Sie die Base64-kodierte Zeichenfolge der Kubernetes Certificate Authority ein. Sichtbar in der EKS-Konsole oder über die API. DescribeCluster
   - ServiceCidr: Geben Sie den Kubernetes Services CIDR ein. Sichtbar in der EKS-Konsole oder über die API. DescribeCluster

- ClusterControlPlaneSecurityGroup: Wählen Sie den SecurityGroupsWert aus der AWS CloudFormation Ausgabe aus, die Sie bei der Erstellung Ihrer VPC generiert haben.

  Die folgenden Schritte zeigen einen Vorgang zum Abrufen der entsprechenden Gruppe.

  i. Öffnen Sie die Amazon-EKS-Konsole.

  ii. Wählen Sie den Namen des Clusters.

  iii. Wählen Sie die Registerkarte Network (Network) aus.

  iv. Verwenden Sie den Wert Zusätzliche Sicherheitsgruppen als Referenz, wenn Sie aus der ClusterControlPlaneSecurityGroupDropdownliste auswählen.

- NodeGroupName: Geben Sie einen Namen für Ihre Knotengruppe ein. Dieser Name kann zu einem späteren Zeitpunkt zum Identifizieren der Auto-Scaling-Knotengruppe verwendet werden, die für Ihre Knoten erstellt wurde.
- NodeAutoScalingGroupMinSize: Geben Sie die Mindestanzahl von Knoten ein, auf die Ihre Auto Scaling Scaling-Gruppe für Knoten skalieren kann.
- NodeAutoScalingGroupDesiredCapacity: Geben Sie die gewünschte Anzahl von Knoten ein, auf die bei der Erstellung Ihres Stacks skaliert werden soll.
- NodeAutoScalingGroupMaxSize: Geben Sie die maximale Anzahl von Knoten ein, auf die Ihre Auto Scaling Scaling-Gruppe für Knoten skalieren kann.
- NodeInstanceType: Wählen Sie einen Instance-Typ für Ihre Knoten. Wenn Ihr Cluster in der AWS Cloud läuft, finden Sie weitere Informationen unterthe section called “Amazon EC2-Instance-Typen”. Wenn Ihr Cluster auf einem Outpost ausgeführt wird, können Sie nur einen Instance-Typ auswählen, der auf Ihrem Outpost verfügbar ist.
- NodeImageIdSSMParam: Vorab mit dem Amazon EC2 Systems Manager Manager-Parameter eines kürzlich für Amazon EKS optimierten AMI für eine variable Kubernetes-Version gefüllt. Um eine andere Kubernetes-Nebenversion zu verwenden, die von Amazon EKS unterstützt wird, ersetzen Sie *1.XX* durch eine andere unterstützte Version. Wir empfehlen, dieselbe Kubernetes-Version wie Ihr Cluster anzugeben.

  Um ein für Amazon EKS optimiertes beschleunigtes AMI zu verwenden, aktualisieren Sie den *NodeImageIdSSMParam* Wert auf den gewünschten SSM-Parameter. Erfahren Sie hier, wie Sie das EKS-AMI IDs von SSM abrufen können.

> **Note**
>
> Die Amazon EKS-Knoten AMIs basieren auf Amazon Linux. Sie können Sicherheits- oder Datenschutzereignisse für Amazon Linux im Amazon-Linux-Sicherheitscenter verfolgen, indem Sie die Registerkarte für die gewünschte Version auswählen. Sie können auch den entsprechenden RSS-Feed abonnieren. Sicherheits- oder Datenschutzereignisse enthalten eine Übersicht über das Problem, welche Pakete betroffen sind und wie Sie Ihre Instances aktualisieren, um das Problem zu beheben.

- NodeImageId: (Optional) Wenn Sie Ihr eigenes benutzerdefiniertes AMI (anstelle eines für Amazon EKS optimierten AMI) verwenden, geben Sie eine Knoten-AMI-ID für Ihre AWS Region ein. Wenn Sie hier einen Wert angeben, überschreibt dieser alle Werte im NodeImageIdSSMParamFeld.
- NodeVolumeSize: Geben Sie eine Root-Volume-Größe für Ihre Knoten in GiB an.
- NodeVolumeType: Geben Sie einen Root-Volume-Typ für Ihre Knoten an.
- KeyName: Geben Sie den Namen eines Amazon EC2 SSH-Schlüsselpaars ein, mit dem Sie nach dem Start über SSH eine Verbindung zu Ihren Knoten herstellen können. Wenn Sie noch kein EC2 Amazon-Schlüsselpaar haben, können Sie eines in der erstellen AWS-Managementkonsole. Weitere Informationen finden Sie unter EC2 Amazon-Schlüsselpaare im EC2 Amazon-Benutzerhandbuch.

> **Note**
>
> Wenn Sie hier kein key pair angeben, schlägt die AWS CloudFormation Stack-Erstellung fehl.

- Deaktivieren IMDSv1: Standardmäßig unterstützt jeder Knoten den Instanz-Metadatendienst Version 1 (IMDSv1) und IMDSv2. Sie können deaktivieren IMDSv1. Um zu verhindern, dass future Knoten und Pods in der Knotengruppe verwendet werden IMDSv1, setzen Sie Disable IMDSv1 auf true. Weitere Informationen finden Sie unter Konfiguration des Instance-Metadatenservice. Weitere Informationen zum Einschränken des Zugriffs darauf auf Ihre Knoten finden Sie unter Beschränken Sie den Zugriff auf das Instance-Profil, das dem Worker-Knoten zugewiesen ist.
- VpcId: Geben Sie die ID für die VPC ein, die Sie erstellt haben. Überprüfen Sie vor der Auswahl eines VPC die VPC-Anforderungen und -Überlegungen.

- Subnets (Subnetze): Wenn sich Ihr Cluster auf einem Outpost befindet, wählen Sie mindestens ein privates Subnetz in Ihrer VPC. Bevor Sie Subnetze auswählen, überprüfen Sie die Anforderungen und Überlegungen zu Subnetzen. Sie können sehen, welche Subnetze privat sind, indem Sie den jeweiligen Subnetzlink in der Registerkarte Networking (Netzwerk) Ihres Clusters öffnen.

6. Treffen Sie die gewünschte Auswahl auf der Seite Configure stack options (Stackoptionen konfigurieren) und wählen Sie dann Next (Weiter) aus.
7. Aktivieren Sie das Kontrollkästchen links neben Ich bestätige, dass AWS CloudFormation möglicherweise IAM-Ressourcen erstellt werden. , und wählen Sie dann Stapel erstellen aus.
8. Wenn Ihr Stack fertig erstellt wurde, wählen Sie ihn in der Konsole aus und klicken Sie auf Outputs (Ausgaben).
9. Notieren Sie das NodeInstanceRolefür die Knotengruppe, die erstellt wurde. Sie benötigen diese, wenn Sie Ihre Amazon-EKS--Arbeitsknoten konfigurieren.

Schritt 2: Knoten, die Ihrem Cluster beitreten sollen, aktivieren

1. Überprüfen Sie, ob Sie bereits über eine `aws-auth` `ConfigMap` verfügen.

```
kubectl describe configmap -n kube-system aws-auth
```

2. Wenn eine `aws-auth` `ConfigMap` angezeigt wird, aktualisieren Sie sie nach Bedarf.

   a. Öffnen Sie `ConfigMap` zum Bearbeiten.

   ```
   kubectl edit -n kube-system configmap/aws-auth
   ```

   b. Fügen Sie nach Bedarf einen neuen `mapRoles`-Eintrag hinzu. Setzen Sie den `rolearn` Wert auf den NodeInstanceRoleWert, den Sie im vorherigen Verfahren aufgezeichnet haben.

   ```
   [...]
   data:
     mapRoles: |
       - rolearn: <ARN of instance role (not instance profile)>
         username: system:node:{{EC2PrivateDNSName}}
         groups:
           - system:bootstrappers
           - system:nodes
   [...]
   ```

c. Speichern Sie die Datei und beenden Sie den Text-Editor.

3. Wenn die Fehlermeldung `Error from server (NotFound): configmaps "aws-auth" not found` angezeigt wird, wenden Sie die standardmäßige `ConfigMap` an.

   a. Laden Sie die Konfigurationszuordnung herunter.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/aws-auth-cm.yaml
   ```

   b. Stellen Sie in der `aws-auth-cm.yaml` Datei `rolearn` den NodeInstanceRoleWert ein, den Sie im vorherigen Verfahren aufgezeichnet haben. Hierzu können Sie einen Texteditor verwenden oder *`my-node-instance-role`* ersetzen und den folgenden Befehl ausführen:

   ```
   sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-instance-role|' aws-auth-cm.yaml
   ```

   c. Wenden Sie die Konfiguration an. Die Ausführung dieses Befehls kann einige Minuten dauern.

   ```
   kubectl apply -f aws-auth-cm.yaml
   ```

4. Sehen Sie sich den Status Ihrer Knoten an und warten Sie, bis diese in den `Ready`-Status eintreten.

   ```
   kubectl get nodes --watch
   ```

   Geben Sie `Ctrl+C` ein, um zum Shell-Prompt zurückzukehren.

   > **Note**
   >
   > Wenn Sie Autorisierungs- oder Ressourcenfehler erhalten, finden Sie weitere Informationen unter the section called “Nicht autorisiert oder Zugriff verweigert (`kubectl`)” im Thema zur Fehlerbehebung.

   Wenn Knoten dem Cluster nicht beitreten können, finden Sie weitere Informationen unter the section called “Knoten können nicht mit dem Cluster verknüpft werden” Beheben von Problemen mit Amazon EKS-Clustern und -Knoten und unter Problembehandlung the section called “Knoten können keinem Cluster beitreten” bei lokalen Amazon EKS-Clustern auf AWS Outposts.

5. Installieren Sie den Amazon-EBS-CSI-Treiber. Weitere Informationen finden Sie unter Installation am GitHub. Stellen Sie im Abschnitt Einrichten der Treiberberechtigung sicher, dass Sie die Anweisungen für die Option Verwenden des IAM-Instance-Profils befolgen. Sie müssen die gp2-Speicherklasse verwenden. Die gp3-Speicherklasse wird nicht unterstützt.

   Führen Sie die folgenden Schritte aus, um eine gp2-Speicherklasse auf Ihrem Cluster zu erstellen.

   a. Führen Sie den folgenden Befehl aus, um die Datei `gp2-storage-class.yaml` zu erstellen.

```
cat >gp2-storage-class.yaml <<EOF
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  annotations:
    storageclass.kubernetes.io/is-default-class: "true"
  name: ebs-sc
provisioner: ebs.csi.aws.com
volumeBindingMode: WaitForFirstConsumer
parameters:
  type: gp2
  encrypted: "true"
allowVolumeExpansion: true
EOF
```

   b. Wenden Sie das Manifest auf Ihren Cluster an.

```
kubectl apply -f gp2-storage-class.yaml
```

6. (Nur GPU-Knoten) Wenn Sie einen GPU-Instance-Typ und ein für Amazon EKS optimiertes beschleunigtes AMI ausgewählt haben, müssen Sie das NVIDIA-Geräte-Plug-In für Kubernetes als DaemonSet auf Ihrem Cluster anwenden. Ersetzen Sie es *vX.X.X* durch die gewünschte s-device-pluginNVIDIA/K8-Version, bevor Sie den folgenden Befehl ausführen.

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/
deployments/static/nvidia-device-plugin.yml
```

Schritt 3: Zusätzliche Maßnahmen

1. (Optional) ??? Stellen Sie eine Beispielanwendung bereit, um Ihren Cluster und Ihre Linux-Worker-Knoten zu testen.

2. Wenn Ihr Cluster auf einem Outpost bereitgestellt wird, überspringen Sie diesen Schritt. Wenn Ihr Cluster in der AWS Cloud bereitgestellt wird, sind die folgenden Informationen optional. (Optional) Wenn die verwaltete IAM-Richtlinie AmazonEKS_CNI_Policy Ihrer IAM-Rolle des Amazon-EKS-Knoten zugeordnet ist, empfehlen wir, sie stattdessen einer IAM-Rolle zuzuweisen, die Sie dem Kubernetes-`aws-node`-Servicekonto zuordnen. Weitere Informationen finden Sie unter the section called “Konfiguration für IRSA”.

Überblick über künstliche Intelligenz (KI) und Machine Learning (ML) in Amazon EKS

Amazon Elastic Kubernetes Service (EKS) ist eine verwaltete Kubernetes-Plattform, die es Unternehmen ermöglicht, KI- und Machine-Learning-Workloads (ML) mit einzigartiger Flexibilität und Kontrolle bereitzustellen, zu verwalten und zu skalieren. EKS basiert auf dem Open-Source-Kubernetes-Ökosystem und ermöglicht es Ihnen, Ihr vorhandenes Kubernetes-Fachwissen zu nutzen und sich gleichzeitig nahtlos in Open-Source-Tools und -Services zu integrieren. AWS

Ganz gleich, ob Sie groß angelegte Modelle trainieren, Online-Inferenzen in Echtzeit ausführen oder generative KI-Anwendungen einsetzen, EKS bietet die Leistung, Skalierbarkeit und Kosteneffizienz, die Ihre Projekte erfordern. AI/ML

Warum sollten Sie sich für EKS im Bereich KI/ML entscheiden?

EKS ist eine verwaltete Kubernetes-Plattform, mit der Sie komplexe Workloads bereitstellen und verwalten können. AI/ML Sie basiert auf dem Open-Source-Kubernetes-Ökosystem, lässt sich in AWS Dienste integrieren und bietet so die Kontrolle und Skalierbarkeit, die für fortgeschrittene Projekte erforderlich sind. Für Teams, die mit AI/ML Implementierungen noch nicht vertraut sind, werden vorhandene Kubernetes-Fähigkeiten direkt übertragen, was eine effiziente Orchestrierung mehrerer Workloads ermöglicht.

EKS unterstützt alles von Betriebssystemanpassungen bis hin zur Skalierung der Rechenleistung, und seine Open-Source-Grundlage fördert die technologische Flexibilität und bewahrt die Auswahlmöglichkeiten für zukünftige Infrastrukturentscheidungen. Die Plattform bietet die Leistungs- und Optimierungsoptionen, die AI/ML Workloads benötigen, und unterstützt Funktionen wie:

- Vollständige Cluster-Kontrolle zur Feinabstimmung von Kosten und Konfigurationen ohne versteckte Abstraktionen.
- Latenzzeiten von weniger als einer Sekunde für Echtzeit-Inferenz-Workloads in der Produktion
- Erweiterte Anpassungen wie Multi-Instance- und Multi-Cloud-Strategien GPUs und Tuning auf Betriebssystemebene
- Möglichkeit, Workloads mithilfe von EKS als einheitlichem Orchestrator für alle Pipelines zu zentralisieren AI/ML

Wichtige Anwendungsfälle

Amazon EKS bietet eine robuste Plattform für eine Vielzahl von AI/ML Workloads und unterstützt verschiedene Technologien und Bereitstellungsmuster:

- (Online-) Inferenz in Echtzeit: EKS ermöglicht mithilfe von Tools wie Triton Inference Server und Amazon EC2 Inf1- und Inf2-Instances sofortige Vorhersagen zu eingehenden Daten TorchServe, z. B. zur Betrugserkennung, mit einer Latenz von weniger als KServeeiner Sekunde. Diese Workloads profitieren von der dynamischen Skalierung mit Karpenter und KEDA und nutzen gleichzeitig Amazon EFS für das Modell-Sharding über Pods hinweg. Amazon ECR Pull Through Cache (PTC) beschleunigt Modellaktualisierungen und Bottlerocket-Daten-Volumes mit Amazon EBS-optimierten Volumes gewährleisten schnellen Datenzugriff.
- Allgemeines Modelltraining: Organizations nutzen EKS, um komplexe Modelle mit großen Datensätzen über längere Zeiträume zu trainieren, indem sie Kubeflow Training Operator, Ray Serve und Torch Distributed Elastic auf Amazon EC2 P4d- und Amazon Trn1-Instances verwenden. EC2 Diese Workloads werden durch Batch-Planung mit Tools wie Volcano, Yunikorn und Kueue unterstützt. Amazon EFS ermöglicht die gemeinsame Nutzung von Modellprüfpunkten, und Amazon S3 verwaltet Modell import/export mit Lebenszyklusrichtlinien für die Versionsverwaltung.
- Retrieval Augmented Generation (RAG)-Pipelines: EKS verwaltet Kundensupport-Chatbots und ähnliche Anwendungen durch die Integration von Abruf- und Generierungsprozessen. Diese Workloads verwenden häufig Tools wie Argo Workflows und Kubeflow für die Orchestrierung, Vektordatenbanken wie Pinecone, Weaviate oder Amazon und stellen Anwendungen Benutzern über den Application Load OpenSearch Balancer Controller (LBC) zur Verfügung. NVIDIA NIM optimiert die GPU-Auslastung, während Prometheus und Grafana die Ressourcennutzung überwachen.
- Einsatz generativer KI-Modelle: Unternehmen setzen mithilfe von Ray Serve, vLLM und Triton Inference Server auf Amazon EC2 G5 - und Inferentia-Beschleunigern Dienste zur Erstellung von Inhalten in Echtzeit auf EKS ein, z. B. Text- oder Bildgenerierung. Diese Implementierungen optimieren die Leistung und die Speichernutzung für groß angelegte Modelle. JupyterHubermöglicht iterative Entwicklung, Gradio bietet einfache Weboberflächen und der S3 Mountpoint CSI-Treiber ermöglicht das Mounten von S3-Buckets als Dateisysteme für den Zugriff auf große Modelldateien.
- Batch-Inferenz (offline): Unternehmen verarbeiten große Datensätze effizient durch geplante Aufträge AWS Batch oder Volcano. Diese Workloads verwenden häufig Inf1- und EC2 Inf2-Instances für AWS Inferentia-Chips, Amazon EC2 G4dn-Instances für NVIDIA T4 oder c5- und

c6i-CPU-Instances GPUs, wodurch die Ressourcennutzung außerhalb der Spitzenzeiten für Analyseaufgaben maximiert wird. Das AWS Neuron SDK und die NVIDIA-GPU-Treiber optimieren die Leistung und ermöglichen gleichzeitig die gemeinsame Nutzung von GPUs. MIG/TS Zu den Speicherlösungen gehören Amazon S3 und Amazon EFS sowie FSx für Lustre mit CSI-Treibern für verschiedene Speicherklassen. Das Modellmanagement nutzt Tools wie Kubeflow Pipelines, Argo Workflows und Ray Cluster, während die Überwachung durch Prometheus, Grafana und benutzerdefinierte Modell-Überwachungstools erfolgt.

Fallstudien

Kunden entscheiden sich aus verschiedenen Gründen für Amazon EKS, beispielsweise zur Optimierung der GPU-Nutzung oder zur Ausführung von Echtzeit-Inferenz-Workloads mit einer Latenz von weniger als einer Sekunde, wie die folgenden Fallstudien zeigen. Eine Liste aller Fallstudien zu Amazon EKS finden Sie unter AWS -Kundenerfolgsgeschichten.

- Unitary verarbeitet täglich 26 Millionen Videos mithilfe von KI zur Inhaltsmoderation, was eine hohe Durchsatzleistung und geringe Latenz erfordert. Das Unternehmen hat die Startzeiten von Containern um 80 % reduziert und gewährleistet so eine schnelle Reaktion auf Skalierungsereignisse bei schwankendem Datenverkehr.
- Miro, die visuelle Kollaborationsplattform mit 70 Millionen Nutzern weltweit, vermeldete eine Reduzierung der Rechenkosten um 80 % im Vergleich zu ihren zuvor selbst verwalteten Kubernetes-Clustern.
- Synthesia, ein Unternehmen, das generative KI-Videoproduktion als Service anbietet, mit dem Kunden anhand von Textvorgaben realistische Videos erstellen können, hat eine 30-fache Verbesserung des Durchsatzes beim ML-Modelltraining erzielt.
- Harri, ein Anbieter von HR-Technologie für die Hotellerie, konnte durch die Migration auf AWS -Graviton-Prozessoren eine um 90 % schnellere Skalierung als Reaktion auf Nachfragespitzen erreichen und seine Rechenkosten um 30 % senken.
- Ada Support, ein Unternehmen für KI-gestützte Kundenservice-Automatisierung, erzielte eine Senkung der Rechenkosten um 15 % bei einer Steigerung der Recheneffizienz um 30 %.
- Snorkel AI, das Unternehmen bei der Erstellung und Anpassung von Basismodellen und großen Sprachmodellen unterstützt, erzielte durch die Implementierung intelligenter Skalierungsmechanismen für seine GPU-Ressourcen Kosteneinsparungen von über 40 %.

Einsatz von Machine Learning in EKS

Um mit der Planung und Nutzung von Plattformen und Workloads für Machine Learning auf EKS in der AWS Cloud zu beginnen, fahren Sie mit dem the section called “Ressourcen” Abschnitt fort.

Ausführung von Echtzeit-Online-Inferenz-Workloads in Amazon EKS

> **Tip**
>
> Melden Sie sich für bevorstehende Amazon AI/ML EKS-Workshops an.

Dieser Abschnitt soll Ihnen dabei helfen, Echtzeit-Online-Inferenz-Workloads in Amazon Elastic Kubernetes Service (EKS) bereitzustellen und zu betreiben. Sie finden Anleitungen zum Aufbau optimierter Cluster mit GPU-beschleunigten Knoten, zur Integration von AWS Services für Speicher und automatische Skalierung, zur Bereitstellung von Beispielmodellen zur Validierung und zu wichtigen architektonischen Überlegungen wie der Entkopplung von CPU- und GPU-Aufgaben, der Auswahl geeigneter AMIs Instance-Typen und der Sicherstellung der Verfügbarkeit von Inferenzendpunkten mit geringer Latenz.

Themen

- Leitfaden für bewährte Methoden zur Cluster-Einrichtung für Echtzeit-Inferenz in Amazon EKS
- Schnellstart: LLM-Inferenz mit hohem Durchsatz mit vLLM auf Amazon EKS

Leitfaden für bewährte Methoden zur Cluster-Einrichtung für Echtzeit-Inferenz in Amazon EKS

> **Tip**
>
> Melden Sie sich für bevorstehende Amazon AI/ML EKS-Workshops an.

Einführung

Dieser Leitfaden bietet eine praktische Anleitung zur Einrichtung eines Amazon Elastic Kubernetes Service (EKS) -Clusters, der für Online-Inferenz-Workloads in Echtzeit optimiert ist, und beinhaltet durchweg von Experten kuratierte Best Practices. AWS Es verwendet eine eigenwillige EKS-Schnellstart-Architektur — eine kuratierte Zusammenstellung von Treibern, Instance-Typen und Konfigurationen, die auf Best Practices für Modelle, Beschleuniger und Skalierung abgestimmt sind. AWS Dieser Ansatz erspart Ihnen die Auswahl der Cluster-Einstellungen, sodass Sie schnell einen funktionsfähigen, vorkonfigurierten Cluster in Betrieb nehmen können. Unterwegs stellen wir Beispiel-Workloads bereit, um Ihr Setup zu validieren, erklären wichtige Architekturkonzepte (wie die Entkopplung CPU-gebundener Aufgaben von GPU-intensiven Berechnungen) und gehen auf häufig gestellte Fragen ein (z. B. warum sollten Sie Bottlerocket AMI anstelle von 023 wählen?) AL2 und skizzieren die nächsten Schritte zur Erweiterung der Funktionen Ihres Clusters.

Dieser Leitfaden wurde speziell für Ingenieure, Plattformadministratoren, Betreiber und data/AI Spezialisten für Machine Learning (ML) und künstliche Intelligenz (KI) entwickelt, die neu im EKS-Ökosystem sind. Er setzt Vertrautheit mit Kubernetes voraus, jedoch keine EKS-Vorkenntnisse. AWS Es soll Ihnen dabei helfen, die erforderlichen Schritte und Prozesse zu verstehen, um Echtzeit-Online-Inferenz-Workloads zum Laufen zu bringen. Der Leitfaden zeigt Ihnen die Grundlagen der Erstellung eines Inferenzclusters mit einem Knoten, einschließlich der Bereitstellung von GPU-Ressourcen, der Integration von Speicher für Modellartefakte, der Bereitstellung eines sicheren AWS Servicezugriffs und der Offenlegung von Inferenzendpunkten. Dabei wird durchgehend Wert auf ein latenzarme, ausfallsicheres Design für benutzerorientierte Anwendungen wie Betrugserkennung, Echtzeit-Chatbots und Stimmungsanalyse in Kundenfeedback-Systemen gelegt.

In diesem Leitfaden konzentrieren wir uns ausschließlich auf die Einrichtung eines grundlegenden, präskriptiven Ausgangspunkts mithilfe von G5-Instances. EC2 Wenn Sie nach AWS Inferentia-spezifischen Cluster-Konfigurationen oder end-to-end Workflows suchen, finden Sie unsere Workshops unter. the section called “Inferenz-Clustern mit Inferentia einrichten” the section called “Ressourcen”

Bevor Sie beginnen

Stellen Sie zunächst sicher, dass Sie die folgenden Aufgaben durchgeführt haben:

- Einrichtung Ihrer Umgebung für Amazon EKS
- Installation der neuesten Version von eksctl
- Installation von Helm

- (Optional) Installation von Docker
- (Optional) Installation der NGC-CLI

Architektur

Echtzeit-Online-Inferenz bezeichnet den Prozess, bei dem ein trainiertes Machine-Learning-Modell verwendet wird, um mit minimaler Latenz Prognosen oder Ergebnisse zu eingehenden Datenströmen zu generieren. Dies ermöglicht beispielsweise die Echtzeit-Betrugserkennung, die Klassifizierung von Images oder die Erstellung von Wissensdiagrammen als Reaktion auf Benutzereingaben. Die Architektur eines Echtzeit-Online-Inferenzsystems ermöglicht Machine Learning-Prognosen mit niedriger Latenz in benutzerorientierten Anwendungen, indem sie die CPU-gebundene Verarbeitung des Web-Datenverkehrs von den GPU-intensiven KI-Berechnungen trennt. Dieser Prozess findet in der Regel innerhalb einer größeren Anwendungsumgebung statt und beinhaltet oft Backend-, Frontend-, Vektor- und Modell-Services. Der Fokus liegt dabei auf spezialisierten Komponenten, um eine unabhängige Skalierung, parallele Entwicklung und Ausfallsicherheit zu ermöglichen. Die Isolierung von Inferenzaufgaben auf dedizierter GPU-Hardware und die Nutzung von Schnittstellen wie z. B. WebSockets eine hohe Parallelität, eine schnelle Verarbeitung von Modellen wie Transformatoren APIs und Benutzerinteraktionen über das Frontend werden gewährleistet. Beachten Sie, dass Vektordatenbanken und Retrieval Augmented Generation (RAG)-Pipelines zwar häufig eine wichtige Rolle in Echtzeit-Inferenzsystemen spielen, diese Komponenten jedoch nicht in diesem Leitfaden behandelt werden. Eine typische Architektur umfasst in der Regel mindestens folgende Komponenten:

- Frontend-Dienst: Dient als benutzerorientierte Schnittstelle, verwaltet die clientseitige Logik, rendert dynamische Inhalte und erleichtert Interaktionen in Echtzeit. Er kommuniziert mit dem Backend-Service, um Inferenzanfragen zu initiieren und Ergebnisse anzuzeigen, wobei häufig Anfragen an den Backend-Service initiiert werden, der WebSockets für Streaming-Updates oder APIs für den strukturierten Datenaustausch verwendet wird. Für diesen Dienst ist in der Regel kein dedizierter Load Balancer erforderlich, da er in Content Delivery Networks (CDNs) gehostet werden kann, z. B. AWS CloudFront für statische Ressourcen, oder direkt von Webservern aus bereitgestellt werden kann, wobei die Skalierung bei Bedarf für dynamische Inhalte über Gruppen mit auto-scaling erfolgt.
- Backend-Service: Fungiert als Orchestrator der Anwendung und verwaltet die Geschäftslogik wie Benutzerauthentifizierung, Datenvalidierung und Dienstkoordination (z. B. über APIs für RESTful Endpunkte oder WebSockets für persistente Verbindungen). Es kommuniziert mit dem Inferenzdienst, skaliert unabhängig auf Multi-Core CPUs und RAM, um hohen Webverkehr zu

bewältigen, ohne darauf angewiesen zu sein GPUs, und benötigt häufig einen Load Balancer (wie AWS Application Load Balancer oder Network Load Balancer), um eingehende Anfragen auf mehrere Instanzen zu verteilen, insbesondere in Szenarien mit hoher Parallelität. Ein Ingress-Controller kann externe Zugriffs- und Routing-Regeln für verbesserte Sicherheit und Datenverkehrsgestaltung verwalten.

- Inference Service: Dient als Kern für KI-Berechnungen und läuft GPUs mit ausreichend VRAM (z. B. 8-12 GB für Modelle wie DistilBert), um Vektoreinbettungen, Wissensextraktion und Modellinferenz (z. B. APIs für Batch-Anfragen oder WebSockets für Echtzeit-Streaming) mithilfe von benutzerdefinierten oder Open-Source-Modellen durchzuführen. Diese Isolierung verhindert Abhängigkeitskonflikte, ermöglicht Modellaktualisierungen ohne Ausfallzeiten und ermöglicht eine horizontale Skalierung mit Lastausgleich für mehrere gleichzeitige Anfragen. Um den Modelldienst effektiv verfügbar zu machen, befindet er sich in der Regel hinter einem Load Balancer, um GPU-gebundene Workloads auf replizierte Instanzen zu verteilen, während eine Eingangsressource oder ein Controller (z. B. der ALB Ingress Controller in AWS) das externe Routing, die SSL-Terminierung und die pfadbasierte Weiterleitung übernimmt, um einen sicheren und effizienten Zugriff zu gewährleisten, ohne den Einzelnen zu überfordern GPUs.

Übersicht über die Lösung

Echtzeit-Online-Inferenzsysteme erfordern eine leistungsstarke, ausfallsichere Architektur, die extrem niedrige Latenzzeiten bietet und gleichzeitig unvorhersehbare, hohe Datenverkehrsausfälle bewältigen kann. In dieser Lösungsübersicht wird erklärt, wie die folgenden AWS Komponenten in dem Amazon EKS-Cluster zusammenarbeiten, den wir erstellen werden, um sicherzustellen, dass unser Cluster Modelle für maschinelles Lernen hosten und verwalten kann, die sofortige Vorhersagen zu Live-Daten mit minimaler Verzögerung für Endbenutzer liefern.

- Amazon EC2 G5-Instances — Für GPU-intensive Inferenzaufgaben verwenden wir die EC2 Instance-Typen g5.xlarge und g5.2xlarge G5, die über eine einzelne (1) NVIDIA A10G-GPU mit 24 GB Arbeitsspeicher verfügen (z. B. 8 Milliarden Parameter bei). FP16 Diese basieren auf der NVIDIA Ampere-Architektur und GPUs werden von NVIDIA A10G Tensor Core GPUs - und AMD EPYC-Prozessoren der 2. Generation angetrieben. Sie unterstützen 4-8 VCPUs, bis zu 10 Gbit/s Netzwerkbandbreite und 250-450 GB lokalen NVMe SSD-Speicher, was eine schnelle Datenbewegung und Rechenleistung für komplexe Modelle gewährleistet und sie somit ideal für Inferenzaufgaben mit niedriger Latenz und hohem Durchsatz macht. Die Auswahl eines EC2 Instance-Typs ist anwendungsspezifisch und hängt von Ihrem Modell (z. B. Bild-, Video-, Textmodell) und Ihren Latenz- und Durchsatzanforderungen ab. Wenn Sie beispielsweise ein Bild- und/oder Videomodell verwenden, möchten Sie möglicherweise EC2 P5-Instances verwenden, um

eine optimale Latenz in Echtzeit zu erzielen. Wir empfehlen, mit EC2 G5-Instances zu beginnen, da dies einen guten Ausgangspunkt für eine schnelle Inbetriebnahme bietet. Anschließend können Sie anhand von Performance-Benchmark-Tests beurteilen, ob sie für Ihre Workloads geeignet sind. Für fortgeschrittenere Fälle sollten Sie EC2 G6-Instances in Betracht ziehen.

- Amazon EC2 M7g-Instances — Für CPU-intensive Aufgaben wie Datenvorverarbeitung, Bearbeitung von API-Anfragen, Hosten des Karpenter Controllers, Add-Ons und anderer Systemkomponenten verwenden wir den Instance-Typ m5.xlarge M7g. EC2 M7g-Instances sind ARM-basierte Instances, die über 4 V, 16 GB Arbeitsspeicher und bis zu 12,5 Gbit/s Netzwerkbandbreite verfügen CPUs und von Graviton3-Prozessoren angetrieben werden. AWS Die Auswahl eines EC2 Instance-Typs ist anwendungsspezifisch und hängt von den Rechen-, Speicher- und Skalierbarkeitsanforderungen Ihres Workloads ab. Für rechenoptimierte Workloads könnten Sie EC2 C7g-Instances in Betracht ziehen, die ebenfalls Graviton3-Prozessoren verwenden, aber für bestimmte Anwendungsfälle für eine höhere Rechenleistung als M7g-Instances optimiert sind. Alternativ bieten neuere EC2 C8g-Instances (sofern verfügbar) eine bis zu 30% bessere Rechenleistung als C7g-Instances. Wir empfehlen, mit EC2 M7g-Instances zu beginnen, da sie kosteneffizient sind und mit einer Vielzahl von Workloads kompatibel sind (z. B. Anwendungsserver, Microservices, Gaming-Server, Datenspeicher mittlerer Größe). Anschließend sollten Sie anhand von Leistungsbenchmarktests prüfen, ob sie für Ihre Workloads geeignet sind.
- Mountpoint-CSI-Treiber für Amazon S3 – Für Workloads auf Einzel-GPU-Instances, bei denen mehrere Pods eine GPU gemeinsam nutzen (z. B. mehrere Pods, die auf demselben Knoten geplant sind, um dessen GPU-Ressourcen zu nutzen), verwenden wir den Mountpoint-S3-CSI-Treiber, um die Speichernutzung zu optimieren. Dies ist für Aufgaben wie die Inferenz großer Modelle in kostenbewussten, wenig komplexen Einrichtungen von entscheidender Bedeutung. Er stellt Amazon-S3-Buckets als POSIX-ähnliches Dateisystem zur Verfügung, das für den Kubernetes-Cluster verfügbar ist. Dadurch können Inferenz-Pods Modellartefakte (z. B. Modellgewichte) direkt in den Speicher lesen, ohne sie zuvor herunterladen zu müssen, und Datensätze mithilfe von Standard-Dateioperationen eingeben. Darüber hinaus verfügt S3 über praktisch unbegrenzte Speicherkapazität und beschleunigt datenintensive Inferenz-Workloads. Die Auswahl eines Speicher-CSI-Treibers ist anwendungsspezifisch und hängt von den Anforderungen Ihrer Workload hinsichtlich Durchsatz und Latenz ab. Obwohl der FSx für OpenZFS CSI Driver eine Latenz von unter einer Millisekunde für zufällige I/O oder vollständig POSIX-konforme, knotenübergreifende gemeinsame persistente Volumes bietet, empfehlen wir, mit dem Mountpoint S3 CSI-Treiber zu beginnen, da er Skalierbarkeit, geringere Kosten für große Datensätze und integrierte Integration mit S3-verwaltetem Objektspeicher für leseintensive Inferenzmuster (z. B. Streaming-Modelleingaben) bietet, und dann anhand von Performance-Benchmark-Tests zu prüfen, ob er für Ihre Workloads geeignet ist.

- EKS Pod Identity Agent — Um den Zugriff auf AWS Dienste zu ermöglichen, verwenden wir den EKS Pod Identity Agent, der einen einzigen Dienstprinzipal verwendet und IAM-Rollenzuordnungen auf Pod-Ebene innerhalb des Amazon EKS-Clusters ermöglicht. EKS Pod Identity bietet eine optimierte Alternative zum herkömmlichen Ansatz derIAM-Rollen für Servicekonten (IRSA), indem es einen einzigen Service-Prinzipal (pods.eks.amazonaws.com) verwendet, anstatt sich auf einzelne OIDC-Anbieter für jeden Cluster zu verlassen, was die Zuweisung von Berechtigungen vereinfacht. Darüber hinaus ermöglicht es die Wiederverwendung von Rollen über mehrere Cluster hinweg und unterstützt erweiterte Features wie IAM- Rollensitzungs-Tags und Ziel-IAM-Rollen.
- EKS-Knoten-Überwachungsagent – Um die kontinuierliche Verfügbarkeit und Zuverlässigkeit der Inferenz-Services sicherzustellen, setzen wir den EKS-Knotenüberwachungsagenten mit automatischer Reparatur ein, der fehlerhafte Knoten automatisch erkennt und ersetzt und so Ausfallzeiten minimiert. Er überwacht Knoten kontinuierlich auf Hardware-, Kernel-, Netzwerk- und Speicherprobleme mithilfe erweiterter Integritätsprüfungen (z. B., KernelReady). NetworkingReady Bei GPU-Knoten erkennt er beschleunigerspezifische Ausfälle und leitet eine effiziente Behebung ein. Hierzu werden fehlerhafte Knoten isoliert, 10 Minuten lang auf die Behebung vorübergehender GPU-Probleme gewartet und bei anhaltenden Ausfällen die Knoten nach 30 Minuten ersetzt.
- Bottlerocket-AMI – Um eine sicherheitsoptimierte Grundlage für unseren EKS-Cluster zu schaffen, setzen wir das Bottlerocket-AMI ein, das ausschließlich die für den Betrieb von Containern erforderlichen Komponenten enthält und minimale Startzeiten für eine schnelle Skalierung bietet. Die Auswahl eines Knoten-AMI ist anwendungsspezifisch und hängt von den Anforderungen Ihrer Workload hinsichtlich Anpassung, Sicherheit und Skalierbarkeit ab. Obwohl das AL2023 AMI mehr Flexibilität für Installationen und Anpassungen auf Host-Ebene bietet (z. B. die Angabe eines dedizierten Cache-Verzeichnisses in einer Konfiguration PV/PVC ohne zusätzliche Knoten), empfehlen wir, mit dem Bottlerocket-AMI zu beginnen, da es einen geringeren Platzbedarf hat und die integrierte Optimierung für containerisierte Workloads (z. B. Microservices, Inferenzserver, skalierbar APIs) bietet, und dann anhand von Performance-Benchmark-Tests zu prüfen, ob es für Ihre Workloads geeignet ist.
- AWS Load Balancer Controller (LBC) — Um Inferenzendpunkte in Echtzeit verfügbar zu machen, verwenden wir den Load AWS Balancer Controller, der automatisch Application Load Balancer () für Traffic und Network Load Balancer (ALBs) für HTTP/HTTPS Traffic auf Basis von Kubernetes Ingress- und Service-Ressourcen bereitstellt und verwaltet und so die Integration von Inferenzmodellen mit externen Clients ermöglicht. NLBs TCP/UDP Darüber hinaus unterstützt es Funktionen wie pfadbasiertes Routing zur Verteilung von Inferenzanfragen auf mehrere Pods oder Knoten, wodurch die Skalierbarkeit bei Verkehrsspitzen gewährleistet und die Latenz durch AWS native Optimierungen wie Verbindungsmultiplexing und Integritätsprüfungen minimiert wird.

1. Erstellen Ihres EKS-Clusters

In diesem Schritt erstellen wir einen Cluster mit CPU-Knoten und einer verwalteten Knotengruppe unter Verwendung einer eksctl-Vorlage mit -Unterstützung. AWS CloudFormation ClusterConfig Durch die Initialisierung des Clusters nur mit CPU-Knoten können wir Karpenter ausschließlich zur Verwaltung von CPU-intensiven Knoten und GPU-Knoten verwenden, um die Ressourcenzuweisung mithilfe von Karpenter zu optimieren, das wir in späteren Schritten erstellen. NodePools Zur Unterstützung unserer Echtzeit-Inferenz-Workloads stellen wir den Cluster mit dem EKS Bottlerocket AMI, dem EKS Node Monitoring Agent, dem EKS Pod Identity Agent, dem Mountpoint S3 CSI Driver, dem Load AWS Balancer Controller (LBC) und den Treibern kube-proxy, vpc-cni und coredns bereit. Die m7g.xlarge-Instances werden für CPU-Systemaufgaben verwendet, darunter das Hosting des Karpenter-Controllers, Add-Ons und anderer Systemkomponenten.

Standardmäßig erstellt `eksctl` eine dedizierte VPC für den Cluster mit einem CIDR-Block von `192.168.0.0/16`. Die VPC umfasst drei öffentliche Subnetze und drei private Subnetze, die jeweils auf drei verschiedene Availability Zones (oder zwei AZs in der `us-east-1` Region) verteilt sind. Dies ist die ideale Methode für die Bereitstellung von Kubernetes-Workloads. Die Vorlage stellt außerdem ein Internet-Gateway bereit, das den öffentlichen Subnetzen über Standard-Routen in ihren Routing-Tabellen und ein einzelnes NAT-Gateway in einem der öffentlichen Subnetze Internetzugang gewährt. Die Standard-Routen in den Routing-Tabellen der privaten Subnetze leiten den ausgehenden Datenverkehr über das NAT-Gateway zum Internetzugang weiter. Weitere Informationen zu dieser Konfiguration finden Sie unter Bereitstellen von Knoten in privaten Subnetzen.

Überprüfung Ihrer Anmeldeinformationen

Prüfen Sie, ob Ihre AWS CLI-Anmeldeinformationen gültig sind und sich bei AWS Diensten authentifizieren können:

```
aws sts get-caller-identity
```

Bei Erfolg gibt die CLI Details zu Ihrer AWS Identität (UserId, Ihrem Konto und Ihrem ARN) zurück.

Überprüfung der Verfügbarkeit von Instances

G5-Instance-Typen stehen in allen Regionen zur Verfügung. Überprüfen Sie Ihre nächstgelegene Region. Beispiel:

```
aws ec2 describe-instance-types --instance-types g5.xlarge g5.2xlarge --region us-
east-1
```

Bei Erfolg ist der G5-Instance-Typ in der von Ihnen angegebenen Region verfügbar.

Das Bottlerocket-AMI ist nicht in allen Regionen verfügbar. Überprüfen Sie dies, indem Sie eine Bottlerocket-AMI-ID für Ihre nächstgelegene Region abrufen. Beispiel:

```
aws ssm get-parameter --name /aws/service/bottlerocket/aws-k8s-1.33/arm64/latest/
image_id \
    --region us-east-1 --query "Parameter.Value" --output text
```

Bei Erfolg ist das Bottlerocket-AMI in der von Ihnen angegebenen Region verfügbar.

Vorbereitung Ihrer Umgebung

Legen Sie zunächst die folgenden Umgebungsvariablen in einem neuen Terminalfenster fest. Hinweis: Ersetzen Sie die Platzhalterbeispiele unbedingt durch Ihre individuellen Werte, einschließlich Cluster-Name, gewünschte Region, Karpenter-Vorversion und Kubernetes-Version.

> **Tip**
>
> Einige Variablen (wie `${AWS_REGION}` und `${K8S_VERSION}`) werden früh im Block definiert und dann in späteren Befehlen referenziert, um Konsistenz zu gewährleisten und Wiederholungen zu vermeiden. Führen Sie die Befehle unbedingt nacheinander aus, damit diese Werte ordnungsgemäß exportiert werden und in nachfolgenden Definitionen zur Verfügung stehen.

```
export TEMPOUT="$(mktemp)"
export K8S_VERSION=1.33
export KARPENTER_VERSION="1.5.0"
export AWS_REGION="us-east-1"
export EKS_CLUSTER_NAME="eks-rt-inference-${AWS_REGION}"
export S3_BUCKET_NAME="eks-rt-inference-models-${AWS_REGION}-$(date +%s)"
export NVIDIA_BOTTLEROCKET_AMI="$(aws ssm get-parameter --name /aws/service/
bottlerocket/aws-k8s-${K8S_VERSION}-nvidia/x86_64/latest/image_id --query
 Parameter.Value --output text)"
```

```
export STANDARD_BOTTLEROCKET_AMI="$(aws ssm get-parameter --name /aws/service/
bottlerocket/aws-k8s-${K8S_VERSION}/arm64/latest/image_id --query Parameter.Value --
output text)"
export AWS_ACCOUNT_ID="$(aws sts get-caller-identity --query Account --output text)"
export ALIAS_VERSION="$(aws ssm get-parameter --name "/aws/service/eks/optimized-
ami/${K8S_VERSION}/amazon-linux-2023/x86_64/standard/recommended/image_id" --query
 Parameter.Value | xargs aws ec2 describe-images --query 'Images[0].Name' --image-ids |
 sed -r 's/^.*(v[[:digit:]]+).*$/\1/')"
```

Erstellung der erforderlichen Rollen und Richtlinien

Karpenter benötigt spezifische IAM-Rollen und -Richtlinien (z. B. die IAM-Rolle des Karpenter Controllers, das Instanzprofil und die Richtlinien), um Instanzen als Kubernetes-Worker-Knoten zu verwalten EC2 . Es verwendet diese Rollen, um Aktionen wie das Starten und Beenden von EC2 Instanzen, das Markieren von Ressourcen und die Interaktion mit anderen Diensten auszuführen. AWS Erstellen Sie die Karpenter-Rollen und -Richtlinien mithilfe der Datei cloudformation.yaml von Karpenter:

```
curl -fsSL https://raw.githubusercontent.com/aws/karpenter-provider-aws/v
${KARPENTER_VERSION}/website/content/en/preview/getting-started/getting-started-with-
karpenter/cloudformation.yaml > "${TEMPOUT}" \
&& aws cloudformation deploy \
  --stack-name "Karpenter-${EKS_CLUSTER_NAME}" \
  --template-file "${TEMPOUT}" \
  --capabilities CAPABILITY_NAMED_IAM \
  --parameter-overrides "ClusterName=${EKS_CLUSTER_NAME}"
```

Das AWS LBC benötigt die Erlaubnis, AWS Load Balancer bereitzustellen und zu verwalten, z. B. ALBs zum Erstellen von Ingress-Ressourcen oder NLBs für Dienste vom Typ. `LoadBalancer` Diese Berechtigungsrichtlinie legen wir bei der Erstellung des Clusters fest. Während der Clustererstellung erstellen wir das Dienstkonto mit eksctl im. ClusterConfig Erstellen Sie die LBC-IAM-Richtlinie:

```
aws iam create-policy \
  --policy-name AWSLoadBalancerControllerIAMPolicy \
  --policy-document "$(curl -fsSL https://raw.githubusercontent.com/kubernetes-sigs/
aws-load-balancer-controller/v2.14.1/docs/install/iam_policy.json)"
```

Wenn der Mountpoint S3 CSI-Treiber installiert ist, sind seine DaemonSet Pods so konfiguriert, dass sie ein Dienstkonto für die Ausführung verwenden. Der Mountpoint für den Mountpoint-S3-CSI-Treiber benötigt die Berechtigung zur Interaktion mit dem Amazon-S3-Bucket, den Sie später in

diesem Leitfaden erstellen. Diese Berechtigungsrichtlinie legen wir bei der Erstellung des Clusters fest. Während der Clustererstellung erstellen wir das Dienstkonto mit eksctl im. ClusterConfig Erstellen Sie die S3-IAM-Richtlinie:

```
aws iam create-policy \
    --policy-name S3CSIDriverPolicy \
    --policy-document "{\"Version\": \"2012-10-17\", \"Statement\": [{\"Effect\":
 \"Allow\", \"Action\": [\"s3:GetObject\", \"s3:PutObject\", \"s3:AbortMultipartUpload
\", \"s3:DeleteObject\", \"s3:ListBucket\"], \"Resource\": [\"arn:aws:s3:::
${S3_BUCKET_NAME}\", \"arn:aws:s3:::${S3_BUCKET_NAME}/*\"]}]}"
```

Hinweis: Sollte bereits eine Rolle mit diesem Namen vorhanden sein, benennen Sie die Rolle um. Die Rolle, die wir in diesem Schritt erstellen, ist spezifisch für Ihren Cluster und Ihren S3-Bucket.

Den Cluster erstellen

In dieser Vorlage erstellt eksctl automatisch ein Kubernetes-Servicekonto für EKS Pod Identity, Knoten-Überwachungsagent, CoreDNS, Kubeproxy und das VPC-CNI-Plugin. Derzeit ist der Mountpoint-S3-CSI-Treiber für EKS Pod Identity nicht verfügbar, daher erstellen wir eine IAM-Rolle für das Servicekonto (IRSA) und einen OIDC-Endpunkt. Darüber hinaus erstellen wir ein Dienstkonto für den Load AWS Balancer Controller (LBC). Für den Zugriff auf Bottlerocket-Knoten hängt eksctl automatisch Amazon SSMManaged InstanceCore for Bottlerocket an, um sichere Shell-Sitzungen über SSM zu ermöglichen.

Führen Sie in demselben Terminal, in dem Sie Ihre Umgebungsvariablen festgelegt haben, den folgenden Befehlsblock aus, um den Cluster zu erstellen:

```
eksctl create cluster -f - <<EOF
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: ${EKS_CLUSTER_NAME}
  region: ${AWS_REGION}
  version: "${K8S_VERSION}"
  tags:
    karpenter.sh/discovery: ${EKS_CLUSTER_NAME}
    # Add more tags if needed for billing
iam:
  # Creates an OIDC endpoint and IRSA service account for the Mountpoint S3 CSI Driver
  # Uses the S3 CSI Driver policy for permissions
  withOIDC: true
```

```
  podIdentityAssociations:
  # Creates the pod identity association and service account
  # Uses the Karpenter controller IAM policy for permissions
  - namespace: "kube-system"
    serviceAccountName: karpenter
    roleName: ${EKS_CLUSTER_NAME}-karpenter
    permissionPolicyARNs:
    - arn:aws:iam::${AWS_ACCOUNT_ID}:policy/KarpenterControllerPolicy-
${EKS_CLUSTER_NAME}
  # Creates the pod identity association and service account
  # Uses the {aws} LBC policy for permissions
  - namespace: kube-system
    serviceAccountName: aws-load-balancer-controller
    createServiceAccount: true
    roleName: AmazonEKSLoadBalancerControllerRole
    permissionPolicyARNs:
    - arn:aws:iam::${AWS_ACCOUNT_ID}:policy/AWSLoadBalancerControllerIAMPolicy
iamIdentityMappings:
- arn: "arn:aws:iam::${AWS_ACCOUNT_ID}:role/KarpenterNodeRole-${EKS_CLUSTER_NAME}"
  username: system:node:{{EC2PrivateDNSName}}
  groups:
  - system:bootstrappers
  - system:nodes
managedNodeGroups:
  # Creates 2 CPU nodes for lightweight system tasks
  - name: ${EKS_CLUSTER_NAME}-m7-cpu
    instanceType: m7g.xlarge
    amiFamily: Bottlerocket
    desiredCapacity: 2
    minSize: 1
    maxSize: 10
    labels:
      role: cpu-worker
# Enable automatic Pod Identity associations for VPC CNI Driver, coreDNS, kube-proxy
addonsConfig:
  autoApplyPodIdentityAssociations: true
addons:
  # Installs the S3 CSI Driver addon and creates IAM role
  # Uses the S3 CSI Driver policy for IRSA permissions
  - name: aws-mountpoint-s3-csi-driver
    attachPolicyARNs:
      - "arn:aws:iam::${AWS_ACCOUNT_ID}:policy/S3CSIDriverPolicy"
  - name: eks-pod-identity-agent
  - name: eks-node-monitoring-agent
```

```
  - name: coredns
  - name: kube-proxy
  - name: vpc-cni
EOF
```

Dieser Vorgang dauert einige Minuten. Wenn Sie den Status überwachen möchten, schauen Sie in der Konsole nach. AWS CloudFormation

2. Verifizierung des Zustands von Cluster-Knoten und Pods

Wir führen einige Zustandsprüfungen durch, um sicherzustellen, dass der Cluster einsatzbereit ist. Wenn der vorherige Befehl abgeschlossen ist, zeigen Sie die Instance-Typen an und überprüfen Sie mit dem folgenden Befehl, ob Ihre CPU-Systemknoten den `Ready`-Status erreicht haben:

```
kubectl get nodes -L node.kubernetes.io/instance-type
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
NAME                              STATUS   ROLES    AGE     VERSION
 INSTANCE-TYPE
ip-192-168-35-103.ec2.internal   Ready    <none>   12m     v1.33.0-eks-802817d
 m7g.xlarge
ip-192-168-7-15.ec2.internal     Ready    <none>   12m     v1.33.0-eks-802817d
 m7g.xlarge
```

Überprüfen Sie alle Pod-Identity-Zuordnungen und wie diese eine Rolle einem Servicekonto in einem Namespace im Cluster zuordnen, mit dem folgenden Befehl:

```
eksctl get podidentityassociation --cluster ${EKS_CLUSTER_NAME} --region ${AWS_REGION}
```

In der Ausgabe sollten die IAM-Rollen für Karpenter („karpenter“) und AWS LBC (“ „) angezeigt werden. aws-load-balancer-controller

Stellen Sie sicher, dass sie verfügbar sind: DaemonSets

```
kubectl get daemonsets -n kube-system
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
NAME                          DESIRED CURRENT READY UP-TO-DATE AVAILABLE NODE SELECTOR
          AGE
aws-node                      3       3       3     3          3         <none>
          12m
dcgm-server                   0       0       0     0          0
 kubernetes.io/os=linux 12m
eks-node-monitoring-agent     3       3       3     3          3
 kubernetes.io/os=linux 12m
eks-pod-identity-agent        3       3       3     3          3         <none>
          12m
kube-proxy                    3       3       3     3          3         <none>
          12m
s3-csi-node                   2       2       2     2          2
 kubernetes.io/os=linux 12m
```

Überprüfen Sie, ob alle Add-Ons auf dem Cluster installiert sind:

```
eksctl get addons --cluster ${EKS_CLUSTER_NAME} --region ${AWS_REGION}
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
NAME                          VERSION              STATUS    ISSUES    IAMROLE
                                     UPDATE AVAILABLE    CONFIGURATION VALUES    POD
 IDENTITY ASSOCIATION ROLES
aws-mountpoint-s3-csi-driver   v1.15.0-eksbuild.1   ACTIVE    0
 arn:aws:iam::143095308808:role/eksctl-eks-rt-inference-us-east-1-addon-aws-m-Role1-
RAUjk4sJnc0L
coredns                       v1.12.1-eksbuild.2   ACTIVE    0
eks-node-monitoring-agent     v1.3.0-eksbuild.2    ACTIVE    0
eks-pod-identity-agent        v1.3.7-eksbuild.2    ACTIVE    0
kube-proxy                    v1.33.0-eksbuild.2   ACTIVE    0
metrics-server                v0.7.2-eksbuild.3    ACTIVE    0
vpc-cni                       v1.19.5-eksbuild.1   ACTIVE    0
```

3. Installation von Karpenter

Installieren Sie den Karpenter-Controller auf Ihren CPU-Worker-Knoten (`cpu-worker`), um Kosten zu optimieren und GPU-Ressourcen zu schonen. Wir installieren ihn im Namespace „kube-system“ und geben das Servicekonto „karpenter“ an, das wir bei der Erstellung des Clusters definiert haben. Darüber hinaus konfiguriert dieser Befehl den Cluster-Namen und eine Spot-Instance-

Unterbrechungs-Warteschlange für CPU-Knoten. Karpenter wird IRSA verwenden, um diese IAM-Rolle zu übernehmen.

```
# Logout of helm registry before pulling from public ECR
helm registry logout public.ecr.aws

# Install Karpenter
helm upgrade --install karpenter oci://public.ecr.aws/karpenter/karpenter --version
 "${KARPENTER_VERSION}" --namespace "kube-system" --create-namespace  \
  --set "settings.clusterName=${EKS_CLUSTER_NAME}" \
  --set "settings.interruptionQueue=${EKS_CLUSTER_NAME}" \
  --set controller.resources.requests.cpu=1 \
  --set controller.resources.requests.memory=1Gi \
  --set controller.resources.limits.cpu=1 \
  --set controller.resources.limits.memory=1Gi \
  --set serviceAccount.annotations."eks\.amazonaws\.com/role-arn"="arn:aws:iam::
${AWS_ACCOUNT_ID}:role/${EKS_CLUSTER_NAME}-karpenter" \
  --wait
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
Release "karpenter" does not exist. Installing it now.
Pulled: public.ecr.aws/karpenter/karpenter:1.5.0
Digest: sha256:9a155c7831fbff070669e58500f68d7ccdcf3f7c808dcb4c21d3885aa20c0a1c
NAME: karpenter
LAST DEPLOYED: Thu Jun 19 09:57:06 2025
NAMESPACE: kube-system
STATUS: deployed
REVISION: 1
TEST SUITE: None
```

Überprüfen Sie, ob Karpenter ausgeführt wird:

```
kubectl get pods -n kube-system -l app.kubernetes.io/name=karpenter
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
NAME                         READY   STATUS    RESTARTS   AGE
karpenter-555895dc-865bc   1/1     Running   0          5m58s
karpenter-555895dc-j7tk9   1/1     Running   0          5m58s
```

4. Richten Sie Karpenter ein NodePools

In diesem Schritt konfigurieren wir die sich gegenseitig ausschließende CPU- und GPU-Karpenter. NodePools Das `limits` Feld in der NodePool Spezifikation schränkt die maximalen Gesamtressourcen (z. B. CPU, Speicher GPUs) ein, die jeder für alle bereitgestellten Knoten verbrauchen NodePool kann. Dadurch wird verhindert, dass zusätzliche Knoten bereitgestellt werden, wenn diese Grenzwerte überschritten werden. NodePools Unterstützt zwar breite Instanzkategorien (z. B.,`g)c`, aber die Angabe bestimmter Instanztypen, Kapazitätstypen und Ressourcenlimits hilft Ihnen dabei, die Kosten für Ihre On-Demand-Workloads einfacher abzuschätzen. In diesen NodePools Fällen verwenden wir eine Vielzahl von Instance-Typen innerhalb der G5-Instance-Familie. Auf diese Weise kann Karpenter auf der Grundlage von Pod-Ressourcenanfragen automatisch den am besten geeigneten Instanztyp auswählen, wodurch die Ressourcennutzung optimiert und gleichzeitig die NodePool Gesamtlimits eingehalten werden. Weitere Informationen finden Sie unter Erstellen. NodePools

Richten Sie die GPU ein NodePool

Darin legen wir Ressourcenlimits fest NodePool, um die Bereitstellung von Knoten mit GPU-Fähigkeiten zu verwalten. Diese Beschränkungen dienen dazu, die Gesamtressourcen aller Knoten im Pool zu begrenzen, wobei insgesamt bis zu 10 Instances zulässig sind. Jede Instanz kann entweder g5.xlarge (4 VCPUs, 16 GiB Arbeitsspeicher, 1 GPU) oder g5.2xlarge (8 VCPUs, 32 GiB Speicher, 1 GPU) sein, solange die Summe v 80 CPUs nicht überschreitet, der Gesamtspeicher 320 GiB nicht überschreitet und die Summe 10 nicht überschreitet. GPUs Der Pool kann beispielsweise 10 g5.2xlarge-Instances (80 vCPUs, 320 GiB, 10 GPUs) oder 10 g5.xlarge-Instances (40 v, 160 GiBCPUs, 10 GPUs) oder eine Mischung wie 5 g5.xlarge- und 5 g5.2xlarge (60 v, 240 GiB, 10 GPUs) bereitstellenCPUs, wodurch Flexibilität je nach Workload-Anforderungen gewährleistet und gleichzeitig Ressourcenbeschränkungen eingehalten werden.

Darüber hinaus geben wir die ID der Nvidia-Variante der Bottlerocket-AMI an. Schließlich legen wir eine Unterbrechungsrichtlinie fest, um leere Knoten nach 30 Minuten (`consolidateAfter: 30m`) zu entfernen, und legen eine maximale Lebensdauer der Knoten auf 30 Tage (`expireAfter: 720h`) fest, um die Kosten zu optimieren und die Knotenintegrität für GPU-intensive Aufgaben aufrechtzuerhalten. Weitere Informationen finden Sie unter Deaktivieren von Karpenter Consolidation für unterbrechungssensitive Workloads und Verwenden Sie ttlSecondsAfter Finished to Auto Clean-Up Kubernetes-Jobs.

```
cat <<EOF | envsubst | kubectl apply -f -
apiVersion: karpenter.sh/v1
kind: NodePool
```

```
metadata:
  name: gpu-a10g-inference-g5
spec:
  template:
    metadata:
      labels:
        role: gpu-worker
        gpu-type: nvidia-a10g
    spec:
      requirements:
        - key: node.kubernetes.io/instance-type
          operator: In
          values: ["g5.xlarge", "g5.2xlarge"]
        - key: "karpenter.sh/capacity-type"
          operator: In
          values: ["on-demand"]
      taints:
        - key: nvidia.com/gpu
          value: "true"
          effect: NoSchedule
      nodeClassRef:
        name: gpu-a10g-inference-ec2
        group: karpenter.k8s.aws
        kind: EC2NodeClass
      expireAfter: 720h
  limits:
    cpu: "80"
    memory: "320Gi"
    nvidia.com/gpu: "10"
  disruption:
    consolidationPolicy: WhenEmpty
    consolidateAfter: 30m
---
apiVersion: karpenter.k8s.aws/v1
kind: EC2NodeClass
metadata:
  name: gpu-a10g-inference-ec2
spec:
  amiFamily: Bottlerocket
  amiSelectorTerms:
    - id: ${NVIDIA_BOTTLEROCKET_AMI}
  role: "KarpenterNodeRole-${EKS_CLUSTER_NAME}"
  subnetSelectorTerms:
    - tags:
```

```
        karpenter.sh/discovery: "${EKS_CLUSTER_NAME}"
  securityGroupSelectorTerms:
    - tags:
        karpenter.sh/discovery: "${EKS_CLUSTER_NAME}"
  tags:
    nvidia.com/gpu: "true"
EOF
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
nodepool.karpenter.sh/gpu-a10g-inference-g5 created
ec2nodeclass.karpenter.k8s.aws/gpu-a10g-inference-ec2 created
```

Stellen Sie sicher, dass das erstellt wurde und fehlerfrei ist: NodePool

```
kubectl get nodepool gpu-a10g-inference-g5 -o yaml
```

Suchen Sie nach „`status.conditions`like“ `ValidationSucceeded: True``NodeClassReady: True`, und bestätigen `Ready: True` Sie, dass der NodePool fehlerfrei ist.

Richten Sie die CPU ein NodePool

Dabei NodePool haben wir Grenzwerte für die Unterstützung von etwa 50 Instances festgelegt, die auf eine moderate CPU-Arbeitslast (z. B. 100-200 Pods) und typische AWS vCPU-Kontingente (z. B. 128-1152) abgestimmt sind. Die Grenzwerte werden unter der Annahme berechnet, dass bis zu 50 m7.xlarge-Instances skaliert werden NodePool sollten: CPU (4 V CPUs pro Instanz × 50 Instanzen = 200 VCPUs) und Arbeitsspeicher (16 GiB pro Instanz × 50 Instanzen = 800 GiB). Diese Grenzwerte sind so konzipiert, dass sie die Gesamtressourcen auf allen Knoten im Pool begrenzen, sodass bis zu 50 m7g.xlarge-Instances (jeweils mit 4 V CPUs und 16 GiB Speicher) zulässig sind, sofern die Gesamt-V 200 CPUs nicht überschreiten und der Gesamtspeicher 800 GiB nicht überschreitet.

Darüber hinaus geben wir die ID der Standardvariante der Bottlerocket-AMI an. Schließlich legen wir eine Unterbrechungsrichtlinie fest, um leere Knoten nach 60 Minuten (`consolidateAfter: 60m`) zu entfernen, und legen eine maximale Lebensdauer der Knoten auf 30 Tage (`expireAfter: 720h`) fest, um die Kosten zu optimieren und die Knotenintegrität für GPU-intensive Aufgaben aufrechtzuerhalten. Weitere Informationen finden Sie unter Deaktivieren von Karpenter Consolidation für unterbrechungssensitive Workloads und Verwenden Sie Fertig, um Kubernetes-Jobs automatisch zu bereinigen. ttlSecondsAfter

```
cat <<EOF | envsubst | kubectl apply -f -
```

```
apiVersion: karpenter.sh/v1
kind: NodePool
metadata:
  name: cpu-inference-m7gxlarge
spec:
  template:
    metadata:
      labels:
        role: cpu-worker
    spec:
      requirements:
        - key: node.kubernetes.io/instance-type
          operator: In
          values: ["m7g.xlarge"]
        - key: karpenter.sh/capacity-type
          operator: In
          values: ["on-demand"]
      taints:
        - key: role
          value: cpu-intensive
          effect: NoSchedule
      nodeClassRef:
        name: cpu-inference-m7gxlarge-ec2
        group: karpenter.k8s.aws
        kind: EC2NodeClass
      expireAfter: 720h
  limits:
    cpu: "200"
    memory: "800Gi"
  disruption:
    consolidationPolicy: WhenEmpty
    consolidateAfter: 60m
---
apiVersion: karpenter.k8s.aws/v1
kind: EC2NodeClass
metadata:
  name: cpu-inference-m7gxlarge-ec2
spec:
  amiFamily: Bottlerocket
  amiSelectorTerms:
    - id: ${STANDARD_BOTTLEROCKET_AMI}
  role: "KarpenterNodeRole-${EKS_CLUSTER_NAME}"
  subnetSelectorTerms:
    - tags:
```

```
        karpenter.sh/discovery: "${EKS_CLUSTER_NAME}"
  securityGroupSelectorTerms:
    - tags:
        karpenter.sh/discovery: "${EKS_CLUSTER_NAME}"
EOF
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
nodepool.karpenter.sh/cpu-inference-m7gxlarge created
ec2nodeclass.karpenter.k8s.aws/cpu-inference-m7gxlarge-ec2 created
```

Stellen Sie sicher, dass das erstellt wurde und fehlerfrei ist: NodePool

```
kubectl get nodepool cpu-inference-m7gxlarge -o yaml
```

Suchen Sie nach „`status.conditions`like“ `ValidationSucceeded: True``NodeClassReady: True`, und bestätigen `Ready: True` Sie, dass der NodePool fehlerfrei ist.

5. Bereitstellung eines GPU-Pods zur Freigabe einer GPU

Sie benötigen das Nvidia-Geräte-Plugin, damit Kubernetes GPU-Geräte für den Kubernetes-Cluster freigeben kann. In der Regel müssten Sie das Plug-in als DaemonSet bereitstellen. Das Bottlerocket-AMI installiert das Plugin jedoch als Teil des AMI vor. Das heißt, wenn Sie Bottlerocket verwenden AMIs, müssen Sie das Nvidia-Geräte-Plugin nicht bereitstellen. DaemonSet Weitere Informationen finden Sie unter Kubernetes Device Plugin to Expose. GPUs

Bereitstellung eines Beispiel-Pods

Karpenter agiert dynamisch: Es stellt GPU-Knoten bereit, wenn eine Workload (Pod) GPU-Ressourcen anfordert. Um zu überprüfen, ob Pods Anfragen stellen und verwenden können GPUs, stellen Sie einen Pod bereit, der die `nvidia.com/gpu` Ressource innerhalb ihrer Limits anfordert (z. B.`nvidia.com/gpu: 1`). Weitere Informationen zu diesen Bezeichnungen finden Sie unter Planen von Workloads mit GPU-Anforderungen unter Verwendung bekannter Bezeichnungen.

```
cat <<EOF | envsubst | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: gpu-nvidia-smi
spec:
  restartPolicy: OnFailure
```

```
  tolerations:
  - key: "nvidia.com/gpu"
    operator: "Exists"
    effect: "NoSchedule"
  nodeSelector:
    role: gpu-worker  # Matches GPU NodePool's label
  containers:
  - name: cuda-container
    image: nvidia/cuda:12.9.1-base-ubuntu20.04
    command: ["nvidia-smi"]
    resources:
      limits:
        nvidia.com/gpu: 1
      requests:
        nvidia.com/gpu: 1
EOF
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
pod/gpu-ndivia-smi created
```

Warten Sie eine Minute und überprüfen Sie dann, ob der Pod den Status „Ausstehend“ContainerCreating, „Wird ausgeführt“ und dann den Status „Abgeschlossen“ hat:

```
kubectl get pod gpu-nvidia-smi -w
```

Stellen Sie sicher, dass der Knoten für den Pod zur GPU gehört NodePool:

```
kubectl get node $(kubectl get pod gpu-nvidia-smi -o jsonpath='{.spec.nodeName}') -o
 custom-columns="Name:.metadata.name,Nodepool:.metadata.labels.karpenter\.sh/nodepool"
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
Name                            Nodepool
ip-192-168-83-245.ec2.internal   gpu-a10g-inference-g5
```

Überprüfen Sie die Pod-Protokolle:

```
kubectl logs gpu-nvidia-smi
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
Thu Jul 17 04:31:33 2025
+-----------------------------------------------------------------------------------------
+
| NVIDIA-SMI 570.148.08                 Driver Version: 570.148.08         CUDA
 Version: 12.9 |
|-----------------------------------------+----------------------
+----------------------+
| GPU  Name                 Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC
 |
| Fan  Temp   Perf          Pwr:Usage/Cap |         Memory-Usage | GPU-Util  Compute M.
 |
|                                         |                      |               MIG M.
 |
|=========================================+======================
+======================|
|   0  NVIDIA A10G                    On  | 00000000:00:1E.0 Off |                    0
 |
|  0%   30C    P8               9W / 300W |      0MiB / 23028MiB |      0%      Default
 |
|                                         |                      |                  N/A
 |
+-----------------------------------------------------------------------------------------
+

+-----------------------------------------------------------------------------------------
+
| Processes:
 |
|  GPU        GI     CI        PID   Type   Process name                 GPU Memory
 |
|                     ID        ID                                        Usage
 |
|
=========================================================================================|
|  No running processes found
 |
+-----------------------------------------------------------------------------------------
+
```

6. (Optional) Modell-Artefakte für die Bereitstellung vorbereiten und hochladen

In diesem Schritt stellen Sie einen Modell-Service für die Echtzeit-Image-Klassifizierung bereit, beginnend mit dem Hochladen der Modellgewichte in einen Amazon-S3-Bucket. Zur Demonstration

verwenden wir das Open-Source-Modell GPUNet-0 Vision von NVIDIA GPUNet, das Inferenzen mit niedriger Latenz auf Bildern mit NVIDIA und TensorRT unterstützt. GPUs Dieses Modell ist vortrainiert ImageNet, ermöglicht es uns, Objekte in Fotos oder Videostreams im Handumdrehen zu klassifizieren. Es gilt als kleines Modell mit 11,9 Millionen Parametern.

So richten Sie Ihre Umgebung ein

Um die Gewichte des Modells GPUNet -0 herunterzuladen In diesem Schritt benötigen Sie Zugriff auf den NGC-Katalog von NVIDIA und müssen Docker auf Ihrem lokalen Computer installiert sein. Befolgen Sie die folgenden Schritte, um ein kostenloses Konto einzurichten und die NGC-CLI zu konfigurieren:

- Registrieren Sie sich für ein kostenloses NGC-Konto und generieren Sie einen API-Schlüssel über das NGC-Dashboard (Benutzersymbol > Einstellungen > API-Schlüssel generieren > Persönlichen Schlüssel generieren > NGC-Katalog).
- Laden Sie die NGC-CLI (Linux/macOS/Windows) herunter, installieren Sie sie und konfigurieren Sie die CLI mit:`ngc config set`. Geben Sie Ihren API-Schlüssel ein, wenn Sie dazu aufgefordert werden. Setzen Sie „org“ auf `nvidia` und drücken Sie die Eingabetaste, um die Standardeinstellungen für andere zu akzeptieren. Bei erfolgreicher Eingabe sollte etwa Folgendes angezeigt werden: `Successfully saved NGC configuration to /Users/your-username/.ngc/config`.

Verifizierung der Berechtigungen für Servicekonten

Bevor wir beginnen, überprüfen Sie die Berechtigungen des Kubernetes-Servicekontos:

```
kubectl get serviceaccount s3-csi-driver-sa -n kube-system -o yaml
```

Bei der Clustererstellung haben wir die CSIDriver S3-Richtlinie einer IAM-Rolle zugeordnet und das Dienstkonto mit Anmerkungen versehen („s3-csi-driver-sa“). Die Mountpoint-S3-CSI-Treiber-Pods übernehmen die Berechtigungen der IAM-Rolle, wenn sie mit S3 interagieren. Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
apiVersion: v1
kind: ServiceAccount
metadata:
  annotations:
    eks.amazonaws.com/role-arn: arn:aws:iam::143095308808:role/eksctl-eks-rt-inference-
us-east-1-addon-aws-m-Role1-fpXXjRYdKN8r
```

```
  creationTimestamp: "2025-07-17T03:55:29Z"
  labels:
    app.kubernetes.io/component: csi-driver
    app.kubernetes.io/instance: aws-mountpoint-s3-csi-driver
    app.kubernetes.io/managed-by: EKS
    app.kubernetes.io/name: aws-mountpoint-s3-csi-driver
  name: s3-csi-driver-sa
  namespace: kube-system
  resourceVersion: "2278"
  uid: 50b36272-6716-4c68-bdc3-c4054df1177c
```

Toleranz hinzufügen

Der S3-CSI-Treiber wird DaemonSet auf allen Knoten als a ausgeführt. Pods verwenden den CSI-Treiber auf diesen Knoten, um S3-Volumes einzubinden. Damit er auf unseren GPU-Knoten planen kann, die Fehler aufweisen, fügen Sie folgende Toleranzen hinzu: DaemonSet

```
kubectl patch daemonset s3-csi-node -n kube-system --type='json' -p='[{"op": "add",
 "path": "/spec/template/spec/tolerations/-", "value": {"key": "nvidia.com/gpu",
 "operator": "Exists", "effect": "NoSchedule"}}]'
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
daemonset.apps/s3-csi-node patched
```

Modellgewichte in S3 hochladen

In diesem Schritt erstellen Sie einen Amazon S3 S3-Bucket, laden die GPUNet -0 Modellgewichte von NVIDIA GPU Cloud (NGC) herunter und laden sie in den Bucket hoch. Auf diese Gewichte greift unsere Anwendung zur Laufzeit für die Inferenz zu.

Erstellen Sie Ihren Amazon-S3-Bucket:

```
aws s3 mb s3://${S3_BUCKET_NAME} --region ${AWS_REGION}
```

Aktivieren Sie die S3-Versionsverwaltung für den Bucket, um zu verhindern, dass versehentliche Löschungen und Überschreibungen zu sofortigem und dauerhaftem Datenverlust führen:

```
aws s3api put-bucket-versioning --bucket ${S3_BUCKET_NAME} --versioning-configuration
 Status=Enabled
```

Wenden Sie eine Lebenszyklusregel auf den Bucket an, um überschriebene oder gelöschte Objektversionen 14 Tage nach ihrer Veraltung zu entfernen, abgelaufene Löschmarkierungen zu entfernen und unvollständige mehrteilige Uploads nach 7 Tagen zu entfernen. Weitere Informationen finden Sie unter Beispiele für S3-Lebenszykluskonfigurationen.

```
aws s3api put-bucket-lifecycle-configuration --bucket $S3_BUCKET_NAME --lifecycle-
configuration '{"Rules":[{"ID":"LifecycleRule","Status":"Enabled","Filter":
{},"Expiration":{"ExpiredObjectDeleteMarker":true},"NoncurrentVersionExpiration":
{"NoncurrentDays":14},"AbortIncompleteMultipartUpload":{"DaysAfterInitiation":7}}]}'
```

Laden Sie die Gewichte des Modells GPUNet -0 von NGC herunter. Zum Beispiel unter macOS:

```
ngc registry model download-version nvidia/dle/gpunet_0_pyt_ckpt:21.12.0_amp --dest ~/
downloads
```

> **Note**
>
> Möglicherweise müssen Sie diesen Download-Befehl an Ihr Betriebssystem anpassen. Damit dieser Befehl auf einem Linux-System funktioniert, müssen Sie das Verzeichnis wahrscheinlich als Teil des Befehls erstellen (z. B. `mkdir ~/downloads`).

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
{
  "download_end": "2025-07-18 08:22:39",
  "download_start": "2025-07-18 08:22:33",
  "download_time": "6s",
  "files_downloaded": 1,
  "local_path": "/Users/your-username/downloads/gpunet_0_pyt_ckpt_v21.12.0_amp",
  "size_downloaded": "181.85 MB",
  "status": "Completed",
  "transfer_id": "gpunet_0_pyt_ckpt[version=21.12.0_amp]"
}
```

Benennen Sie die Checkpoint-Datei so um, dass sie der erwarteten Benennung in unserem Anwendungscode in späteren Schritten entspricht (es ist kein Extrahieren erforderlich, da es sich um einen standardmäßigen PyTorch *.pth.tar-Checkpoint handelt, der das Modellstatuswörterbuch enthält):

```
mv ~/downloads/gpunet_0_pyt_ckpt_v21.12.0_amp/0.65ms.pth.tar gpunet-0.pth
```

Aktivieren Sie die AWS Common Runtime in der AWS CLI, um den S3-Durchsatz zu optimieren:

```
aws configure set s3.preferred_transfer_client crt
```

Laden Sie die Modellgewichte in Ihren S3-Bucket hoch:

```
aws s3 cp gpunet-0.pth s3://${S3_BUCKET_NAME}/gpunet-0.pth
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
upload: ./gpunet-0.pth to s3://eks-rt-inference-models-us-east-1-1752722786/
gpunet-0.pth
```

Modell-Service erstellen

In diesem Schritt richten Sie eine FastAPI-Webanwendung für die GPU-beschleunigte Bildklassifizierung mithilfe des GPUNet -0 Vision-Modells ein. Die Anwendung lädt zur Laufzeit Modellgewichte von Amazon S3 herunter, ruft die Modellarchitektur zum Zwischenspeichern aus dem NVIDIA-Repository ab und lädt ImageNet Klassenbezeichnungen über HTTP herunter. Die Anwendung beinhaltet Image-Vorverarbeitungstransformationen und stellt zwei Endpunkte bereit: einen Root-GET-Endpunkt `/predict` für die Statusüberprüfung und einen POST-Endpunkt, der eine Bild-URL akzeptiert.

Wir bieten das Modell mithilfe von FastAPI an und laden Gewichte zur Laufzeit aus Amazon S3 in einem containerisierten Setup für schnelles Prototyping und Kubernetes-Bereitstellung. PyTorch Weitere Methoden wie optimiertes Batching oder Hochdurchsatz-Engines finden Sie unter Bereitstellen von ML-Modellen.

Erstellen der Anwendung

Erstellen Sie ein Verzeichnis für Ihre Anwendungsdateien, z. B. `model-testing`, wechseln Sie dann in dieses Verzeichnis und fügen Sie den folgenden Code zu einer neuen Datei mit dem Namen `app.py` hinzu:

```
import os
import torch
import json
import requests
```

```
from fastapi import FastAPI, HTTPException
from PIL import Image
from io import BytesIO, StringIO
import torchvision.transforms as transforms
from torch.nn.functional import softmax
import warnings
from contextlib import redirect_stdout, redirect_stderr
import argparse
import boto3
app = FastAPI()

# Suppress specific warnings from the model code (quantization is optional and unused
 here)
warnings.simplefilter("ignore", UserWarning)

device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

# Load model code from cache (if present)
# Use backed cache directory
torch.hub.set_dir('/cache/torch/hub')

# Allowlist for secure deserialization (handles potential issues in older checkpoints)
torch.serialization.add_safe_globals([argparse.Namespace])
# Load the model architecture only on container startup (changed to pretrained=False)
# Precision (FP32 for full accuracy, could be 'fp16' for speed on Ampere+ GPUs)
with redirect_stdout(StringIO()), redirect_stderr(StringIO()):
    gpunet = torch.hub.load('NVIDIA/DeepLearningExamples:torchhub', 'nvidia_gpunet',
 pretrained=False, model_type='GPUNet-0', model_math='fp32')

# Download weights from S3 if not present, then load them
model_path = os.getenv('MODEL_PATH', '/cache/torch/hub/checkpoints/gpunet-0.pth')
os.makedirs(os.path.dirname(model_path), exist_ok=True)  # Ensure checkpoints dir
 exists
if not os.path.exists(model_path):
    s3 = boto3.client('s3')
    s3.download_file(os.getenv('S3_BUCKET_NAME'), 'gpunet-0.pth', model_path)
checkpoint = torch.load(model_path, map_location=device, weights_only=True)
gpunet.load_state_dict(checkpoint['state_dict'])
# Move to GPU/CPU
gpunet.to(device)
gpunet.eval()

# Preprocessing
preprocess = transforms.Compose([
```

```
    transforms.Resize(256),
    transforms.CenterCrop(224),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
])

# Load ImageNet labels
labels_url = "https://s3.amazonaws.com/deep-learning-models/image-models/
imagenet_class_index.json"
response = requests.get(labels_url)
json_data = json.loads(response.text)
labels = [json_data[str(i)][1].replace('_', ' ') for i in range(1000)]

# Required, FastAPI root
@app.get("/")
async def hello():
    return {"status": "hello"}

# Serve model requests
@app.post("/predict")
async def predict(image_url: str):
    try:
        response = requests.get(image_url)
        response.raise_for_status()
        img = Image.open(BytesIO(response.content)).convert("RGB")
        input_tensor = preprocess(img).unsqueeze(0).to(device)

        with torch.no_grad():
            output = gpunet(input_tensor)

        probs = softmax(output, dim=1)[0]
        top5_idx = probs.topk(5).indices.cpu().numpy()
        top5_probs = probs.topk(5).values.cpu().numpy()

        results = [{ "label": labels[idx], "probability": float(prob) } for idx, prob
 in zip(top5_idx, top5_probs)]

        return {"predictions": results}
    except Exception as e:
        raise HTTPException(status_code=400, detail=str(e))
```

Docker-Datei erstellen

Das folgende Dockerfile erstellt ein Container-Image für unsere Anwendung unter Verwendung des GPUNet Modells aus dem Repository NVIDIA Deep Learning Examples for Tensor Cores. GitHub

Wir reduzieren die Größe des Container-Images, indem wir eine reine PyTorch Runtime-Basis verwenden, nur wichtige Pakete mit Cache-Bereinigung installieren, Modellcode vorab zwischenspeichern und vermeiden, dass Gewichte im Container-Image „überbacken“ werden, um schnellere Abrufe und Updates zu ermöglichen. Weitere Informationen finden Sie unter Reduzieren der Größe von Container-Images.

Erstellen Sie im selben Verzeichnis wie `app.py` die `Dockerfile`:

```
FROM pytorch/pytorch:2.4.0-cuda12.4-cudnn9-runtime

# Install required system packages required for git cloning
RUN apt-get update && apt-get install -y git && rm -rf /var/lib/apt/lists/*

# Install application dependencies
RUN pip install --no-cache-dir fastapi uvicorn requests pillow boto3 timm==0.5.4

# Pre-cache the GPUNet code from Torch Hub (without weights)
# Clone the repository containing the GPUNet code
RUN mkdir -p /cache/torch/hub && \
    cd /cache/torch/hub && \
    git clone --branch torchhub --depth 1 https://github.com/NVIDIA/
DeepLearningExamples NVIDIA_DeepLearningExamples_torchhub

COPY app.py /app/app.py

WORKDIR /app

CMD ["uvicorn", "app:app", "--host", "0.0.0.0", "--port", "80"]
```

Testen der Anwendung

Erstellen Sie das Container-Image für die Inferenzanwendung aus demselben Verzeichnis wie Ihr `app.py` und `Dockerfile` erstellen Sie dabei die folgende Architektur: AMD64

```
docker build --platform linux/amd64 -t gpunet-inference-app .
```

Legen Sie Umgebungsvariablen für Ihre AWS Anmeldeinformationen und optional ein AWS Sitzungstoken fest. Beispiel:

```
export AWS_REGION="us-east-1"
export AWS_ACCESS_KEY_ID=ABCEXAMPLESCUJFEIELSMUHHAZ
export AWS_SECRET_ACCESS_KEY=123EXAMPLEMZREoQXr8XkiicsOgWDQ5TpUsq0/Z
```

Führen Sie den Container lokal aus und fügen Sie AWS Anmeldeinformationen als Umgebungsvariablen für den S3-Zugriff ein. Beispiel:

```
docker run --platform linux/amd64 -p 8080:80 \
  -e S3_BUCKET_NAME=${S3_BUCKET_NAME} \
  -e AWS_ACCESS_KEY_ID=${AWS_ACCESS_KEY_ID} \
  -e AWS_SECRET_ACCESS_KEY=${AWS_SECRET_ACCESS_KEY} \
  -e AWS_DEFAULT_REGION=${AWS_REGION} \
  gpunet-inference-app
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
INFO:     Started server process [1]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://0.0.0.0:80 (Press CTRL+C to quit)
```

Testen Sie in einem neuen Terminalfenster den Inferenz-Endpunkt, indem Sie eine Beispiel-POST-Anfrage mit einer öffentlichen Image-URL als Abfrageparameter senden:

```
curl -X POST "http://localhost:8080/predict?image_url=http://images.cocodataset.org/
test-stuff2017/000000024309.jpg"
```

Die erwartete Ausgabe sollte eine JSON-Antwort mit den fünf besten Prognosen sein, ähnlich wie hier (die tatsächlichen Bezeichnungen und Wahrscheinlichkeiten können je nach Image und Modellgenauigkeit leicht variieren):

```
{"predictions":[{"label":"desk","probability":0.28885871171951294},
{"label":"laptop","probability":0.24679335951805115},
{"label":"notebook","probability":0.08539070934057236},
{"label":"library","probability":0.030645888298749924},
{"label":"monitor","probability":0.02989606373012066}]}
```

Beenden Sie die Anwendung mit „Strg + C“.

Übertragen des Containers an Amazon ECR

In diesem Schritt laden wir das Container-Image für den Service GPUNet -0 Model in Amazon Elastic Container Registry (ECR) hoch, sodass es für die Bereitstellung auf Amazon EKS verfügbar ist. Dieser Prozess umfasst die Erstellung eines neuen ECR-Repositorys zur Speicherung des Images, die Authentifizierung bei ECR sowie das Kennzeichnen und Übertragen des Container-Images in unsere Registry.

Navigieren Sie zunächst zurück zu dem Verzeichnis, in dem Sie zu Beginn dieser Anleitung Ihre Umgebungsvariablen festgelegt haben. Beispiel:

```
cd ..
```

Erstellen Sie ein Repository in Amazon ECR:

```
aws ecr create-repository --repository-name gpunet-inference-app --region ${AWS_REGION}
```

Melden Sie sich bei Amazon ECR an:

```
aws ecr get-login-password --region ${AWS_REGION} | docker login --username AWS --
password-stdin ${AWS_ACCOUNT_ID}.dkr.ecr.${AWS_REGION}.amazonaws.com
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
Login Succeeded
```

Kennzeichnen Sie das Image:

```
docker tag gpunet-inference-app:latest ${AWS_ACCOUNT_ID}.dkr.ecr.
${AWS_REGION}.amazonaws.com/gpunet-inference-app:latest
```

Übertragen Sie das Image in Ihr Amazon-ECR-Repository:

```
docker push ${AWS_ACCOUNT_ID}.dkr.ecr.${AWS_REGION}.amazonaws.com/gpunet-inference-
app:latest
```

Dieser letzte Schritt kann einige Minuten in Anspruch nehmen.

7. (Optional) Modell-Service verfügbar machen

In diesem Schritt stellen Sie Ihren Echtzeit-Inferenzmodell-Service mithilfe des Load AWS Balancer Controllers (LBC) extern auf Amazon EKS zur Verfügung. Dazu gehört die Einrichtung des LBC, die Bereitstellung von Modellgewichten aus Amazon S3 als persistentes Volume mithilfe des Mountpoint-S3-CSI-Treibers, die Bereitstellung eines GPU-beschleunigten Anwendungs-Pods, die Erstellung eines Services und eines Ingresses zur Bereitstellung eines Application Load Balancer (ALB) sowie das Testen des Endpunkts.

Überprüfen Sie zunächst die Pod-Identity-Zuordnung für den AWS LBC und stellen Sie sicher, dass das Dienstkonto ordnungsgemäß mit der erforderlichen IAM-Rolle verknüpft ist:

```
eksctl get podidentityassociation --cluster ${EKS_CLUSTER_NAME} --namespace kube-system
 --service-account-name aws-load-balancer-controller
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
ASSOCIATION ARN                                                NAMESPACE    SERVICE
 ACCOUNT NAME        IAM ROLE ARN    OWNER ARN
arn:aws:eks:us-east-1:143095308808:podidentityassociation/eks-rt-inference-us-
east-1/a-buavluu2wp1jropya    kube-system     aws-load-balancer-controller
 arn:aws:iam::143095308808:role/AmazonEKSLoadBalancerControllerRole
```

Kennzeichnen Ihrer Cluster-Sicherheitsgruppe

Der Load AWS Balancer Controller unterstützt nur eine einzige Sicherheitsgruppe mit dem Tag-Schlüssel `karpenter.sh/discovery: "${EKS_CLUSTER_NAME}"` für die Sicherheitsgruppenauswahl von Karpenter. Beim Erstellen eines Clusters mit eksctl wird die Standard-Cluster-Sicherheitsgruppe (die das `"kubernetes.io/cluster/<cluster-name>: owned"`-Tag hat) nicht automatisch mit `karpenter.sh/discovery`-Tags versehen. Dieses Tag ist für Karpenter unerlässlich, um diese Sicherheitsgruppe zu erkennen und den bereitgestellten Knoten zuzuordnen. Durch das Anhängen dieser Sicherheitsgruppe wird die Kompatibilität mit dem Load AWS Balancer Controller (LBC) gewährleistet, sodass dieser automatisch Regeln für eingehenden Datenverkehr für Dienste verwalten kann, die über Ingress verfügbar gemacht werden, z. B. den Modelldienst in diesen Schritten.

Exportieren Sie die VPC-ID für Ihren Cluster:

```
CLUSTER_VPC_ID="$(aws eks describe-cluster --name ${EKS_CLUSTER_NAME} --query
 cluster.resourcesVpcConfig.vpcId --output text)"
```

Exportieren Sie die Standard-Sicherheitsgruppe für Ihren Cluster:

```
CLUSTER_SG_ID="$(aws ec2 describe-security-groups --filters Name=vpc-id,Values=
$CLUSTER_VPC_ID Name=tag-key,Values=kubernetes.io/cluster/${EKS_CLUSTER_NAME} --query
 'SecurityGroups[].[GroupId]' --output text)"
```

Fügen Sie das `karpenter.sh/discovery`-Tag zur Standard-Cluster-Sicherheitsgruppe hinzu. Dadurch können unsere CPU- und EC2 NodeClass GPU-Selektoren sie verwenden:

```
aws ec2 create-tags --resources ${CLUSTER_SG_ID} --tags Key=karpenter.sh/
discovery,Value=${EKS_CLUSTER_NAME}
```

Überprüfen Sie, ob das Tag hinzugefügt wurde:

```
aws ec2 describe-security-groups --group-ids ${CLUSTER_SG_ID} --query
 "SecurityGroups[].Tags"
```

Unter den Ergebnissen sollten Sie Folgendes mit dem Tag und Ihrem Cluster-Namen sehen. Beispiel:

```
{
  "Key": "karpenter.sh/discovery",
  "Value": "eks-rt-inference-us-east-1"
}
```

Den AWS Load Balancer Controller (LBC) einrichten

Der AWS LBC ist unverzichtbar für die Verwaltung des eingehenden Datenverkehrs zu AI/ML Workloads auf Amazon EKS und gewährleistet den Zugriff auf Inferenzendpunkte oder Datenverarbeitungspipelines. Durch die Integration mit AWS Application Load Balancers (ALB) und Network Load Balancers (NLB) leitet der LBC den Datenverkehr dynamisch an containerisierte Anwendungen weiter, z. B. solche, die umfangreiche Sprachmodelle, Computer-Vision-Modelle oder Echtzeit-Inferenzdienste ausführen. Da wir das Servicekonto und die Pod-Identity-Zuordnung bereits bei der Erstellung des Clusters erstellt haben, legen wir den `serviceAccount.name` so fest, dass er mit der Definition in unserer Cluster-Konfiguration übereinstimmt (`aws-load-balancer-controller`).

Fügen Sie das Helm-Chart-Repository von -own eks-charts hinzu: AWS

```
helm repo add eks https://aws.github.io/eks-charts
```

Aktualisieren Sie Ihre lokalen Helm-Repositorys mit den neuesten Diagrammen:

```
helm repo update eks
```

Stellen Sie das AWS LBC mithilfe von Helm bereit, geben Sie den Namen des EKS-Clusters an und verweisen Sie auf das vorab erstellte Dienstkonto:

```
helm install aws-load-balancer-controller eks/aws-load-balancer-controller \
  -n kube-system \
  --set clusterName=${EKS_CLUSTER_NAME} \
  --set serviceAccount.create=false \
  --set serviceAccount.name=aws-load-balancer-controller
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
NAME: aws-load-balancer-controller
LAST DEPLOYED: Wed Jul 9 15:03:31 2025
NAMESPACE: kube-system
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
AWS Load Balancer controller installed!
```

Einbinden des Modells in ein persistentes Volume

In diesem Schritt mounten Sie Modellgewichte aus Ihrem Amazon S3-Bucket mithilfe eines PersistentVolume (PV), das vom Mountpoint for Amazon S3 CSI-Treiber unterstützt wird. Dadurch können Kubernetes-Pods auf S3-Objekte als lokale Dateien zugreifen, wodurch ressourcenintensive Downloads in flüchtige Pod-Speicher oder Init-Container vermieden werden – ideal für große Modellgewichte mit mehreren Gigabyte.

Die PV bindet die gesamte Bucket-Root ein (kein Pfad in `volumeAttributes` angegeben), unterstützt den gleichzeitigen Lesezugriff durch mehrere Pods und macht Dateien wie die Modellgewichte (`/models/gpunet-0.pth`) innerhalb des Containers für die Inferenz verfügbar. Dadurch wird sichergestellt, dass der Fallback-„Download“ in unserer Anwendung (`app.py`) nicht

ausgelöst wird, da die Datei über das Mount vorhanden ist. Durch die Entkopplung des Modells vom Container-Image werden der gemeinsame Zugriff und unabhängige Modellversionsaktualisierungen ohne Image-Neuerstellung ermöglicht.

Erstellen Sie den PersistentVolume (PV)

Erstellen Sie eine PersistentVolume (PV-) Ressource, um den S3-Bucket zu mounten, der Ihre Modellgewichte enthält, und ermöglichen Sie so den schreibgeschützten Zugriff für mehrere Pods, ohne dass Dateien zur Laufzeit heruntergeladen werden müssen:

```
cat <<EOF | envsubst | kubectl apply -f -
apiVersion: v1
kind: PersistentVolume
metadata:
  name: s3-model-pv
spec:
  capacity:
    storage: 5Gi  # Ignored by the driver; can be any value
  accessModes:
    - ReadOnlyMany  # Read only
  persistentVolumeReclaimPolicy: Retain
  storageClassName: ""  # Required for static provisioning
  claimRef:
    namespace: default  # Adjust if you prefer a different namespace
    name: s3-model-pvc
  mountOptions:
    - allow-other  # Enables multi-user access (useful for non-root pods)
    - region ${AWS_REGION} # Optional, include if your bucket is in a different region
 than the cluster
  csi:
    driver: s3.csi.aws.com
    volumeHandle: gpunet-model-volume  # Must be unique across all PVs
    volumeAttributes:
      bucketName: ${S3_BUCKET_NAME}
EOF
```

Erstellen Sie das PersistentVolumeClaim (PVC)

Erstellen Sie ein PersistentVolumeClaim (PVC), das an das PV gebunden werden soll, und fordern Sie schreibgeschützten Zugriff auf die Daten des bereitgestellten S3-Modells an:

```
cat <<EOF | envsubst | kubectl apply -f -
apiVersion: v1
```

```
kind: PersistentVolumeClaim
metadata:
  name: s3-model-pvc
spec:
  accessModes:
    - ReadOnlyMany
  storageClassName: ""  # Required for static provisioning
  resources:
    requests:
      storage: 5Gi  # Ignored, match PV capacity
  volumeName: s3-model-pv  # Bind to the PV created above
EOF
```

Bereitstellen der Anwendung

Stellen Sie die Inferenz-Anwendung als Kubernetes-Bereitstellung bereit, indem Sie das S3-gestützte persistente Volume für den Modellzugriff mounten, GPU-Knotenselektoren und Toleranzen anwenden und Umgebungsvariablen für den Modellpfad festlegen. Diese Bereitstellung legt den Modellpfad (Umgebungsvariable von "`/models/gpunet-0.pth`") fest, sodass unsere Anwendung (in `app.py`) standardmäßig diesen Pfad verwendet Wenn das Volume der Bereitstellung in `/models` (schreibgeschützt) eingebunden ist, wird der Modell-Download nicht ausgelöst, wenn die Datei bereits über das PVC vorhanden ist.

```
cat <<EOF | envsubst | kubectl apply -f -
apiVersion: apps/v1
kind: Deployment
metadata:
  name: gpunet-inference-app
spec:
  replicas: 1
  selector:
    matchLabels:
      app: gpunet-inference-app
  template:
    metadata:
      labels:
        app: gpunet-inference-app
    spec:
      tolerations:
      - key: "nvidia.com/gpu"
        operator: "Exists"
        effect: "NoSchedule"
```

```
      nodeSelector:
        role: gpu-worker
      containers:
      - name: inference
        image: ${AWS_ACCOUNT_ID}.dkr.ecr.${AWS_REGION}.amazonaws.com/gpunet-inference-
app:latest
        ports:
        - containerPort: 80
        env:
        - name: MODEL_PATH
          value: "/models/gpunet-0.pth"
        resources:
          limits:
            nvidia.com/gpu: 1
          requests:
            nvidia.com/gpu: 1
        volumeMounts:
        - name: model-volume
          mountPath: /models
          readOnly: true
      volumes:
      - name: model-volume
        persistentVolumeClaim:
          claimName: s3-model-pvc
EOF
```

Es kann einige Minuten dauern, bis Karpenter einen GPU-Knoten bereitstellt, falls noch keiner verfügbar ist. Überprüfen Sie, ob sich der Inferenz-Pod im Status „Ausgeführt“ befindet:

```
kubectl get pods -l app=gpunet-inference-app
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
NAME                                  READY   STATUS    RESTARTS   AGE
gpunet-inference-app-5d4b6c7f8-abcde        1/1     Running   0          2m
```

Service mit Ingress und Load Balancer verfügbar machen

Erstellen Sie einen ClusterIP- Service, um die Inferenz-Bereitstellung intern innerhalb des EKS-Clusters freizugeben, wobei Sie den Port der Anwendung als Ziel angeben:

```
cat <<EOF | envsubst | kubectl apply -f -
```

```
apiVersion: v1
kind: Service
metadata:
  name: gpunet-model-service
spec:
  type: ClusterIP
  ports:
  - port: 80
    targetPort: 80
  selector:
    app: gpunet-inference-app
EOF
```

Erstellen Sie eine Ingress-Ressource, um einen mit dem Internet verbundenen Application Load Balancer (ALB) über den AWS LBC bereitzustellen und externen Datenverkehr an den Inferenzdienst weiterzuleiten:

```
cat <<EOF | envsubst | kubectl apply -f -
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: gpunet-model-ingress
  annotations:
    alb.ingress.kubernetes.io/scheme: internet-facing
    alb.ingress.kubernetes.io/target-type: ip
spec:
  ingressClassName: alb
  rules:
  - http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: gpunet-model-service
            port:
              number: 80
EOF
```

Warten Sie einige Minuten, bis der Application Load Balancer (ALB) die Bereitstellung abgeschlossen hat. Überwachen Sie den Status der Ingress-Ressource, um sicherzustellen, dass der ALB bereitgestellt wurde:

```
kubectl get ingress gpunet-model-ingress
```

Die erwartete Ausgabe sollte wie folgt aussehen (mit ausgefülltem Feld ADRESSE):

```
NAME                   CLASS   HOSTS   ADDRESS
 PORTS   AGE
gpunet-model-ingress   alb     *       k8s-default-gpunetmo-183de3f819-516310036.us-
east-1.elb.amazonaws.com   80      6m58s
```

Extrahieren und exportieren Sie den ALB-Host-Namen aus dem Ingress-Status zur Verwendung in nachfolgenden Tests:

```
export ALB_HOSTNAME=$(kubectl get ingress gpunet-model-ingress -o
 jsonpath='{.status.loadBalancer.ingress[0].hostname}')
```

Modell-Service testen

Validieren Sie den freigegebenen Inferenzendpunkt, indem Sie eine POST-Anfrage mit einer Beispielbild-URL (z. B. aus dem COCO-Datensatz) senden und so eine Echtzeit-Prognose simulieren:

```
curl -X POST "http://${ALB_HOSTNAME}/predict?image_url=http://images.cocodataset.org/
test-stuff2017/000000024309.jpg"
```

Die erwartete Ausgabe sollte eine JSON-Antwort mit den fünf besten Prognosen sein, ähnlich wie hier (die tatsächlichen Bezeichnungen und Wahrscheinlichkeiten können je nach Image und Modellgenauigkeit leicht variieren):

```
{"predictions":[{"label":"desk","probability":0.2888975441455841},
{"label":"laptop","probability":0.2464350312948227},
{"label":"notebook","probability":0.08554483205080032},
{"label":"library","probability":0.030612602829933167},
{"label":"monitor","probability":0.02989672815084457}]}
```

Optional können Sie in einer neuen POST-Anfrage mit dem Testen anderer Images fortfahren. Beispiel:

```
http://images.cocodataset.org/test-stuff2017/000000024309.jpg
```

```
http://images.cocodataset.org/test-stuff2017/000000028117.jpg
http://images.cocodataset.org/test-stuff2017/000000006149.jpg
http://images.cocodataset.org/test-stuff2017/000000004954.jpg
```

Schlussfolgerung

In diesem Handbuch richten Sie einen Amazon-EKS-Cluster ein, der für GPU-beschleunigte Echtzeit-Inferenz-Workloads optimiert ist. Sie haben einen Cluster mit EC2 G5-Instanzen bereitgestellt, den Mountpoint S3 CSI-Treiber, den EKS Pod Identity Agent, den EKS Node Monitoring Agent, das Bottlerocket AMI, den Load AWS Balancer Controller (LBC) und Karpenter zur Verwaltung von CPU und GPU installiert. NodePools Sie haben das NVIDIA-Geräte-Plugin verwendet, um die GPU-Planung zu aktivieren, und S3 mit einem und für den Modellzugriff konfiguriert. PersistentVolume PersistentVolumeClaim Sie haben das Setup validiert, indem Sie einen Beispiel-GPU-Pod bereitgestellt, den Modellzugriff für das NVIDIA GPUNet-0-Modell auf Amazon S3 eingerichtet, die Pod-Initialisierung aktiviert und den Inferenzservice über Application Load Balancer verfügbar gemacht haben. Um Ihr Cluster optimal zu nutzen, konfigurieren Sie den EKS-Knoten-Überwachungsagent mit automatischer Reparatur. Führen Sie unbedingt Benchmark-Tests durch, einschließlich Bewertungen der GPU-Leistung, Latenz und des Durchsatzes, um die Reaktionszeiten zu optimieren. Weitere Informationen finden Sie unter Verwenden von Überwachungs- und Observability-Tools für Ihre Workloads. AI/ML

Bereinigen

Um future Gebühren zu vermeiden, müssen Sie den zugehörigen CloudFormation Stack manuell löschen, um alle in diesem Handbuch erstellten Ressourcen, einschließlich des VPC-Netzwerks, zu löschen.

Löschen Sie den CloudFormation Stack mithilfe des `--wait` Flags mit eksctl:

```
eksctl delete cluster --region ${AWS_REGION} --name ${EKS_CLUSTER_NAME} --wait
```

Nach Abschluss des Vorgangs sollte die folgende Antwort angezeigt werden:

```
2025-07-29 13:03:55 [#]  all cluster resources were deleted
```

Löschen Sie den in dieser Anleitung erstellten Amazon-S3-Bucket mithilfe der Amazon-S3-Konsole.

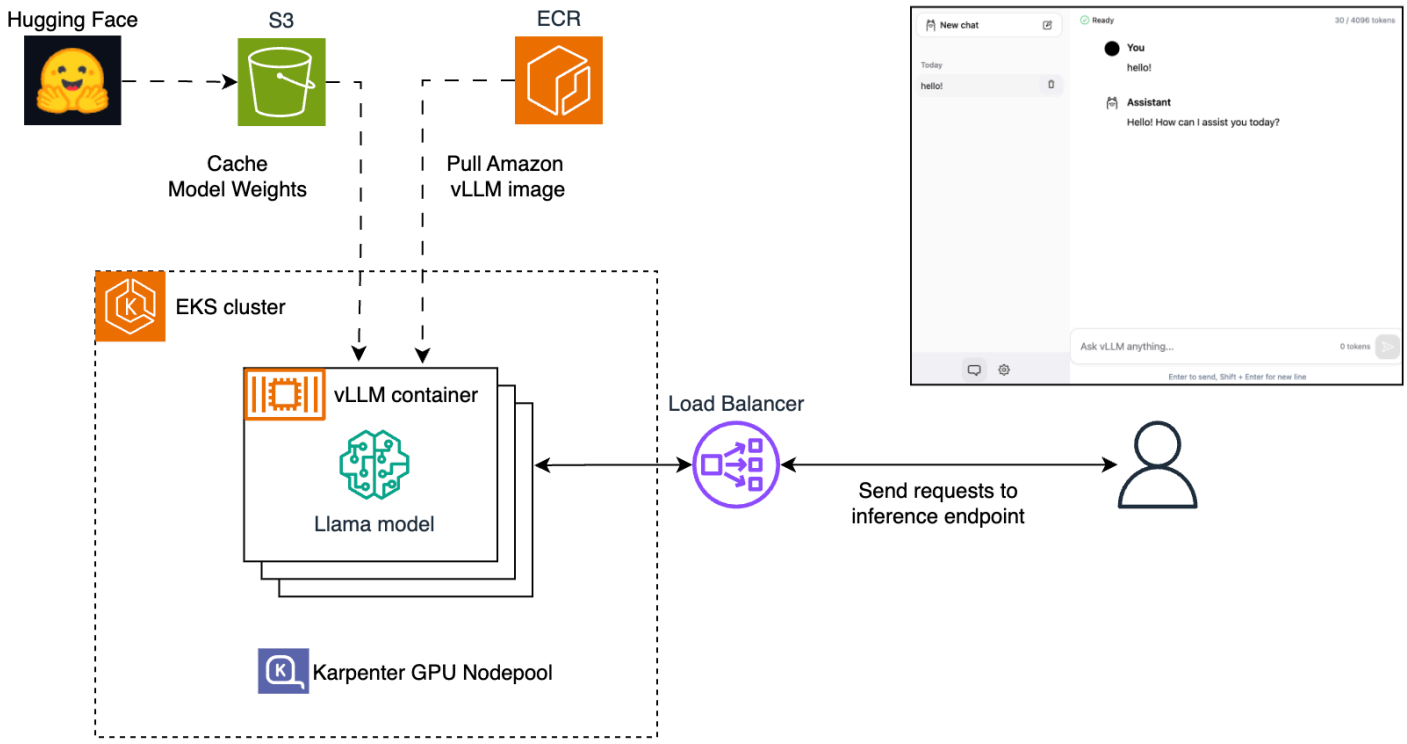
Schnellstart: LLM-Inferenz mit hohem Durchsatz mit vLLM auf Amazon EKS

Einführung

Diese Schnellstartanleitung bietet eine exemplarische Vorgehensweise für die Bereitstellung von Large Language Models (LLMs) auf Amazon EKS mithilfe von vLLM und GPUs für textbasierte Echtzeit-Inferenzanwendungen.

Die Lösung nutzt Amazon EKS für die Container-Orchestrierung und vLLM für effizientes Model Serving, sodass Sie skalierbare KI-Anwendungen mit GPU-Beschleunigung und Inferenz-Serving mit hohem Durchsatz erstellen können. Zu Demonstrationszwecken wird das Llama 3.1 8B Instruct-Modell verwendet, aber Sie können jedes andere LLM einsetzen, das von vLLM unterstützt wird (eine Liste der unterstützten Modelle finden Sie in der VllM-Dokumentation). Um die LLM-Inferenz zu testen, verwenden wir eine Chatbot-Beispielanwendung, die auf dem Projekt basiert. nextjs-vllm-ui Schließlich verwenden wir GuideLM, um die vLLM-Konfigurationsparameter zu vergleichen und zu optimieren, um die Inferenzleistung zu optimieren.

vLLM-Architektur auf EKS



Wenn Sie dieses Verfahren abgeschlossen haben, verfügen Sie über einen vLLM-Inferenzendpunkt, der für Durchsatz und geringe Latenz optimiert ist, und Sie können über eine Chat-Frontend-

Anwendung mit einem Lama-Modell interagieren, was einen typischen Anwendungsfall für Chatbot-Assistenten und andere LLM-basierte Anwendungen veranschaulicht.

Weitere Anleitungen und erweiterte Ressourcen zur Implementierung finden Sie in unserem EKS-Best-Practices-Leitfaden für AI/ML Workloads und produktionsreife KI in EKS-Inferenzdiagrammen.

Bevor Sie beginnen

Bevor Sie beginnen, stellen Sie sicher, dass Sie über Folgendes verfügen:

- Ein Amazon EKS-Cluster mit den folgenden Hauptkomponenten: Karpenter Nodepools mit der G5- oder EC2 G6-Instance-Familie, das auf Ihren GPU-fähigen Worker-Knoten installierte NVIDIA Device Plugin und der installierte S3 Mountpoint CSI-Treiber. Um dieses Basis-Setup zu erstellen, folgen Sie den Schritten unter bis zum Abschluss von Schritt #4. the section called “Cluster erstellen”
- Ein Hugging Face Face-Konto. Um sich zu registrieren, besuchen Sie https://huggingface.co/login.

Modellspeicher mit Amazon S3 einrichten

Speichern Sie große LLM-Dateien effizient in Amazon S3, um Speicher von Rechenressourcen zu trennen. Dieser Ansatz rationalisiert Modellaktualisierungen, senkt die Kosten und vereinfacht die Verwaltung von Produktionskonfigurationen. S3 verarbeitet riesige Dateien zuverlässig, und durch die Integration mit Kubernetes über den Mountpoint CSI-Treiber können Pods auf Modelle wie lokalen Speicher zugreifen — zeitaufwändige Downloads beim Start sind nicht erforderlich. Gehen Sie wie folgt vor, um einen S3-Bucket zu erstellen, ein LLM hochzuladen und es als Volume in Ihrem Inference Serving-Container zu mounten.

Auf EKS sind auch andere Speicherlösungen für das Modell-Caching verfügbar, z. B. EFS und FSx Lustre. Weitere Informationen finden Sie unter EKS Best Practices.

Festlegen von Umgebungsvariablen

Erstellen Sie einen eindeutigen Namen für einen neuen Amazon S3 S3-Bucket, den wir später in diesem Handbuch erstellen werden. Verwenden Sie nach der Erstellung für alle Schritte denselben Bucket-Namen. Beispiel:

```
MY_BUCKET_NAME=model-store-$(date +%s)
```

Definieren Sie Umgebungsvariablen und speichern Sie sie in einer Datei:

```
cat << EOF > .env-quickstart-vllm
export BUCKET_NAME=${MY_BUCKET_NAME}
export AWS_REGION=us-east-1
export AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
EOF
```

Laden Sie Umgebungsvariablen in Ihre Shell-Umgebung. Wenn Sie die aktuelle Shell-Umgebung schließen und eine neue öffnen, stellen Sie sicher, dass Sie die Umgebungsvariablen mit demselben Befehl neu beziehen:

```
source .env-quickstart-vllm
```

Erstellen Sie einen S3-Bucket zum Speichern von Modelldateien

Erstellen Sie einen S3-Bucket zum Speichern von Modelldateien:

```
aws s3 mb s3://${BUCKET_NAME} --region ${AWS_REGION}
```

Laden Sie das Modell von Hugging Face herunter

Hugging Face ist einer der wichtigsten Model-Hubs für den Zugriff auf LLM-Modelle. Um das Lama-Modell herunterzuladen, müssen Sie die Modelllizenz akzeptieren und die Token-Authentifizierung einrichten:

1. Akzeptieren Sie die Llama 3.1 8B Instruct-Modelllizenz unter https://huggingface. co/meta-llama/Llama-3.1-8B-Instruct.
2. Generieren Sie ein Zugriffstoken (gehen Sie zu Ihrem Profil > Einstellungen > Zugriffstoken und erstellen Sie dann ein neues Token mit dem Token-Typ Lesen).

Lege eine Umgebungsvariable mit deinem Hugging Face Face-Token fest:

```
export HF_TOKEN=your_token_here
```

Installieren Sie das Pip3-Paket, falls es nicht bereits in Ihrer Umgebung installiert ist. Beispielbefehl in Amazon Linux 2023:

```
sudo dnf install -y python3-pip
```

Installieren Sie die Hugging Face CLI:

```
pip install huggingface-hub
```

Laden Sie das Modell Llama-3.1-8B-Instruct von Hugging Face (~15 GB) herunter. Verwenden Sie dabei das `--exclude` Flag, um das alte PyTorch Format zu überspringen und nur die optimierten Dateien im Safetensors-Format herunterzuladen, wodurch die Download-Größe reduziert wird und gleichzeitig die volle Kompatibilität mit gängigen Inferenz-Engines erhalten bleibt:

```
huggingface-cli download meta-llama/Meta-Llama-3.1-8B-Instruct \
  --exclude "original/*" \
  --local-dir ./llama-3.1-8b-instruct \
  --token $HF_TOKEN
```

Überprüfen Sie die heruntergeladenen Dateien:

```
$ ls llama-3.1-8b-instruct
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
LICENSE        config.json                      model-00002-of-00004.safetensors
 model.safetensors.index.json  tokenizer_config.json
README.md      generation_config.json           model-00003-of-00004.safetensors
 special_tokens_map.json
USE_POLICY.md  model-00001-of-00004.safetensors  model-00004-of-00004.safetensors
 tokenizer.json
```

Laden Sie Modelldateien hoch

Aktivieren Sie AWS Common Runtime (CRT) für eine verbesserte S3-Übertragungsleistung. Der CRT-basierte Transferclient bietet verbesserten Durchsatz und Zuverlässigkeit bei Operationen mit großen Dateien:

```
aws configure set s3.preferred_transfer_client crt
```

Laden Sie das Modell hoch:

```
aws s3 cp ./llama-3.1-8b-instruct s3://$BUCKET_NAME/llama-3.1-8b-instruct \
  --recursive
```

Die erwartete Ausgabe sollte in etwa wie folgt aussehen:

```
...
upload: llama-3.1-8b-instruct/tokenizer.json to s3://model-store-1753EXAMPLE/
llama-3.1-8b-instruct/tokenizer.json
upload: llama-3.1-8b-instruct/model-00004-of-00004.safetensors to s3://model-
store-1753890326/llama-3.1-8b-instruct/model-00004-of-00004.safetensors
upload: llama-3.1-8b-instruct/model-00002-of-00004.safetensors to s3://model-
store-1753890326/llama-3.1-8b-instruct/model-00002-of-00004.safetensors
upload: llama-3.1-8b-instruct/model-00003-of-00004.safetensors to s3://model-
store-1753890326/llama-3.1-8b-instruct/model-00003-of-00004.safetensors
upload: llama-3.1-8b-instruct/model-00001-of-00004.safetensors to s3://model-
store-1753890326/llama-3.1-8b-instruct/model-00001-of-00004.safetensors
```

Richten Sie S3-Mountpoint-CSI-Berechtigungen ein

Der S3 Mountpoint CSI-Treiber ermöglicht die native Integration zwischen Kubernetes und S3, sodass Pods direkt auf Modelldateien zugreifen können, als wären sie lokaler Speicher, sodass beim Start des Containers keine lokalen Kopien erforderlich sind.

Erstellen Sie eine IAM-Richtlinie, damit der S3-Mountpoint aus Ihrem S3-Bucket lesen kann:

```
aws iam create-policy \
  --policy-name S3BucketAccess-${BUCKET_NAME} \
  --policy-document "{\"Version\": \"2012-10-17\", \"Statement\": [{\"Effect\":
 \"Allow\", \"Action\": [\"s3:GetObject\", \"s3:GetObjectVersion\", \"s3:ListBucket
\", \"s3:GetBucketLocation\"], \"Resource\": [\"arn:aws:s3:::${BUCKET_NAME}\",
 \"arn:aws:s3:::${BUCKET_NAME}/*\"]}]}"
```

Finden Sie den IAM-Rollennamen, der vom S3 Mountpoint CSI-Treiber verwendet wird, indem Sie die Anmerkungen zum S3 CSI-Treiber-Dienstkonto überprüfen:

```
ROLE_NAME=$(kubectl get serviceaccount s3-csi-driver-sa -n kube-system -o
 jsonpath='{.metadata.annotations.eks\.amazonaws\.com/role-arn}' | cut -d'/' -f2)
```

Fügen Sie Ihre IAM-Richtlinie der S3 Mountpoint CSI-Rolle hinzu:

```
aws iam attach-role-policy \
 --role-name ${ROLE_NAME} \
 --policy-arn arn:aws:iam::${AWS_ACCOUNT_ID}:policy/S3BucketAccess-${BUCKET_NAME}
```

Wenn S3 Mountpoint CSI nicht im Cluster installiert ist, folgen Sie den Bereitstellungsschritten unter. the section called “Cluster erstellen”

Mounten Sie den S3-Bucket als Kubernetes-Volume

Erstellen Sie ein Persistent Volume (PV) und einen Persistent Volume Claim (PVC), um über mehrere Inferenz-Pods hinweg schreibgeschützten Zugriff auf den S3-Bucket zu ermöglichen. Der ReadOnlyMany Zugriffsmodus gewährleistet den gleichzeitigen Zugriff auf Modelldateien, während der CSI-Treiber die Installation des S3-Buckets übernimmt:

```
cat <<EOF | envsubst | kubectl apply -f -
apiVersion: v1
kind: PersistentVolume
metadata:
  name: model-store
spec:
  storageClassName: ""
  capacity:
    storage: 100Gi
  accessModes:
    - ReadOnlyMany
  persistentVolumeReclaimPolicy: Retain
  mountOptions:
    - region ${AWS_REGION}
  csi:
    driver: s3.csi.aws.com
    volumeHandle: model-store
    volumeAttributes:
      bucketName: ${BUCKET_NAME}
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: model-store
spec:
  storageClassName: ""
  volumeName: model-store
  accessModes:
    - ReadOnlyMany
  resources:
    requests:
      storage: 100Gi
EOF
```

Einrichtung der GPU-Infrastruktur

Cluster-Knoten

Wir verwenden den EKS-Cluster, der in erstellt wurdethe section called “Cluster erstellen”. Dieser Cluster umfasst Karpenter-Knotenpools, die GPU-fähige Knoten mit ausreichend Knotenspeicher zum Herunterladen des vLLM-Container-Images bereitstellen können. Wenn Sie Ihren benutzerdefinierten EKS-Cluster verwenden, stellen Sie sicher, dass dieser GPU-fähige Knoten starten kann.

Auswahl der Instance

Für die richtige Instanzauswahl für die LLM-Inferenz muss sichergestellt werden, dass der verfügbare GPU-Speicher ausreicht, um Modellgewichte zu laden. Die Modellgewichte für Llama 3.1 8B Instruct betragen ungefähr 16 GB (Größe der Modelldateien .safetensor), daher müssen wir dem vllm-Prozess mindestens diese Menge an Speicher zur Verfügung stellen, um das Modell zu laden.

Amazon EC2 G5-Instances mit A10G GPUs und EC2 G6-Instances mit L4 bieten GPUs beide 24 GB VRAM pro GPU, ausreichend für das Laden von Llama 3.1 8B Instruct-Gewichten. Wenn Sie ein Modell mit größeren Gewichtungen bereitstellen, sollten Sie die Verwendung eines Instance-Typs mit mehreren GPUs oder eines Setups mit mehreren Knoten in Betracht ziehen.

NVIDIA-Gerätetreiber

NVIDIA-Treiber bieten die erforderliche Laufzeitumgebung für Container, um effizient auf GPU-Ressourcen zugreifen zu können. Sie ermöglicht die Zuweisung und Verwaltung von GPU-Ressourcen innerhalb von Kubernetes und stellt sie als planbare Ressourcen GPUs zur Verfügung.

Unser Cluster verwendet EKS Bottlerocket AMIs, das alle erforderlichen NVIDIA-Gerätetreiber und -Plugins auf allen GPU-fähigen Knoten enthält und so den sofortigen GPU-Zugriff für containerisierte Workloads ohne zusätzliche Einrichtung gewährleistet. Wenn Sie andere Arten von EKS-Knoten verwenden, müssen Sie sicherstellen, dass alle erforderlichen Treiber und Plugins installiert sind.

Testen Sie die GPU-Infrastruktur

Testen Sie die GPU-Fähigkeiten Ihres Clusters, indem Sie die folgenden Schritte ausführen, um sicherzustellen, dass Pods auf NVIDIA-GPU-Ressourcen zugreifen und auf GPU-fähigen Knoten korrekt planen können.

Stellen Sie einen Nvidia SMI Test-Pod bereit:

```
cat <<EOF | envsubst | kubectl apply -f -
```

```
apiVersion: v1
kind: Pod
metadata:
  name: gpu-nvidia-smi-test
spec:
  restartPolicy: OnFailure
  tolerations:
  - key: "nvidia.com/gpu"
    operator: "Exists"
    effect: "NoSchedule"
  nodeSelector:
    role: gpu-worker  # Matches GPU NodePool's label
  containers:
  - name: cuda-container
    image: nvidia/cuda:12.9.1-base-ubuntu20.04
    command: ["nvidia-smi"]
    resources:
      requests:
        memory: "24Gi"
      limits:
        nvidia.com/gpu: 1
EOF
```

Überprüfe die Pod-Logs, um zu überprüfen, ob GPU-Details aufgeführt sind, ähnlich wie in der folgenden Ausgabe (nicht unbedingt dasselbe GPU-Modell):

```
$ kubectl wait --for=jsonpath='{.status.phase}'=Succeeded pod/gpu-nvidia-smi-test
$ kubectl logs gpu-nvidia-smi-test
```

```
Wed Jul 30 15:39:58 2025
+-----------------------------------------------------------------------------------------
+
| NVIDIA-SMI 570.172.08             Driver Version: 570.172.08     CUDA Version: 12.9
   |
|-----------------------------------------+------------------------
+----------------------+
| GPU  Name                 Persistence-M | Bus-Id          Disp.A | Volatile Uncorr.
 ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |           Memory-Usage | GPU-Util  Compute
 M. |
|                                         |                        |               MIG
 M. |
```

```
|=========================================+========================
+======================|
|   0  NVIDIA A10G                    On  |   00000000:00:1E.0 Off |
 0 |
|  0%   30C    P8              9W /  300W |       0MiB /  23028MiB |      0%
 Default |
|                                         |                        |
 N/A |
+-----------------------------------------+------------------------
+----------------------+

+---------------------------------------------------------------------------------------
+
| Processes:
   |
|  GPU   GI   CI              PID   Type   Process name                        GPU
 Memory |
|        ID   ID                                                               Usage
   |
|
=======================================================================================|
|  No running processes found
   |
+---------------------------------------------------------------------------------------
+
```

Diese Ausgabe zeigt, dass Pods erfolgreich auf GPU-Ressourcen zugreifen können.

WICHTIG: Dieser Pod verwendet eine NodeSelector-Konfiguration, die auf die Karpenter-Knotenpools abgestimmt ist. the section called “Cluster erstellen” Wenn Sie verschiedene Knotenpools verwenden, stellen Sie sicher, dass der Pod entsprechend mit NodeSelector und Tolerations übereinstimmt.

Stellen Sie Inference Container bereit

Der Serving-Stack bestimmt sowohl die Leistung als auch die Skalierbarkeit Ihrer Inferenzinfrastruktur. vLLM hat sich zu einer führenden Lösung für Produktionsbereitstellungen entwickelt. Die Architektur von vLLM bietet kontinuierliches Batching für dynamische Anforderungsverarbeitung, Kernel-Optimierungen für schnellere Inferenzen und eine effiziente GPU-Speicherverwaltung. PagedAttention Diese Funktionen, kombiniert mit einer produktionsbereiten REST-API und der Unterstützung gängiger Modellformate, machen es zu einer optimalen Wahl für leistungsstarke Inferenzbereitstellungen.

Wählen Sie das Deep Learning-Container-Image AWS

AWS Deep Learning Containers (DLCs) bieten voroptimierte Umgebungen mit Sicherheitsupdates, AWS Infrastrukturkompatibilität und optimierten Treiberkonfigurationen. Dies reduziert die Komplexität der Bereitstellung und den Wartungsaufwand und stellt gleichzeitig die Produktionsbereitschaft sicher.

Für diese Bereitstellung verwenden wir den AWS DLC für vLLM 0.9, der Nvidia-Bibliotheken und optimierte GPU-Leistungskonfigurationen enthält, die speziell auf die Inferenz von Transformatormodellen auf GPU-Instanzen abgestimmt sind. AWS

```
image: 763104351884.dkr.ecr.us-east-1.amazonaws.com/vllm:0.9-gpu-py312-ec2
```

Wenden Sie die vLLM-Kubernetes-Manifeste an

Es gibt mehrere Möglichkeiten, vLLM in EKS bereitzustellen. In diesem Handbuch wird die vLLM-Bereitstellung mithilfe einer Kubernetes-Bereitstellung demonstriert. Dabei handelt es sich um eine Kubernetes-native und einfache Methode für den Einstieg. Erweiterte Bereitstellungsoptionen finden Sie in den vLLM-Dokumenten und KI-on-EKS-Blueprints.

Definieren Sie Bereitstellungsparameter mithilfe von Kubernetes-Manifesten, um die Ressourcenzuweisung, die Platzierung von Knoten, Integritätstests, die Bereitstellung des Dienstes usw. zu kontrollieren. Konfigurieren Sie Ihre Bereitstellung so, dass ein GPU-fähiger Pod mithilfe AWS des Deep Learning-Container-Images für vLLM ausgeführt wird. Legen Sie optimierte Parameter für die LLM-Inferenz fest und machen Sie den VLLM-OpenAPI-kompatiblen Endpunkt über den Load Balancer-Dienst verfügbar: AWS

```
cat <<EOF | envsubst | kubectl apply -f -
apiVersion: apps/v1
kind: Deployment
metadata:
  name: vllm-inference-app
spec:
  replicas: 1
  selector:
    matchLabels:
      app: vllm-inference-app
  template:
    metadata:
      labels:
        app: vllm-inference-app
```

```
    spec:
      tolerations:
      - key: "nvidia.com/gpu"
        operator: "Exists"
        effect: "NoSchedule"
      nodeSelector:
        role: gpu-worker
      containers:
      - name: vllm-inference
        image: 763104351884.dkr.ecr.us-east-1.amazonaws.com/vllm:0.9-gpu-py312-ec2
        ports:
        - containerPort: 8000
        env:
        - name: MODEL_PATH
          value: "/mnt/models/llama-3.1-8b-instruct"
        args:
        - "--model=/mnt/models/llama-3.1-8b-instruct"
        - "--host=0.0.0.0"
        - "--port=8000"
        - "--tensor-parallel-size=1"
        - "--gpu-memory-utilization=0.9"
        - "--max-model-len=8192"
        - "--max-num-seqs=1"
        readinessProbe:
          httpGet:
            path: /health
            port: 8000
          initialDelaySeconds: 30
          periodSeconds: 5
          timeoutSeconds: 10
        resources:
          limits:
            nvidia.com/gpu: 1
          requests:
            memory: "24Gi"
            cpu: "4"
            ephemeral-storage: "25Gi"  # Ensure enough node storage for vLLM container
image
        volumeMounts:
        - name: models
          mountPath: /mnt/models
          readOnly: true
      volumes:
      - name: models
```

```
        persistentVolumeClaim:
          claimName: model-store
---
apiVersion: v1
kind: Service
metadata:
  name: vllm-inference-svc
  annotations:
    service.beta.kubernetes.io/aws-load-balancer-type: nlb
    service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing
spec:
  type: LoadBalancer
  ports:
    - port: 80
      targetPort: 8000
      protocol: TCP
  selector:
    app: vllm-inference-app
EOF
```

Überprüfen Sie, ob sich der vLLM-Pod im folgenden Zustand befindet: `Ready 1/1`

```
kubectl get pod -l app=vllm-inference-app -w
```

Erwartete Ausgabe:

```
NAME                                  READY   UP-TO-DATE   AVAILABLE   AGE
vllm-inference-app-65df5fddc8-5kmjm   1/1     1            1           5m
```

Es kann mehrere Minuten dauern, bis das Container-Image abgerufen wird und vLLM Modelldateien in den GPU-Speicher lädt. Fahren Sie erst fort, wenn der Pod bereit und verfügbar ist.

Machen Sie den Dienst verfügbar

Stellen Sie den Inferenzendpunkt lokal über die Kubernetes-Portweiterleitung zur lokalen Entwicklung und zum Testen bereit. Lassen Sie diesen Befehl in einem separaten Terminalfenster laufen:

```
export POD_NAME=$(kubectl get pod -l app=vllm-inference-app -o
 jsonpath='{.items[0].metadata.name}')
kubectl port-forward pod/$POD_NAME 8000:8000
```

Der Load AWS Balancer Controller erstellt automatisch einen Network Load Balancer, der den vLLM-Dienstendpunkt extern verfügbar macht. Rufen Sie den NLB-Endpunkt ab, indem Sie Folgendes ausführen:

```
NLB=$(kubectl get service vllm-inference-svc -o
 jsonpath='{.status.loadBalancer.ingress[0].hostname}')
```

Müssen Sie den AWS Load Balancer Controller installieren? Folgen Sie den Bereitstellungsschritten unterthe section called “ AWS Load Balancer Balancer-Controller”.

Bilden von Inferenzen

Überprüfen Sie den Inferenz-Pod

Überprüfen Sie die Funktionalität des Inferenzcontainers lokal über den weitergeleiteten Port. Senden Sie eine Verbindungsanforderung und stellen Sie sicher, dass die Antwort den HTTP-Code 200 enthält:

```
$ curl -IX GET "http://localhost:8000/v1/models"
```

```
HTTP/1.1 200 OK
date: Mon, 13 Oct 2025 23:24:57 GMT
server: uvicorn
content-length: 516
content-type: application/json
```

Testen Sie die Inferenzfunktionen und validieren Sie die externe Konnektivität, indem Sie eine Abschlussanfrage über den NLB-Endpunkt an das LLM senden:

```
curl -X POST "http://$NLB:80/v1/completions" \
  -H "Content-Type: application/json" \
  -d '{
    "model": "/mnt/models/llama-3.1-8b-instruct",
    "prompt": "Explain artificial intelligence:",
    "max_tokens": 512,
    "temperature": 0.7
  }'
```

Dieser Endpunkt folgt dem OpenAI-API-Format, wodurch er mit bestehenden Anwendungen kompatibel ist und gleichzeitig konfigurierbare Generierungsparameter wie Antwortlänge und Temperatur zur Steuerung der Ausgangsdiversität bereitstellt.

Führen Sie die Chatbot-App aus

Zur Veranschaulichung wird in diesem Handbuch eine Chatbot-Beispielanwendung ausgeführt, die Project verwendet nextjs-vllm-ui, um Benutzerinteraktionen mit dem Modell zu veranschaulichen.

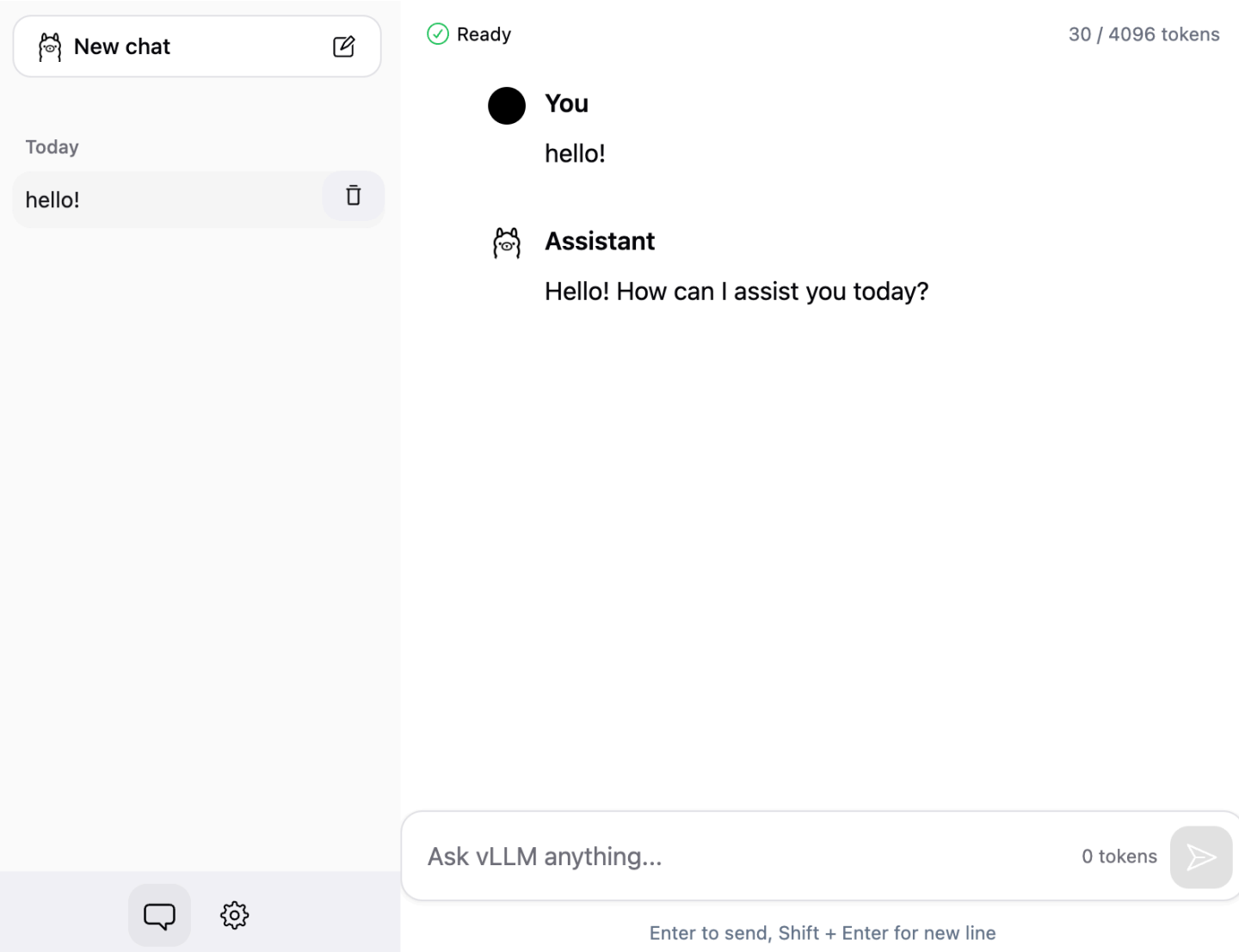
Führen Sie eine Chatbot-Benutzeroberfläche als Docker-Container aus, der den Port 3000 dem Localhost zuordnet und eine Verbindung zum vLLM-NLB-Endpunkt herstellt:

```
docker run --rm \
  -p 3000:3000 \
  -e VLLM_URL="http://${NLB}:80" \
  --name nextjs-vllm-ui-demo \
  ghcr.io/yoziru/nextjs-vllm-ui:latest
```

Öffnen Sie Ihren Webbrowser und navigieren Sie zu: http://localhost:3000/

Sie sollten die Chat-Oberfläche sehen, über die Sie mit dem Lama-Modell interagieren können.

Chat-Benutzeroberfläche



Optimieren Sie die Leistung von Inferenzen

Spezialisierte Inferenz-Engines wie vLLM bieten erweiterte Funktionen, die die Inferenzleistung erheblich steigern, darunter kontinuierliches Batching, effizientes KV-Caching und optimierte Memory-Aufmerksamkeitsmechanismen. Sie können die vLLM-Konfigurationsparameter anpassen, um die Inferenzleistung zu verbessern und gleichzeitig Ihre spezifischen Anwendungsfallanforderungen und Workload-Muster zu erfüllen. Die richtige Konfiguration ist unerlässlich, um eine GPU-Sättigung zu erreichen. Sie stellt sicher, dass Sie den größtmöglichen Nutzen aus teuren GPU-Ressourcen ziehen und gleichzeitig einen hohen Durchsatz, geringe Latenz und einen kostengünstigen Betrieb bieten. Die folgenden Optimierungen helfen Ihnen dabei, die Leistung Ihrer vLLM-Bereitstellung auf EKS zu maximieren.

Benchmark-vLLM-Konfigurationen

Um die vLLM-Konfigurationsparameter an Ihren Anwendungsfall anzupassen, vergleichen Sie verschiedene Einstellungen mit einem umfassenden Inferenz-Benchmarking-Tool wie GuideLM. Dabei werden wichtige Kennzahlen wie der Durchsatz von Anfragen pro Sekunde (RPS), end-to-end Latenz (E2E), Time to First Token (TTFT) und Tail-Latenz (TPOT) erfasst, um verschiedene Konfigurationen zu vergleichen.

Basiskonfiguration von vLLM

Dies ist die Basiskonfiguration, die für die Ausführung von vLLM verwendet wurde:

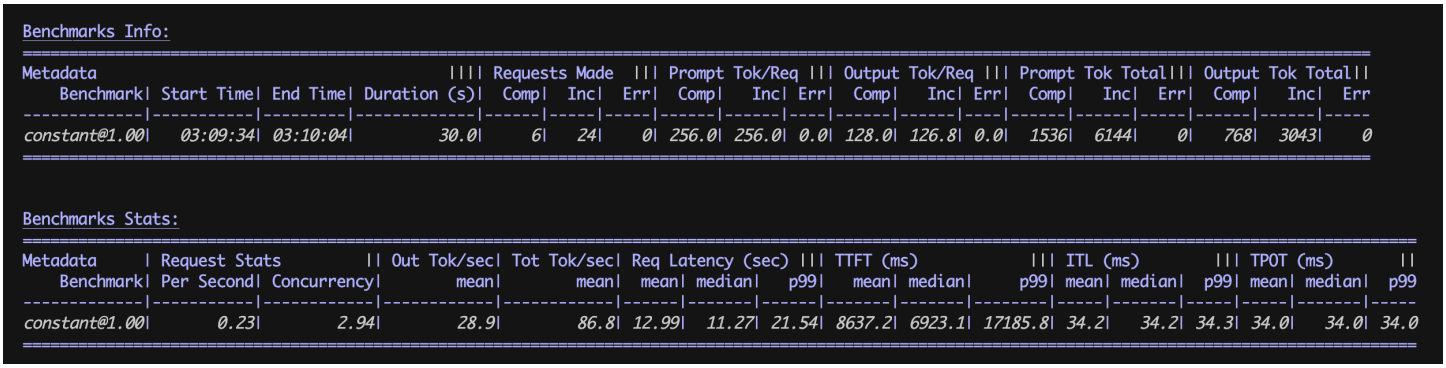
vLLM-Parameter	Description
tensor_parallel_size: 1	Verteilen Sie das Modell auf 1 GPU
gpu_memory_utilization: 0,90	Reservieren Sie 10% GPU-Speicher für System-Overhead
max_sequence_length: 8192	Maximale Gesamtlänge der Sequenz (Eingabe + Ausgabe)
max_num_seqs: 1	Maximale Anzahl gleichzeitiger Anfragen pro GPU (Batching)

Führen Sie GuideLM mit diesem Basis-Setup aus, um eine Leistungsbasis festzulegen. Für diesen Test ist GuideLLM so konfiguriert, dass es 1 Anfrage pro Sekunde generiert, mit 256 Token-Anfragen und 128-Token-Antworten.

```
guidellm benchmark \
--target "http://${NLB}:80" \
--processor meta-llama/Llama-3.1-8B-Instruct \
--rate-type constant \
--rate 1 \
--max-seconds 30 \
--data "prompt_tokens=256,output_tokens=128"
```

Erwartete Ausgabe:

Benchmark-Baseline-Ergebnisse



```
Benchmarks Info:
═══════════════════════════════════════════════════════════════════════════════════════════════════════════════════════════════
Metadata                                  |||| Requests Made  ||| Prompt Tok/Req ||| Output Tok/Req ||| Prompt Tok Total||| Output Tok Total||
    Benchmark| Start Time| End Time| Duration (s)|  Comp|  Inc|  Err|  Comp|   Inc| Err|  Comp|   Inc| Err|  Comp|   Inc|  Err|  Comp|   Inc|  Err
-------------|-----------|---------|-------------|------|-----|-----|------|------|----|------|------|----|------|------|-----|------|------|-----
constant@1.00|   03:09:34| 03:10:04|         30.0|     6|   24|    0| 256.0| 256.0| 0.0| 128.0| 126.8| 0.0|  1536|  6144|    0|   768|  3043|    0
═══════════════════════════════════════════════════════════════════════════════════════════════════════════════════════════════


Benchmarks Stats:
═══════════════════════════════════════════════════════════════════════════════════════════════════════════════════════════════
Metadata     | Request Stats         || Out Tok/sec| Tot Tok/sec| Req Latency (sec) ||| TTFT (ms)           ||| ITL (ms)        ||| TPOT (ms)       ||
    Benchmark| Per Second| Concurrency|       mean|        mean|  mean| median|   p99|   mean| median|     p99| mean| median|  p99| mean| median|  p99
-------------|-----------|------------|-----------|------------|------|-------|------|-------|-------|--------|-----|-------|-----|-----|-------|-----
constant@1.00|       0.23|        2.94|       28.9|        86.8| 12.99|  11.27| 21.54| 8637.2| 6923.1| 17185.8| 34.2|   34.2| 34.3| 34.0|   34.0| 34.0
═══════════════════════════════════════════════════════════════════════════════════════════════════════════════════════════════
```

Optimierte vLLM-Konfiguration

Passen Sie die vLLM-Parameter an, um die GPU-Ressourcen und die Parallelisierung besser zu nutzen:

vLLM-Parameter	Description
tensor_parallel_size: 1	Bleib bei 1 GPU. Die Tensor-Parallelisierung muss der Anzahl von entsprechen, die von vLLM GPUs verwendet werden soll
gpu_memory_utilization: 0.92	Reduzieren Sie nach Möglichkeit den Overhead des GPU-Speic hers und stellen Sie gleichzeitig sicher, dass vLLM weiterhin fehlerfrei läuft
max_sequence_length: 4096	Passen Sie die maximale Sequenz an die Anforderungen Ihres Anwendungsfalls an. Eine niedrigere maximale Sequenz setzt Ressourcen frei, die für eine erhöhte Parallelisierung verwendet werden können
max_num_seqs: 8	Eine Erhöhung der maximalen Sequenz erhöht den Durchsatz, erhöht aber auch die Latenz. Erhöhen Sie diesen Wert, um den Durchsatz zu maximieren und gleichzeitig sicherzustellen, dass die Latenz Ihren Anwendungsfallanforderungen entspricht

Wenden Sie diese Änderungen mit dem Befehl kubectl patch auf die laufende Bereitstellung an:

```
kubectl patch deployment vllm-inference-app --type='json' -p='[
  {"op": "replace", "path": "/spec/template/spec/containers/0/args/4", "value": "--gpu-
memory-utilization=0.92"},
```

```
  {"op": "replace", "path": "/spec/template/spec/containers/0/args/5", "value": "--max-
model-len=4096"},
  {"op": "replace", "path": "/spec/template/spec/containers/0/args/6", "value": "--max-
num-seqs=8"}
]'
```

Überprüfen Sie, ob sich der vLLM-Pod im folgenden Zustand befindet: `Ready 1/1`

```
kubectl get pod -l app=vllm-inference-app -w
```

Erwartete Ausgabe:

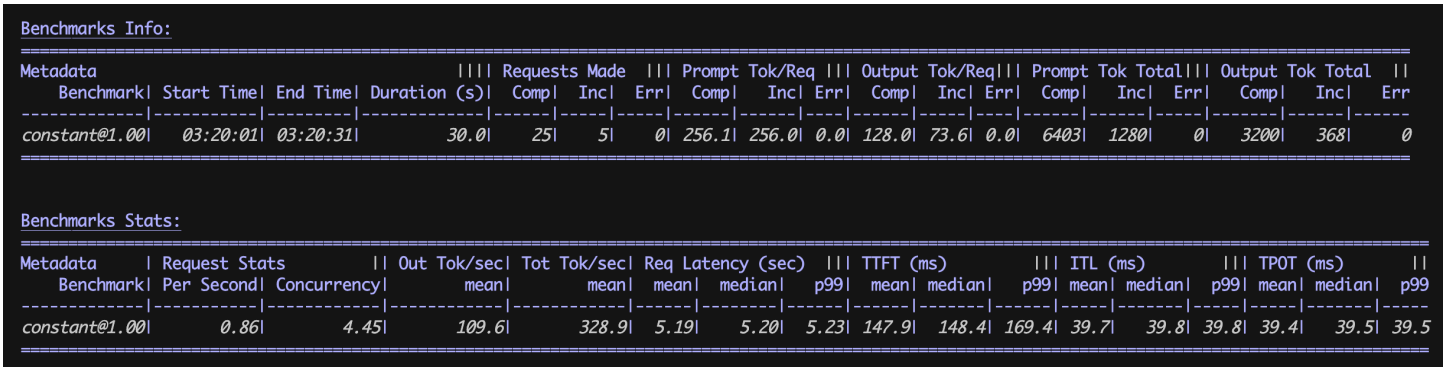
```
NAME                                    READY   UP-TO-DATE   AVAILABLE   AGE
vllm-inference-app-65df5fddc8-5kmjm   1/1     1            1           5m
```

Führen Sie dann GuideLM erneut aus und verwenden Sie dabei dieselben Benchmarking-Werte wie zuvor:

```
guidellm benchmark \
--target "http://${NLB}:80" \
--processor meta-llama/Llama-3.1-8B-Instruct \
--rate-type constant \
--rate 1 \
--max-seconds 30 \
--data "prompt_tokens=256,output_tokens=128"
```

Erwartete Ausgabe:

Optimierte Benchmark-Ergebnisse



Benchmarking-Ergebnisse

Berechnen Sie die Benchmarking-Ergebnisse in einer Tabelle sowohl für die Basiskonfiguration als auch für die optimierte vLLM-Konfiguration:

Durchschnittswerte	Basiskonfiguration	Optimierte Konfiguration
RPS	0,23 Anfragen/Sek	0,86 Anforderungen/Sekunde
E2E	12,99 s	5,19 s
TTFT	8637,2 ms	147,9 ms
TOPF	34,0 ms	39,5 ms

Die optimierten vLLM-Konfigurationen verbesserten den Inferenzdurchsatz (RPS) erheblich und reduzierten die Latenz (E2E, TTFT) bei nur einer geringfügigen Erhöhung der Tail-Latenz (TPOT) um nur eine Millisekunde. Diese Ergebnisse zeigen, wie vLLM die Inferenzleistung erheblich verbessert, sodass jeder Container mehr Anfragen in kürzerer Zeit verarbeiten kann, was einen kostengünstigen Betrieb ermöglicht.

Amazon EKS-Cluster-Konfiguration für AI/ML Workloads

Dieser Abschnitt soll Ihnen bei der Konfiguration von Amazon EKS-Clustern helfen, die für AI/ML Workloads optimiert sind. Sie finden Anleitungen zur Ausführung von GPU-beschleunigten Containern mit optimiertem Linux- und Windows-Betriebssystem AMIs, zur Einrichtung von Trainingsclustern mit Elastic Fabric Adapter (EFA) für Hochleistungsnetzwerke und zur Erstellung von Inferenzclustern mit AWS Inferentia-Instances, einschließlich Voraussetzungen, Verfahren und Überlegungen zur Bereitstellung. step-by-step

Themen

- Verwenden Sie EKS-optimierte beschleunigte AMIs GPU-Instanzen
- Installieren Sie das Kubernetes-Geräte-Plugin für GPUs
- Ausführung von GPU-beschleunigten Containern (Windows in EC2-G-Serie)
- Ausführung von Machine-Learning-Trainings in Amazon EKS mit Elastic Fabric Adapter
- Verwenden Sie AWS Inferentia-Instances mit Amazon EKS für Machine Learning

Verwenden Sie EKS-optimierte beschleunigte AMIs GPU-Instanzen

Amazon EKS unterstützt EKS-optimiertes Amazon Linux und AMIs Bottlerocket für GPU-Instances. Die EKS-optimierten Accelerated AMIs vereinfachen die Ausführung von KI- und ML-Workloads in EKS-Clustern, indem sie vorgefertigte, validierte Betriebssystem-Images für den beschleunigten Kubernetes-Stack bereitstellen. Zusätzlich zu den Kernkomponenten von Kubernetes, die im Standard EKS-Optimized enthalten sind, enthalten die EKS-optimierten Accelerated die Kernelmodule und Treiber AMIs, die für den Betrieb der NVIDIA-GPU und -Instanzen sowie der GPU G Inferentia- und P EC2 Trainium-Instances in EKS-Clustern erforderlich AMIs sind. AWS EC2

Die folgende Tabelle zeigt die unterstützten GPU-Instance-Typen für jede EKS-optimierte beschleunigte AMI-Variante. Die neuesten Updates der AMI-Varianten finden Sie in den EKS-optimierten AL2023-Versionen und Bottlerocket-Versionen unter GitHub .

EKS AMI-Variante	EC2 Instanztypen
AL2023 x86_64 NVIDIA	p6-b300, p6-b200, p5, p5e, p5en, p4d, p4de, p3, p3dn, gr6, g6, g6e, g6f, gr6f, g6f, g6f, g6f, g6f, g5dn
AL2023 ARM NVIDIA	p6e-gb200, 5g
AL2023 x86_64 Neuron	inf1, inf2, trn1, trn2
Flaschenrakete x86_64 aws-k8s-nvidia	p6-b300, p6-b200, p5, p5e, p5en, p4d, p4de, p3, p3dn, gr6, g6e, g6f, gr6f, g6f, g6f, g6f, g5, g4dn
Flaschenrakete aarch64/arm64 aws-k8s-nvidia	g 5 g
Flaschenrakete x86_64 aws-k8s	inf1, inf2, trn1, trn2

EKS-optimiertes NVIDIA AMIs

Durch die Verwendung von EKS-optimiertem NVIDIA AMIs stimmen Sie der Cloud-Endbenutzer-Lizenzvereinbarung (EULA) von NVIDIA zu.

Die neueste EKS-optimierte NVIDIA-Version finden Sie unter und. AMIs the section called “Holen Sie sich die neuesten IDs” the section called “Aktuelle IDs abrufen”

Wenn Sie Amazon Elastic Fabric Adaptor (EFA) mit dem EKS-optimierten AL2 023 oder Bottlerocket NVIDIA verwenden AMIs, müssen Sie das EFA-Geräte-Plugin separat installieren. Weitere Informationen finden Sie unter the section called “Trainings-Cluster mit EFA einrichten”.

AL2EKS 023 NVIDIA AMIs

Wenn Sie den NVIDIA-GPU-Operator mit der EKS-optimierten AL2 023 NVIDIA verwenden AMIs, müssen Sie die Installation des Treibers und des Toolkits durch den Operator deaktivieren, da diese bereits im EKS enthalten sind. AMIs Die für EKS optimierten AL2 023 NVIDIA enthalten AMIs weder das NVIDIA Kubernetes-Geräte-Plugin noch den NVIDIA DRA-Treiber, und diese müssen separat installiert werden. Weitere Informationen finden Sie unter the section called “Installieren Sie das NVIDIA Kubernetes-Geräte-Plugin”.

Zusätzlich zu den standardmäßigen EKS AMI-Komponenten AMIs umfasst die EKS-optimierte AL2 023 NVIDIA die folgenden Komponenten.

- NVIDIA-Treiber
- NVIDIA CUDA-Benutzermodus-Treiber
- NVIDIA-Container-Toolkit
- NVIDIA-Fabric-Manager
- NVIDIA blieb hartnäckig
- NVIDIA IMEX-Treiber
- NVLink NVIDIA-Subnetzmanager
- EFA minimal (Kernelmodul und RDMA-Core)

Einzelheiten zum NVIDIA CUDA-Benutzermodus-Treiber und zum CUDA, der in Anwendungscontainern runtime/libraries verwendet wird, finden Sie in der NVIDIA-Dokumentation. Bei der von angezeigten CUDA-Version `nvidia-smi` handelt es sich um die Version des NVIDIA CUDA-Benutzermodus-Treibers, der auf dem Host installiert ist und mit dem in Anwendungscontainern verwendeten CUDA runtime/libraries kompatibel sein muss.

Die für EKS optimierte Version AL2 023 NVIDIA AMIs unterstützt den Kernel 6.12 für die Kubernetes-Versionen 1.33 und höher und die NVIDIA-Treiberversion 580 für alle Kubernetes-Versionen. Für die Verwendung von CUDA 13+ ist der NVIDIA 580-Treiber erforderlich.

Einzelheiten zu den Komponentenversionen, die in der enthalten sind, finden Sie in den EKS-optimierten AL2 023-Versionen unter GitHub . AMIs Einzelheiten zur AMIs Konfiguration der NVIDIA-

Abhängigkeiten durch das EKS AL2 023 NVIDIA AMI-Installationsskript und das Kernelladeskript finden Sie im EKS 023 NVIDIA AMI-Installationsskript. Sie können die Liste der installierten Pakete und ihrer Versionen auf einer laufenden EC2 Instance mit dem `dnf list installed` Befehl finden.

Wenn Sie eine benutzerdefinierte Version AMIs mit EKS-Optimized AMIs als Basis erstellen, wird die Ausführung eines Betriebssystem-Upgrades nicht empfohlen oder unterstützt (z. `dnf upgrade`) oder eines der Kubernetes- oder GPU-Pakete aktualisieren, die im EKS-optimierten Paket enthalten sind AMIs, da dadurch die Komponentenkompatibilität beeinträchtigt werden kann. Wenn Sie das Betriebssystem oder die Pakete, die in EKS-Optimized enthalten sind, aktualisieren, wird empfohlen AMIs, vor der Bereitstellung in der Produktion gründliche Tests in einer Entwicklungs- oder Staging-Umgebung durchzuführen.

Bei der Erstellung benutzerdefinierter GPU-Instanzen empfiehlt es sich, AMIs AMIs für jeden Instance-Typ, jede Generation und Familie, die Sie ausführen möchten, separate benutzerdefinierte Instances zu erstellen. Die für EKS optimierten beschleunigten AMIs Systeme installieren Treiber und Pakete selektiv zur Laufzeit, basierend auf der Generation und Familie des zugrunde liegenden Instance-Typs. Weitere Informationen finden Sie in den EKS AMI-Skripts für Installation und Laufzeit.

EKS Bottlerocket NVIDIA AMIs

Wenn Sie den NVIDIA-GPU-Operator mit dem EKS-optimierten Bottlerocket NVIDIA verwenden AMIs, müssen Sie die Operator-Installation des Treibers, des Toolkits und des Geräte-Plugins deaktivieren, da diese bereits im EKS enthalten sind. AMIs

Zusätzlich zu den standardmäßigen EKS AMI-Komponenten umfasst das EKS-optimierte Bottlerocket NVIDIA AMIs die folgenden Komponenten. Die minimalen Abhängigkeiten für EFA (Kernelmodul und RDMA-Core) sind in allen Bottlerocket-Varianten installiert.

- NVIDIA Kubernetes-Geräte-Plugin
- NVIDIA-Treiber
- NVIDIA CUDA-Benutzermodus-Treiber
- NVIDIA-Container-Toolkit
- NVIDIA-Fabric-Manager
- NVIDIA blieb hartnäckig
- NVIDIA IMEX-Treiber
- NVLink NVIDIA-Subnetzmanager

- NVIDIA MIG-Manager

Einzelheiten zum NVIDIA CUDA-Benutzermodus-Treiber und zum CUDA, der in Anwendungscontainern runtime/libraries verwendet wird, finden Sie in der NVIDIA-Dokumentation. Bei der von angezeigten CUDA-Version `nvidia-smi` handelt es sich um die Version des NVIDIA CUDA-Benutzermodus-Treibers, der auf dem Host installiert ist und mit dem in Anwendungscontainern verwendeten CUDA runtime/libraries kompatibel sein muss.

Einzelheiten zu den installierten Paketen und ihren Versionen finden Sie in den Bottlerocket-Versionsinformationen in der Bottlerocket-Dokumentation. Das EKS-optimierte Bottlerocket NVIDIA AMIs unterstützt Kernel 6.12 für Kubernetes-Versionen 1.33 und höher und die NVIDIA-Treiberversion 580 für Kubernetes-Versionen 1.34 und höher. Für die Verwendung von CUDA 13+ ist der NVIDIA 580-Treiber erforderlich.

EKS-optimiertes Neuron AMIs

Einzelheiten zur Ausführung von Trainings- und Inferenz-Workloads mithilfe von Neuron mit Amazon EKS finden Sie in den folgenden Referenzen:

- Container — Kubernetes — Erste Schritte in der Neuron-Dokumentation AWS
- Schulungsbeispiel in AWS Neuron EKS Samples auf GitHub
- Bereitstellen von ML-Inferenz-Workloads mit Inferentia auf Amazon EKS

Das neueste EKS-optimierte Neuron finden Sie unter und. AMIs the section called “Holen Sie sich die neuesten IDs” the section called “Aktuelle IDs abrufen”

Wenn Sie Amazon Elastic Fabric Adaptor (EFA) mit dem EKS-optimierten AL2 023 oder Bottlerocket Neuron verwenden AMIs, müssen Sie das EFA-Geräte-Plugin separat installieren. Weitere Informationen finden Sie unter the section called “Trainings-Cluster mit EFA einrichten”.

AL2EKS 023 Neuron AMIs

Die EKS-optimierten AL2 023 Neuron enthalten AMIs weder das Neuron Kubernetes-Geräte-Plugin noch die Neuron Kubernetes-Scheduler-Erweiterung, und diese müssen separat installiert werden. Weitere Informationen finden Sie unter the section called “Installieren Sie das Neuron Kubernetes-Geräte-Plugin”.

Zusätzlich zu den standardmäßigen EKS AMI-Komponenten AMIs umfasst das EKS-optimierte AL2 023 Neuron die folgenden Komponenten.

- Neuronentreiber () aws-neuronx-dkms
- Werkzeuge für Neuronen () aws-neuronx-tools
- EFA minimal (Kernelmodul und RDMA-Core)

Einzelheiten zur AMIs Konfiguration der Neuron-Abhängigkeiten durch EKS AL2 023 Neuron AMI finden Sie im EKS 023 Neuron AMI-Installationsskript. In den für EKS optimierten AL2023-Versionen finden Sie Informationen GitHub zu den Komponentenversionen, die in der enthalten sind. AMIs Sie können die Liste der installierten Pakete und ihrer Versionen auf einer laufenden EC2 Instanz mit dem Befehl finden. `dnf list installed`

EKS Bottlerocket Neuron AMIs

Die Standard-Bottlerocket-Varianten (aws-k8s) beinhalten die Neuron-Abhängigkeiten, die automatisch erkannt und geladen werden, wenn sie auf Inferentia- oder Trainium-Instances ausgeführt werden. AWS EC2

Die EKS-optimierten Bottlerocket enthalten weder das Neuron Kubernetes-Geräte-Plugin noch die Neuron Kubernetes-Scheduler-Erweiterung, und diese AMIs müssen separat installiert werden. Weitere Informationen finden Sie unter the section called “Installieren Sie das Neuron Kubernetes-Geräte-Plugin”.

Zusätzlich zu den standardmäßigen EKS AMI-Komponenten umfasst das EKS-optimierte Bottlerocket Neuron AMIs die folgenden Komponenten.

- Neuronentreiber () aws-neuronx-dkms
- EFA minimal (Kernelmodul und RDMA-Core)

Wenn Sie das EKS-optimierte Bottlerocket AMIs mit Neuron-Instances verwenden, muss Folgendes in den Bottlerocket-Benutzerdaten konfiguriert werden. Diese Einstellung ermöglicht es dem Container, den Besitz des bereitgestellten Neuron-Geräts auf der Grundlage der in der Workload-Spezifikation angegebenen Werte und zu übernehmen. `runAsUser` `runAsGroup` Weitere Informationen zur Neuron-Unterstützung in Bottlerocket finden Sie in der Readme-Datei für Schnellstart auf EKS unter. GitHub

```
[settings]
[settings.kubernetes]
device-ownership-from-security-context = true
```

Informationen zur Neuron-Treiberversion, die im EKS-optimierten Bottlerocket enthalten ist, finden Sie im Bottlerocket-Kernel-Kit-Changelog. AMIs

Installieren Sie das Kubernetes-Geräte-Plugin für GPUs

Kubernetes-Geräte-Plugins waren der wichtigste Mechanismus GPUs, um spezielle Infrastrukturen wie Netzwerkschnittstellen und Netzwerkadapter als nutzbare Ressourcen für Kubernetes-Workloads zu bewerben. Dynamic Resource Allocation (DRA) ist zwar die future für das Gerätemanagement in Kubernetes, aber die meisten spezialisierten Infrastrukturanbieter unterstützen DRA-Treiber noch früh. Kubernetes-Geräte-Plugins sind auch heute noch ein weit verbreiteter Ansatz für die Verwendung GPUs in Kubernetes-Clustern.

Überlegungen

- Wenn Sie das für EKS optimierte AL2 023 AMIs mit NVIDIA verwenden, müssen Sie das NVIDIA GPUs Kubernetes-Geräte-Plugin installieren. Sie können das NVIDIA Kubernetes-Geräte-Plugin mit Helm, einem Kubernetes-Tooling Ihrer Wahl oder dem NVIDIA-GPU-Operator installieren und verwalten.
- Wenn Sie das EKS-optimierte Bottlerocket AMIs mit NVIDIA verwenden, müssen Sie das NVIDIA Kubernetes-Geräte-Plugin nicht installieren GPUs, da es bereits im EKS-optimierten Bottlerocket enthalten ist. AMIs Dies gilt auch, wenn Sie GPU-Instanzen mit dem EKS-Automatikmodus verwenden.
- Wenn Sie das EKS-optimierte AL2 023 oder Bottlerocket AMIs mit AWS Inferentia oder Trainium verwenden GPUs, müssen Sie das Neuron Kubernetes-Geräte-Plugin und optional die Neuron Kubernetes-Scheduler-Erweiterung installieren. Weitere Informationen finden Sie in der Neuron-Dokumentation zur Ausführung auf EKS.

Installieren Sie das NVIDIA Kubernetes-Geräte-Plugin

Das folgende Verfahren beschreibt, wie Sie das NVIDIA Kubernetes-Geräte-Plug-In installieren und einen Beispieltest auf NVIDIA-GPU-Instanzen ausführen.

Voraussetzungen

- Der EKS-Cluster wurde erstellt
- NVIDIA-GPU-Knoten, die im Cluster mit EKS-optimiertem AL2 023 NVIDIA AMI ausgeführt werden
- Helm ist in Ihrer Befehlszeilenumgebung installiert. Weitere Informationen finden Sie unter Anweisungen zur Einrichtung von Helm.

Verfahren

1. Fügen Sie das nvdp Helm-Chart-Repository hinzu.

```
helm repo add nvdp https://nvidia.github.io/k8s-device-plugin
```

2. Aktualisieren Sie Ihr lokales Helm-Repository, um sicherzustellen, dass Sie über die aktuellsten Diagramme verfügen.

```
helm repo update
```

3. Holen Sie sich die neueste Version des NVIDIA Kubernetes-Geräte-Plugins

```
helm search repo nvdp --devel
```

```
NAME                        CHART VERSION APP VERSION DESCRIPTION
nvdp/gpu-feature-discovery 0.17.4        0.17.4      ...
nvdp/nvidia-device-plugin  0.17.4        0.17.4      ...
```

4. Installieren Sie das NVIDIA Kubernetes-Geräte-Plugin auf Ihrem Cluster und `0.17.4` ersetzen Sie es durch die neueste Version aus dem obigen Befehl.

```
helm install nvdp nvdp/nvidia-device-plugin \
  --namespace nvidia \
  --create-namespace \
  --version 0.17.4 \
  --set gfd.enabled=true
```

5. Stellen Sie sicher, dass das NVIDIA Kubernetes-Geräte-Plugin in Ihrem Cluster ausgeführt wird. Die folgende Ausgabe zeigt die Ausgabe mit zwei Knoten im Cluster.

```
kubectl get ds -n nvidia nvdp-nvidia-device-plugin
```

```
NAME                        DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE
 SELECTOR   AGE
nvdp-nvidia-device-plugin   2         2         2       2            2
 <none>          11m
```

6. Stellen Sie mit dem folgenden Befehl sicher, dass Ihre Knoten über Allocatable GPUs verfügen.

```
kubectl get nodes "-o=custom-
columns=NAME:.metadata.name,GPU:.status.allocatable.nvidia\.com/gpu"
```

```
NAME                                           GPU
ip-192-168-11-225.us-west-2.compute.internal   1
ip-192-168-24-96.us-west-2.compute.internal    1
```

7. Erstellen Sie eine Datei mit dem Namen `nvidia-smi.yaml` und dem folgenden Inhalt. Dieses Manifest startet ein minimales AL2 023-Container-Image, das `nvidia-smi` auf einem Knoten ausgeführt wird.

```
apiVersion: v1
kind: Pod
metadata:
  name: nvidia-smi
spec:
  restartPolicy: OnFailure
  containers:
    - name: gpu-demo
      image: public.ecr.aws/amazonlinux/amazonlinux:2023-minimal
      command: ['/bin/sh', '-c']
      args: ['nvidia-smi && tail -f /dev/null']
      resources:
        limits:
          nvidia.com/gpu: 1
  tolerations:
    - key: 'nvidia.com/gpu'
      operator: 'Equal'
      value: 'true'
      effect: 'NoSchedule'
```

8. Wenden Sie das Manifest mit dem folgenden Befehl an.

```
kubectl apply -f nvidia-smi.yaml
```

9. Nachdem der Pod ausgeführt wurde, zeigen Sie mit dem folgenden Befehl seine Protokolle an.

```
kubectl logs nvidia-smi
```

Eine Beispielausgabe sieht wie folgt aus.

```
+-----------------------------------------------------------------------------------------
+
| NVIDIA-SMI XXX.XXX.XX            Driver Version: XXX.XXX.XX     CUDA Version: XX.X
     |
|-----------------------------------------+-----------------------
+----------------------+
| GPU  Name                 Persistence-M | Bus-Id          Disp.A | Volatile Uncorr.
 ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |           Memory-Usage | GPU-Util
 Compute M. |
|                                         |                        |
 MIG M. |
|=========================================+========================
+======================|
|   0  NVIDIA L4                      On  |   00000000:31:00.0 Off |
   0 |
| N/A   27C    P8             11W /   72W |       0MiB /  23034MiB |      0%
 Default |
|                                         |                        |
 N/A |
+-----------------------------------------+-----------------------
+----------------------+

+-----------------------------------------------------------------------------------------
+
| Processes:
     |
|  GPU   GI   CI              PID   Type   Process name                        GPU
 Memory |
|        ID   ID                                                               Usage
     |
|
=========================================================================================|
|  No running processes found
     |
+-----------------------------------------------------------------------------------------
+
```

Installieren Sie das Neuron Kubernetes-Geräte-Plugin

Das folgende Verfahren beschreibt, wie Sie das Neuron Kubernetes-Geräte-Plugin installieren und einen Beispieltest auf einer Inferentia-Instanz ausführen.

Voraussetzungen

- Der EKS-Cluster wurde erstellt
- Neuron-GPU-Knoten, die im Cluster mit EKS-optimiertem AL2 023 Neuron AMI oder Bottlerocket AMI ausgeführt werden
- Helm ist in Ihrer Befehlszeilenumgebung installiert. Weitere Informationen finden Sie unter Anweisungen zur Einrichtung von Helm.

Verfahren

1. Installieren Sie das Neuron Kubernetes-Geräte-Plugin auf Ihrem Cluster.

```
helm upgrade --install neuron-helm-chart oci://public.ecr.aws/neuron/neuron-helm-
chart \
    --set "npd.enabled=false"
```

2. Stellen Sie sicher, dass das Neuron Kubernetes-Geräte-Plugin in Ihrem Cluster ausgeführt wird. Die folgende Ausgabe zeigt die Ausgabe mit einem einzelnen Neuron-Knoten im Cluster.

```
kubectl get ds -n kube-system neuron-device-plugin
```

```
NAME                   DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE
 SELECTOR   AGE
neuron-device-plugin   1         1         1       1            1           <none>
       72s
```

3. Stellen Sie mit dem folgenden Befehl sicher, dass Ihre Knoten über Allocatable NueronCores verfügen.

```
kubectl get nodes "-o=custom-
columns=NAME:.metadata.name,NeuronCore:.status.allocatable.aws\.amazon\.com/
neuroncore"
```

```
NAME                                           NeuronCore
```

```
ip-192-168-47-173.us-west-2.compute.internal   2
```

4. Stellen Sie mit dem folgenden Befehl sicher, dass Ihre Knoten über Allocatable NueronDevices verfügen.

```
kubectl get nodes "-o=custom-
columns=NAME:.metadata.name,NeuronDevice:.status.allocatable.aws\.amazon\.com/neuron"
```

```
NAME                                          NeuronDevice
ip-192-168-47-173.us-west-2.compute.internal   1
```

5. Erstellen Sie eine Datei mit dem Namen `neuron-ls.yaml` und dem folgenden Inhalt. Dieses Manifest startet einen Neuron Monitor-Container, in dem das Tool installiert ist. `neuron-ls`

```
apiVersion: v1
kind: Pod
metadata:
  name: neuron-ls
spec:
  restartPolicy: Never
  containers:
  - name: neuron-container
    image: public.ecr.aws/g4h4h0b5/neuron-monitor:1.0.0
    command: ["/bin/sh"]
    args: ["-c", "neuron-ls"]
    resources:
      limits:
        aws.amazon.com/neuron: 1
  tolerations:
  - key: "aws.amazon.com/neuron"
    operator: "Exists"
    effect: "NoSchedule"
```

6. Wenden Sie das Manifest mit dem folgenden Befehl an.

```
kubectl apply -f neuron-ls.yaml
```

7. Nachdem der Pod ausgeführt wurde, zeigen Sie mit dem folgenden Befehl seine Protokolle an.

```
kubectl logs neuron-ls
```

Unten sehen Sie eine Beispielausgabe.

```
instance-type: inf2.xlarge
instance-id: ...
+--------+--------+--------+---------+
| NEURON | NEURON | NEURON |   PCI   |
| DEVICE | CORES  | MEMORY |   BDF   |
+--------+--------+--------+---------+
| 0      | 2      | 32 GB  | 00:1f.0 |
+--------+--------+--------+---------+
```

Ausführung von GPU-beschleunigten Containern (Windows in EC2-G-Serie)

> **Important**
>
> Das Kubernetes-Geräte-Plugin für DirectX von TensorWorks ist ein Drittanbieter-Tool, das von AWS weder empfohlen, unterstützt noch gewartet wird. AWS übernimmt keine Verantwortung für die Sicherheit, Zuverlässigkeit oder Leistung dieses Plugins.

Erfahren Sie, wie Sie GPU-beschleunigte Windows-Container-Workloads in Amazon EKS (Elastic Kubernetes Service) mit NVIDIA-GPUs und dem Kubernetes-Geräte-Plugin für DirectX von TensorWorks ausführen. Weitere Informationen finden Sie unter Kubernetes-Geräte-Plugin für DirectX.

Es gibt zwei Hauptansätze für die Einrichtung der GPU-Beschleunigung für Ihre Windows-Container:

- Option 1: Entwickeln Sie ein benutzerdefiniertes, für Windows optimiertes EKS-AMI mit vorab installierten erforderlichen GPU-Treibern.
  - Verwenden Sie diesen Ansatz, wenn Sie eine konsistente, vorkonfigurierte Umgebung für die Ausführung GPU-beschleunigter Windows-Container benötigen und den zusätzlichen Aufwand für die Erstellung und Wartung des benutzerdefinierten AMIs in Kauf nehmen können.
- Option 2: Installieren Sie die erforderlichen GPU-Treiber auf Ihren EKS-Worker-Knoten, nachdem Sie Ihre Instance gestartet haben.
  - Verwenden Sie diesen Ansatz, wenn Sie einen einfacheren Einrichtungsprozess wünschen und es Ihnen nichts ausmacht, die GPU-Treiber auf jedem neuen Worker-Knoten zu installieren.

Dieser Ansatz eignet sich besser für eine Entwicklungsumgebung, in der Sie GPU-beschleunigte Workloads evaluieren oder Prototypen erstellen.

Beide Ansätze können mithilfe der in diesem Leitfaden beschriebenen Schritte umgesetzt werden.

Überlegungen

Dieser Leitfaden enthält Schritte zur Installation und Einrichtung der GPU-Beschleunigung für Ihre Windows-Container unter Verwendung von NVIDIA-GPUs, NVIDIA GRID-Treibern und dem Kubernetes-Geräte-Plugin für DirectX von TensorWorks. Die Schritte wurden getestet und verifiziert, um GPU-Beschleunigung für Ihre Windows-Container-Workloads in Amazon EKS bereitzustellen. Weitere Informationen zu kompatiblen Treibern und Geräte-Plugins finden Sie unter the section called “Bekannte Beschränkungen”. Beachten Sie Folgendes, bevor Sie fortfahren:

- Es wurden nur Instance-Typen der G-Familie mit NVIDIA-GRID-Treibern getestet und ihre Funktion mit diesem Handbuch bestätigt. Andere Instance-Typen und Treiberkombinationen können möglicherweise ebenfalls GPU-beschleunigte Windows-Container ausführen, erfordern jedoch möglicherweise zusätzliche Konfigurationsschritte, die in dieser Anleitung nicht behandelt werden.
- Es wurden nur DirectX-basierte Workloads getestet und ihre Funktion mit diesem Handbuch bestätigt. Andere GPU-APIs wie OpenGL, Vulkan und OpenCL sind möglicherweise mit der Ausführung von GPU-beschleunigten Windows-Containern kompatibel, erfordern jedoch möglicherweise zusätzliche Konfigurationsschritte, die in diesem Handbuch nicht behandelt werden.
- Es gibt einige bekannte Einschränkungen, die Sie beachten sollten, bevor Sie GPU-beschleunigte Windows-Container ausführen. Weitere Informationen finden Sie im Abschnitt the section called “Bekannte Beschränkungen”.

Voraussetzungen

Um die GPU-Beschleunigung für Ihre Windows-Container in Amazon EKS zu aktivieren, müssen Sie zunächst die folgenden Voraussetzungen erfüllen:

- Starten Sie einen Amazon-EKS-Cluster mit Kubernetes v1.27 oder neuer.
- Stellen Sie Windows-Knoten mit Windows Server 2022 oder neuer bereit.
- Stellen Sie Windows-Knoten in den Instance-Typen der G-Familie bereit, beispielsweise G4 oder G5.

- Stellen Sie Windows-Knoten mit einer Container-Laufzeitumgebung mit containerd `1.7.x` oder `2.x.x` bereit. (Lesen Sie the section called “Versionsinformationen abrufen”, um die containerd-Version in Ihrem Amazon-EKS-optimierten AMI zu überprüfen.)

Installation des GPU-Treibers auf jedem Windows-Knoten

Um die NVIDIA-GRID-Treiber auf Ihren EKS-Worker-Knoten zu installieren, befolgen Sie die Schritte, die unter NVIDIA-Treiber für Ihre Amazon-EC2-Instance beschrieben sind. Navigieren Sie zu den Installationsoptionen – Option 3: GRID-Treiber und befolgen Sie die Installationsschritte.

Installation für Windows Server Core

Installieren Sie für Windows Server Core, das keine Desktop-Erfahrung bietet, NVIDIA-GRID-Treiber im Hintergrund mithilfe der folgenden Befehle:

```
$nvidiaInstallerFilePath = nvidia-driver-installer.exe # Replace with path to installer
$installerArguments = "-s -clean -noreboot -noeula"
Start-Process -FilePath $nvidiaInstallerFilePath -ArgumentList $installerArguments -
Wait -NoNewWindow -PassThru
```

Überprüfen der Installation

Führen Sie den folgenden PowerShell-Befehl aus, um Diagnoseinformationen zu den GPUs auf der Instance anzuzeigen:

```
nvidia-smi
```

Dieser Befehl zeigt die Version des NVIDIA-Treibers sowie Informationen zur GPU-Hardware an. Stellen Sie sicher, dass die Ausgabe dieses Befehls mit der NVIDIA GRID-Treiberversion übereinstimmt, die Sie installiert haben sollten.

Bereitstellen des GPU-Geräte-Plugins auf jedem Knoten

Um die Erkennung und Freigabe der GPU-Ressourcen für Container auf Ihren Windows-Knoten zu ermöglichen, benötigen Sie ein Geräte-Plugin. Stellen Sie das DirectX-Geräte-Plugin von Tensorworks auf jedem Worker-Knoten bereit, indem Sie es als DaemonSet in Ihrem EKS-Cluster ausführen. Befolgen Sie die Installationsanleitung in der README.md, welche die folgenden Schritte umfasst. Es wird empfohlen, Folgendes zu tun:

- Stellen Sie das Geräte-Plugin im `kube-system`-Namespace bereit.
- Legen Sie angemessene Ressourcenbeschränkungen für das DaemonSet fest, um sicherzustellen, dass es nicht übermäßig viele Ressourcen auf Ihren Knoten verbraucht.

> **Note**
>
> Das Geräte-Plugin DaemonSet wird auf jedem Knoten als Host-Prozess-Container mit erweiterten Berechtigungen ausgeführt. Es wird empfohlen, RBAC-Steuerelemente zu implementieren, um den Zugriff auf dieses DaemonSet einzuschränken, sodass nur autorisierte Benutzer privilegierte Befehle ausführen können.

Beim Ausführen von GPU-beschleunigten Containern unterstützt das Geräte-Plugin zwei Modi:

- Single-Tenancy-Modus: In diesem Modus werden alle GPU-Ressourcen einem einzelnen Container auf der Instance zugewiesen. Installieren Sie die Geräte-Plugins mit Single-Tenancy-Unterstützung mithilfe des folgenden Befehls. Weitere Informationen finden Sie in der Datei README.md.

```
kubectl apply -f "https://raw.githubusercontent.com/TensorWorks/directx-device-plugins/main/deployments/default-daemonsets.yml"
```

- Multi-Tenancy-Modus: Dieser Modus ermöglicht die gemeinsame Nutzung von GPU-Ressourcen durch mehrere Container auf der Instance. Installieren Sie die Geräte-Plugins mit Multi-Tenancy-Unterstützung mithilfe des folgenden Befehls. Weitere Informationen finden Sie in der Datei README.md.

```
kubectl apply -f "https://raw.githubusercontent.com/TensorWorks/directx-device-plugins/main/deployments/multitenancy-inline.yml"
```

  Alternativ können Sie auch eine ConfigMap verwenden, um die Multi-Tenancy festzulegen.

```
kubectl apply -f "https://raw.githubusercontent.com/TensorWorks/directx-device-plugins/main/deployments/multitenancy-configmap.yml"
```

Überprüfung der Bereitstellung des Geräte-Plugins

Nachdem Sie das Geräte-Plugin bereitgestellt haben, ersetzen Sie `<namespace>` und führen Sie den folgenden Befehl aus, um zu überprüfen, ob das DirectX-Geräte-Plugin auf allen Ihren Windows-Knoten ordnungsgemäß ausgeführt wird.

```
kubectl get ds device-plugin-wddm -n <namespace>
```

Überprüfen, ob Container für die Bereitstellung bereit sind

Sobald das Geräte-Plugin DaemonSet auf den GPU-gestützten Windows-Worker-Knoten ausgeführt wird, überprüfen Sie mit dem folgenden Befehl, ob jeder Knoten über zuweisbare GPUs verfügt. Die entsprechende Zahl sollte mit der Anzahl der DirectX-Geräte auf jedem Knoten übereinstimmen.

```
kubectl get nodes "-o=custom-
columns=NAME:.metadata.name,DirectX:.status.allocatable.directx\.microsoft\.com/
display"
```

Ausführung von Windows-Containern mit GPU-Beschleunigung

Geben Sie vor dem Starten Ihrer Pods den Ressourcen-Namen `directx.microsoft.com/display` in `.spec.containers[].resources` an. Dies zeigt an, dass Ihre Container GPU-fähige Funktionen benötigen. `kube-scheduler`versucht, Ihre Pods auf Ihrem vorkonfigurierten Windows-Knoten mit verfügbaren GPU-Ressourcen zu platzieren.

Ein Beispiel hierfür ist der folgende Beispielbefehl, der `Job` startet, um eine Monte-Carlo-Simulation zur Schätzung des Pi-Werts auszuführen. Dieses Beispiel stammt aus dem GitHub-Repository Kubernetes-Geräte-Plugins für DirectX, das mehrere Beispiele enthält, mit denen Sie die GPU-Fähigkeiten Ihres Windows-Knotens testen können.

```
cat <<EOF | kubectl apply -f -
apiVersion: batch/v1
kind: Job
metadata:
  name: example-cuda-montecarlo-wddm
spec:
  template:
    spec:
      containers:
```

```
      - name: example-cuda-montecarlo-wddm
        image: "index.docker.io/tensorworks/example-cuda-montecarlo:0.0.1"
        resources:
          limits:
            directx.microsoft.com/display: 1
      nodeSelector:
        "kubernetes.io/os": windows
      restartPolicy: Never
  backoffLimit: 0
EOF
```

Bekannte Beschränkungen

Alle GPUs sind verwendbar

Alle GPUs auf der Instance können von jedem ausgeführten Container auf dem Host genutzt werden, auch wenn Sie eine bestimmte Anzahl von GPUs für einen bestimmten Container anfordern. Darüber hinaus ist das Standardverhalten so, dass alle auf dem Host ausgeführten Container die GPU mit dem Index 0 verwenden, auch wenn auf dem Knoten mehrere GPUs verfügbar sind Damit Multi-GPU-Aufgaben ordnungsgemäß funktionieren, müssen Sie das zu verwendende GPU-Gerät im Code Ihrer Anwendung explizit angeben.

Die genaue Implementierung zur Zuweisung eines für die Anwendung zu verwendenden Geräts hängt von der von Ihnen verwendeten Programmiersprache oder dem verwendeten Framework ab. Wenn Sie beispielsweise CUDA-Programmierung verwenden, können Sie zur Auswahl einer bestimmten GPU das zu verwendende Gerät explizit in Ihrem Anwendungscode mithilfe der Funktion cudaSetDevice() angeben.

Die Notwendigkeit, das Gerät explizit anzugeben, ist auf ein bekanntes Problem zurückzuführen, das Windows-Container betrifft. Sie können den Fortschritt bei der Behebung dieses Problems unter microsoft/Windows-Containers issue #333 verfolgen. Die folgende Tabelle stellt eine visuelle Darstellung und ein praktisches Beispiel für dieses GPU-Zuweisungsverhalten dar.

Stellen Sie sich ein Szenario vor, in dem es einen einzelnen Windows-Knoten vom EC2-Instance-Typ `g4dn.12xlarge` gibt, der mit vier GPUs ausgestattet ist. Nehmen Sie an, auf dieser Instance werden drei Pods gestartet. Die Tabelle zeigt, dass unabhängig von der Anzahl der von jedem Container angeforderten GPUs alle drei Pods Zugriff auf alle vier GPUs der Instance haben und standardmäßig die GPU mit dem Geräteindex 0 verwenden.

Pod	Gewünschte GPUs	Tatsächlicher GPU-Zugriff	Standardmäßige GPU-Nutzung	Verfügbare GPU-Indizes	Gesamtzahl der Instance-GPUs
Pod 1	1 GPU	Alle 4 GPUs	GPU mit index 0	0, 1, 2, 3	4
Pod 2	2 GPUs	Alle 4 GPUs	GPU mit index 0	0, 1, 2, 3	4
Pod 3	1 GPU	Alle 4 GPUs	GPU mit index 0	0, 1, 2, 3	4

Support für Kubernetes-Geräte-Plugins

Die offizielle Implementierung des Kubernetes-Geräte-Plugins durch NVIDIA unterstützt Windows nicht. Sie können den Fortschritt bei der Hinzufügung der offiziellen Windows-Unterstützung im NVIDIA/k8s-device-plugin Problem #419 verfolgen.

Einschränkungen für GPU-Rechen-Instances

Abhängig von Ihrer AWS-Konto-Konfiguration gelten möglicherweise Beschränkungen hinsichtlich der Anzahl und Art der Amazon-EC2-GPU-Rechen-Instances, die Sie starten können. Wenn Sie zusätzliche Kapazität benötigen, können Sie eine Kontingenterhöhung anfordern.

Notwendigkeit, eine für Windows-GPUs-optimierte AMI zu erstellen

Amazon EKS stellt keine EKS-Windows-GPU-optimierte AMI oder eine von EC2 Image Builder verwaltete Komponente zur Verfügung. Sie müssen die Schritte in dieser Anleitung befolgen, um eine benutzerdefinierte EKS-Windows-optimierte AMI mit den erforderlichen vorinstallierten GPU-Treibern zu erstellen, oder die erforderlichen GPU-Treiber auf Ihren EKS-Worker-Knoten installieren, nachdem Sie Ihre Instances gestartet haben.

Inferentia und Trainium werden nicht unterstützt

AWS Inferentia und AWS Trainium-basierte Workloads werden unter Windows nicht unterstützt.

Ausführung von Machine-Learning-Trainings in Amazon EKS mit Elastic Fabric Adapter

In diesem Thema wird beschrieben, wie Elastic Fabric Adapter (EFA) in Pods integriert werden, die in Ihrem Amazon-EKS-Cluster bereitgestellt werden. Elastic Fabric Adapter (EFA) ist eine Netzwerkschnittstelle für EC2 Amazon-Instances, mit der Sie Anwendungen ausführen können, die ein hohes Maß an Kommunikation zwischen Knoten erfordern, und zwar in großem Umfang. AWS Die speziell für die Umgehung von Betriebssystemen entwickelte Hardware-Schnittstelle verbessert die Leistung der Instance-übergreifenden Kommunikation, was für die Skalierung dieser Anwendungen von entscheidender Bedeutung ist. Mit EFA können High Performance Computing (HPC) -Anwendungen, die das Message Passing Interface (MPI) und Machine Learning (ML) -Anwendungen verwenden, die die NVIDIA Collective Communications Library (NCCL) verwenden, auf Tausende von oder skalieren. CPUs GPUs Dadurch erhalten Sie die Anwendungsleistung von lokalen HPC-Clustern mit der On-Demand-Elastizität und Flexibilität der Cloud. AWS Durch die Integration von EFA in Anwendungen, die auf Amazon-EKS-Clustern ausgeführt werden, können Sie die Zeit für die Durchführung umfangreicher verteilter Schulungs-Workloads verkürzen, ohne dass zusätzliche Instances zu Ihrem Cluster hinzugefügt werden müssen. Weitere Informationen zu EFA finden Sie unter Elastic Fabric Adapter.

Instance-Typen mit EFA

Das AWS EFA Kubernetes Device Plugin unterstützt alle EC2 Amazon-Instance-Typen, die EFA haben. Eine Liste aller Instance-Typen mit EFA finden Sie unter Unterstützte Instance-Typen im EC2 Amazon-Benutzerhandbuch. Um ML-Anwendungen schnell auszuführen, empfehlen wir jedoch, dass eine Instance zusätzlich zu AWS EFA über Hardwarebeschleunigungschips wie nVidia GPUs, Inferentia-Chips oder AWS Trainium-Chips verfügt. Eine Liste der Instance-Typen mit Hardwarebeschleunigungschips und EFA finden Sie unter Accelerated Computing im EC2 Amazon-Benutzerhandbuch.

Wenn Sie Instance-Typen vergleichen, um zwischen ihnen zu wählen, berücksichtigen Sie die Anzahl der für diesen Instance-Typ verfügbaren EFA-Netzwerkkarten sowie die Anzahl der Beschleunigerkarten, die CPU-Leistung und die Speichergröße. Sie können bis zu einem EFA pro Netzwerkkarte zuweisen. Ein EFA zählt als Netzwerkschnittstelle. Wie viele EFA für jeden Instance-Typ mit EFA verfügbar sind, finden Sie in der Liste der Netzwerkkarten im EC2 Amazon-Benutzerhandbuch.

EFA- und reine EFA-Schnittstellen

Ein Elastic Fabric Adapter (EFA) ist eine Netzwerkschnittstelle, die die Funktionen eines Elastic Network Adapters (ENA) und einer OS-Bypass-Schnittstelle kombiniert und auf dem Scalable AWS Reliable Datagram (SRD) -Protokoll basiert. Die EFA-Funktionen ermöglichen Anwendungen die direkte Kommunikation mit der Hardware für eine latenzarme Übertragung. Sie können über reine EFA-Schnittstellen auf die EFA-Funktionen zugreifen und so die Kommunikation auf Schnittstellen innerhalb derselben Availability Zone beschränken.

Um Knoten zu erstellen, die nur über EFA-Schnittstellen verfügen können, müssen Sie eine benutzerdefinierte EC2 Startvorlage verwenden und den Wert auf setzen. `InterfaceType efa-only` In Ihrer benutzerdefinierten Startvorlage können Sie die Netzwerkkarte nicht `0` auf eine reine EFA-Schnittstelle festlegen, da dies die primäre Netzwerkkarte und Netzwerkschnittstelle der Instance ist. EC2 Für reine EFA-Schnittstellen benötigen Sie VPC-CNI-Version `1.18.5` oder höher. Wenn Sie Amazon Linux 2 verwenden, muss die AMI-Version für reine EFA-Schnittstellen `v20240928` oder höher sein.

Das folgende Verfahren führt Sie zum Erstellen eines EKS-Clusters `eksctl` mit Knoten, die über NVidia GPUs - und EFA-Schnittstellen verfügen. Sie können `eksctl` nicht zum Erstellen von Knoten und Knotengruppen verwenden, die nur EFA-Schnittstellen verwenden.

Voraussetzungen

- Ein vorhandener Amazon-EKS-Cluster. Wenn Sie keinen vorhandenen Cluster haben, erstellen Sie einen mit Erste Schritte. Ihr Cluster muss in einer VPC bereitgestellt werden, in der mindestens ein privates Subnetz mit genügend verfügbaren IP-Adressen zum Bereitstellen von Knoten vorhanden sind. Das private Subnetz muss über einen ausgehenden Internetzugang verfügen, der von einem externen Gerät, z. B. einem NAT-Gateway, bereitgestellt wird.

  Wenn Sie planen, `eksctl` zu verwenden, um Ihre Knotengruppe zu erstellen, kann `eksctl` auch einen Cluster für Sie erstellen.
- Version `2.12.3` oder höher oder Version `1.27.160` oder höher der auf Ihrem Gerät installierten und konfigurierten AWS Befehlszeilenschnittstelle (AWS CLI) oder AWS CloudShell. Um Ihre aktuelle Version zu überprüfen, verwenden Sie `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Paketmanager wie `yum` `apt-get`, oder Homebrew für macOS liegen oft mehrere Versionen hinter der neuesten Version der AWS CLI. Informationen zur Installation der neuesten Version finden Sie unter Installation und Schnellkonfiguration mit aws configure im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle. Die AWS CLI-Version, in der

installiert ist, AWS CloudShell kann auch mehrere Versionen hinter der neuesten Version liegen. Informationen zur Aktualisierung finden Sie im AWS CloudShell Benutzerhandbuch unter AWS CLI in Ihrem Home-Verzeichnis installieren.

- Das `kubectl`-Befehlszeilen-Tool ist auf Ihrem Gerät oder in der AWS CloudShell installiert. Die Version kann mit der Kubernetes-Version Ihres Clusters identisch sein oder bis zu einer Nebenversion älter oder neuer sein. Wenn Ihre Clusterversion beispielsweise `1.29` ist, können Sie `kubectl`-Version `1.28`, `1.29`, oder `1.30` damit verwenden. Informationen zum Installieren oder Aktualisieren von `kubectl` finden Sie unter the section called “kubectl und eksctl einrichten”.
- Sie müssen das Amazon-VPC-CNI-Plugin für die Kubernetes-Version `1.7.10` oder höher installiert haben, bevor Sie Worker-Knoten starten, die mehrere Elastic Fabric Adapter, wie etwa `p4d` oder `p5` unterstützen. Weitere Informationen zum Aktualisieren Ihres Amazon-VPC-CNI-Plugins für die Kubernetes-Version finden Sie unter the section called “Amazon VPC CNI”.
- Für p6-b200-Instances müssen Sie das EFA-Geräte-Plugin-Version v0.5.6 oder höher verwenden.

> **Important**
>
> Eine wichtige Überlegung, die für die Übernahme von EFA mit Kubernetes erforderlich ist, ist die Konfiguration und Verwaltung von Huge Pages als Ressource im Cluster. Weitere Informationen dazu finden Sie unter Manage Huge Pages in der Kubernetes-Dokumentation. EC2 Amazon-Instances, auf denen der EFA-Treiber installiert ist, weisen 5128 Huge Pages mit 2 MiB vorab zu, die Sie als Ressourcen anfordern können, die Sie in Ihren Jobspezifikationen nutzen können.

Knotengruppen erstellen

Das folgende Verfahren hilft Ihnen dabei, eine Knotengruppe mit einer `p4d.24xlarge` gesicherten Knotengruppe mit EFA-Schnittstellen und GPUDirect RDMA zu erstellen und einen Beispieltest der NVIDIA Collective Communications Library (NCCL) für die NCCL-Leistung mit mehreren Knoten durchzuführen. EFAs Das Beispiel kann als Vorlage für verteiltes Deep-Learning-Training auf Amazon EKS verwendet EFAs werden.

1. Ermitteln Sie, welche EC2 Amazon-Instance-Typen, die EFA unterstützen, in der AWS Region verfügbar sind, in der Sie Knoten bereitstellen möchten. *`region-code`*Ersetzen Sie es durch die AWS Region, in der Sie Ihre Knotengruppe bereitstellen möchten.

```
aws ec2 describe-instance-types --region region-code \
    --filters Name=network-info.efa-supported,Values=true \
    --query "InstanceTypes[*].[InstanceType]" --output text
```

Wenn Sie Knoten bereitstellen, muss der Instance-Typ, den Sie bereitstellen möchten, in der AWS Region verfügbar sein, in der sich Ihr Cluster befindet.

2. Bestimmen Sie, in welchen Availability Zones der Instance-Typ bereitgestellt werden soll. In diesem Tutorial wird der `p5.48xlarge` Instance-Typ verwendet und muss in der Ausgabe für die AWS Region zurückgegeben werden, die Sie im vorherigen Schritt angegeben haben. Wenn Sie Knoten in einem Produktionscluster bereitstellen, *`p5.48xlarge`* ersetzen Sie ihn durch einen beliebigen Instance-Typ, der im vorherigen Schritt zurückgegeben wurde.

```
aws ec2 describe-instance-type-offerings --region region-code \
    --location-type availability-zone --filters Name=instance-
type,Values=p4d.24xlarge,p5.48xlarge \
    --query 'InstanceTypeOfferings[*].Location' --output text
```

Eine Beispielausgabe sieht wie folgt aus.

```
us-west-2a    us-west-2c    us-west-2b
```

Beachten Sie, dass Availability Zones zur Verwendung in späteren Schritten zurückgegeben wurden Wenn Sie Knoten in einem Cluster bereitstellen, muss Ihre VPC über Subnetze mit verfügbaren IP-Adressen in einer der in der Ausgabe zurückgegebenen Availability Zones verfügen.

3. Erstellen Sie eine Knotengruppe mithilfe von `eksctl`. Sie benötigen eine Version `0.215.0` oder eine neuere Version des `eksctl` Befehlszeilentools, das auf Ihrem Gerät installiert ist oder AWS CloudShell. Informationen zum Installieren und Aktualisieren von `eksctl` finden Sie in der Dokumentation zu `eksctl` unter Installation.

   a. Fügen Sie eine Datei namens *`efa-cluster.yaml`* mit dem folgenden Inhalt hinzu. Ersetzen Sie die Beispielwerte durch Ihre eigenen Werte. Sie können `p5.48xlarge` mit einer anderen Instance ersetzen. Wenn Sie dies tun, stellen Sie sicher, dass die Werte für `availabilityZones` Availability Zones sind, die in Schritt 1 für den Instance-Typ zurückgegeben wurden.

```
apiVersion: eksctl.io/v1alpha5
```

```
kind: ClusterConfig

metadata:
  name: my-efa-cluster
  region: region-code
  version: "1.XX"

iam:
  withOIDC: true

availabilityZones: ["us-west-2a", "us-west-2c"]

managedNodeGroups:
  - name: my-efa-ng
    instanceType: p5.48xlarge
    minSize: 1
    desiredCapacity: 2
    maxSize: 3
    availabilityZones: ["us-west-2a"]
    volumeSize: 300
    privateNetworking: true
    efaEnabled: true
```

b. Erstellen einer verwalteten Knotengruppe in einem bestehenden Cluster

```
eksctl create nodegroup -f efa-cluster.yaml
```

Wenn Sie über keinen vorhandenen Cluster verfügen, können Sie den folgenden Befehl ausführen, um einen Cluster und die Knotengruppe zu erstellen.

```
eksctl create cluster -f efa-cluster.yaml
```

> **Note**
>
> Da der in diesem Beispiel verwendete Instance-Typ GPUs, installiert `eksctl` automatisch das NVIDIA Kubernetes-Geräte-Plug-In auf jeder Instance für Sie, wenn Sie Amazon Linux 2 verwenden. Dies ist für Bottlerocket nicht erforderlich, da das NVIDIA-Geräte-Plugin in die EKS-NVIDIA-Variante von Bottlerocket integriert ist. Wenn `efaEnabled` in der Knotengruppen-Konfiguration auf `true` gesetzt ist, stellt `eksctl` automatisch das EFA-Geräte-Plugin auf den Knoten bereit.

Verwendung von Bottlerocket mit EFA

Bottlerocket-AMI-Version 1.28.0 und höher bieten offiziellen Support für EFA. Um Bottlerocket für EFA-fähige Knoten zu verwenden, geben Sie `amiFamily: Bottlerocket` in Ihrer Konfiguration an. Wenn Sie eine benutzerdefinierte AMI-ID verwenden müssen, müssen Sie die Standard-`nodeGroups` anstelle von `managedNodeGroups` verwenden.

Hier ein Beispiel für eine Konfiguration:

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-efa-bottlerocket-cluster
  region: region-code
  version: "1.XX"

iam:
  withOIDC: true

availabilityZones: ["us-west-2a", "us-west-2c"]

managedNodeGroups:
  - name: my-efa-bottlerocket-ng
    instanceType: p5.48xlarge
    minSize: 1
    desiredCapacity: 2
    maxSize: 3
    availabilityZones: ["us-west-2a"]
    volumeSize: 300
    privateNetworking: true
    efaEnabled: true
    amiFamily: Bottlerocket
    bottlerocket:
      enableAdminContainer: true
      settings:
        kernel:
          sysctl:
            "vm.nr_hugepages": "3000"  # Configures 3000 * 2Mi = 6000Mi hugepages
```

Die oben genannte `vm.nr_hugepages`-sysctl-Einstellung konfiguriert die Anzahl der 2Mi-Hugepages. In diesem Beispiel bedeutet 3 000: 3 000 x 2Mi = 6 000Mi Hugepages.

Installation des EFA-Geräte-Plugins überprüfen

Wenn Sie eine Knotengruppe mit `efaEnabled: true` erstellen, stellt `eksctl` automatisch das EFA-Kubernetes-Geräte-Plugin für Sie bereit. Sie können überprüfen, ob das Geräte-Plugin installiert ist und ordnungsgemäß funktioniert:

1. Überprüfen Sie den Status DaemonSet :

   ```
   kubectl get daemonsets -n kube-system
   ```

   Beispielausgabe:

   ```
   NAME                                  DESIRED   CURRENT   READY   UP-TO-DATE
    AVAILABLE   NODE SELECTOR   AGE
   aws-efa-k8s-device-plugin-daemonset   2         2         2       2            2
         <none>          6m16s
   ...
   ```

   Hier läuft das EFA-Geräte-Plugin DaemonSet auf zwei Knoten. Beide sind READY und AVAILABLE.

2. Überprüfen Sie als Nächstes die Pods, die DaemonSet erstellt wurden von:

   ```
   kubectl get pods -n kube-system -l name=aws-efa-k8s-device-plugin
   ```

   Beispielausgabe:

   ```
   NAME                                        READY   STATUS    RESTARTS   AGE
   aws-efa-k8s-device-plugin-daemonset-d68bs   1/1     Running   0          6m16s
   aws-efa-k8s-device-plugin-daemonset-w4l8t   1/1     Running   0          6m16s
   ```

   Die EFA-Geräte-Plugin-Pods befinden sich im Status „Ausgeführt“, was bestätigt, dass das Plugin erfolgreich bereitgestellt wurde und betriebsbereit ist.

3. Überprüfen Sie die Ressourcen-Registrierung:

   Sie können überprüfen, ob die `vpc.amazonaws.com/efa`-Ressource beim kubelet registriert ist, indem Sie die Knoten beschreiben:

   ```
   kubectl describe nodes
   ```

Wenn die EFA-Ressource ordnungsgemäß registriert ist, wird sie unter „Kapazität“ und „Zuweisbare Ressourcen“ des Knotens aufgeführt. Beispiel:

```
Capacity:
  ...
  vpc.amazonaws.com/efa:  4
Allocatable:
  ...
  vpc.amazonaws.com/efa:  4
```

Diese Ausgabe bestätigt, dass der Knoten die EFA-Ressource erkennt und sie für Pods verfügbar macht, die sie anfordern.

(Optional) Leistung der EFA testen

Wir empfehlen Ihnen, die EFA-Einrichtung zu testen. Sie können die NCCL-Tests im `aws-samples/awsome-distributed-training` Repository am verwenden. GitHub NCCL-Tests bewerten die Leistung des Netzwerks mithilfe der Nvidia Collective Communication Library. Mit den folgenden Schritten können Sie NCCL-Tests in Amazon EKS durchführen.

1. Bereitstellen des Kubeflow-MPI-Operators:

   Für die NCCL-Tests können Sie den Kubeflow MPI Operator anwenden. Der MPI Operator macht es einfach, auf Kubernetes verteiltes Training im Allreduce-Stil durchzuführen. Weitere Informationen finden Sie unter MPI-Operator on. GitHub

2. Führen Sie den NCCL-Leistungstest mit mehreren Knoten aus, um RDMA/EFA zu überprüfen: GPUDirect

   Führen Sie den standardmäßigen NCCL-Leistungstest aus, um die NCCL-Leistung mit GPUDirect RDMA over EFA zu überprüfen. Weitere Informationen finden Sie im offiziellen NCCL-Tests-Repo unter. GitHub

   Führen Sie die folgenden Schritte aus, um einen NCCL-Leistungstest mit zwei Knoten auszuführen. Im Beispiel für einen NCCL-Testjob fordert jeder Worker acht GPUs, 5210 Mi, vier und 8000 Mi Arbeitsspeicher an EFAs, was praktisch bedeutet, dass jeder Worker alle Ressourcen einer Instanz verbraucht. `hugepages-2Mi` `p5.48xlarge`

   a. Erstellen Sie das Manifest: MPIJob

Kopieren Sie Folgendes in eine Datei mit dem Namen `nccl-tests.yaml`:

```
apiVersion: kubeflow.org/v2beta1
kind: MPIJob
metadata:
  name: nccl-tests
spec:
  runPolicy:
    cleanPodPolicy: Running
    backoffLimit: 20
  slotsPerWorker: 8
  mpiReplicaSpecs:
    Launcher:
      replicas: 1
      template:
         spec:
          restartPolicy: OnFailure
          containers:
          - image: public.ecr.aws/hpc-cloud/nccl-tests:latest
            imagePullPolicy: IfNotPresent
            name: test-nccl-launcher
            env:
             - name: PATH
               value: $PATH:/opt/amazon/efa/bin:/usr/bin
             - name: LD_LIBRARY_PATH
               value: /opt/amazon/openmpi/lib:/opt/nccl/build/lib:/opt/amazon/efa/
lib:/opt/aws-ofi-nccl/install/lib:/usr/local/nvidia/lib:$LD_LIBRARY_PATH
             - name: NCCL_DEBUG
               value: INFO
             - name: NCCL_BUFFSIZE
               value: '8388608'
             - name: NCCL_P2P_NET_CHUNKSIZE
               value: '524288'
             - name: NCCL_TUNER_PLUGIN
               value: /opt/aws-ofi-nccl/install/lib/libnccl-ofi-tuner.so
            command:
            - /opt/amazon/openmpi/bin/mpirun
            - --allow-run-as-root
            - --tag-output
            - -np
            - "16"
            - -N
            - "8"
```

```
            - --bind-to
            - none
            - -x
            - PATH
            - -x
            - LD_LIBRARY_PATH
            - -x
            - NCCL_DEBUG=INFO
            - -x
            - NCCL_BUFFSIZE
            - -x
            - NCCL_P2P_NET_CHUNKSIZE
            - -x
            - NCCL_TUNER_PLUGIN
            - --mca
            - pml
            - ^cm,ucx
            - --mca
            - btl
            - tcp,self
            - --mca
            - btl_tcp_if_exclude
            - lo,docker0,veth_def_agent
            - /opt/nccl-tests/build/all_reduce_perf
            - -b
            - "8"
            - -e
            - "16G"
            - -f
            - "2"
            - -g
            - "1"
            - -c
            - "1"
            - -n
            - "100"
    Worker:
      replicas: 2
      template:
        spec:
          nodeSelector:
            node.kubernetes.io/instance-type: "p5.48xlarge"
          containers:
          - image: public.ecr.aws/hpc-cloud/nccl-tests:latest
```

```
            imagePullPolicy: IfNotPresent
            name: nccl-tests-worker
            volumeMounts:
            - name: shmem
              mountPath: /dev/shm
            resources:
              limits:
                nvidia.com/gpu: 8
                hugepages-2Mi: 5120Mi
                vpc.amazonaws.com/efa: 32
                memory: 32000Mi
              requests:
                nvidia.com/gpu: 8
                hugepages-2Mi: 5120Mi
                vpc.amazonaws.com/efa: 32
                memory: 32000Mi
          volumes:
          - name: shmem
            hostPath:
              path: /dev/shm
```

b. Wenden Sie die NCCL-Tests MPIJob an:

   Reichen Sie das `MPIJob` durch Anwendung des Manifests ein. Dadurch werden zwei `p5.48xlarge` EC2 Amazon-Instances erstellt.

```
kubectl apply -f nccl-tests.yaml
```

   Eine Beispielausgabe sieht wie folgt aus.

```
mpijob.kubeflow.org/nccl-tests created
```

c. Überprüfen, ob der Auftrag Pods gestartet hat:

   Zeigen Sie Ihre ausgeführten Pods an.

```
kubectl get pods
```

   Eine Beispielausgabe sieht wie folgt aus.

```
NAME                             READY   STATUS     RESTARTS   AGE
nccl-tests-launcher-nbql9    0/1     Init:0/1   0          2m49s
```

```
nccl-tests-worker-0          1/1     Running    0          2m49s
nccl-tests-worker-1          1/1     Running    0          2m49s
```

Der MPI-Operator erstellt einen Start-Pod und zwei Worker-Pods (einen auf jedem Knoten).

d. Überprüfen Sie anhand der Protokolle, ob der Auftrag erfolgreich ausgeführt wird:

Zeigen Sie das Protokoll für den `nccl-tests-launcher`-Pod an. Ersetzen Sie *`nbql9`* mit dem Wert aus Ihrer Ausgabe.

```
kubectl logs -f nccl-tests-launcher-nbql9
```

If the test completed successfully, you can deploy your applications that use the Nvidia Collective Communication Library.

Verwenden Sie AWS Inferentia-Instances mit Amazon EKS für Machine Learning

In diesem Thema wird beschrieben, wie Sie einen Amazon EKS-Cluster mit Knoten erstellen, auf denen Amazon EC2 Inf1-Instances ausgeführt werden, und (optional) eine Beispielanwendung bereitstellen. Amazon EC2 Inf1-Instances werden mit AWS Inferentia-Chips betrieben, die speziell für hohe Leistung und kostengünstigste Inferenz in der Cloud entwickelt wurden. AWS Modelle für maschinelles Lernen werden mithilfe von AWS Neuron, einem speziellen Software Development Kit (SDK), das aus Compiler-, Runtime- und Profiling-Tools besteht, in Containern bereitgestellt, die die Inferenzleistung von Inferentia-Chips für maschinelles Lernen optimieren. AWS Neuron unterstützt beliebte Frameworks für maschinelles Lernen wie, und. TensorFlow PyTorch MXNet

> **Note**
>
> Das logische Neuron-Gerät IDs muss zusammenhängend sein. Wenn ein Pod, der mehrere Neuron-Geräte anfordert, für einen `inf1.24xlarge` Instanztyp `inf1.6xlarge` oder geplant ist (der mehr als ein Neuron-Gerät hat), kann dieser Pod nicht gestartet werden, wenn der Kubernetes-Scheduler ein nicht zusammenhängendes Gerät auswählt. IDs Weitere Informationen finden Sie unter Das logische Gerät muss zusammenhängend sein. IDs GitHub

Voraussetzungen

- `eksctl` ist auf Ihrem Computer installiert. Eine Installationsanleitung finden Sie bei Bedarf in der Dokumentation zu `eksctl` unter Installation.
- Installieren Sie `kubectl` auf Ihrem Computer. Weitere Informationen finden Sie unter the section called “kubectl und eksctl einrichten”.
- Installieren Sie `python3` auf Ihrem Computer (optional). Wenn Sie es noch nicht installiert haben, finden Sie unter Python-Downloads entsprechende Installationsanweisungen.

Erstellen eines Clusters

1. Erstellen Sie einen Cluster mit EC2 Inf1-Amazon-Instance-Knoten. Sie können *`inf1.2xlarge`* mit jedem Inf1-Instance-Typ ersetzen. Das `eksctl` Hilfsprogramm erkennt, dass Sie eine Knotengruppe mit einem `Inf1` Instance-Typ starten, und startet Ihre Knoten mit einem der für Amazon EKS optimierten beschleunigten Amazon Linux AMIs.

   > **Note**
   >
   > Sie können IAM-Rollen nicht für Dienstkonten bei TensorFlow Serving verwenden.

   ```
   eksctl create cluster \
       --name inferentia \
       --region region-code \
       --nodegroup-name ng-inf1 \
       --node-type inf1.2xlarge \
       --nodes 2 \
       --nodes-min 1 \
       --nodes-max 4 \
       --ssh-access \
       --ssh-public-key your-key \
       --with-oidc
   ```

   > **Note**
   >
   > Notieren Sie den Wert der folgenden Ausgabezeile. Dies wird in einem späteren (optionalen) Schritt verwendet.

```
[9]  adding identity "arn:aws: iam::111122223333:role/eksctl-inferentia-nodegroup-ng-
in-NodeInstanceRole-FI7HIYS3BS09" to auth ConfigMap
```

Beim Starten einer Knotengruppe mit `Inf1` Instanzen `eksctl` wird automatisch das AWS Neuron Kubernetes-Geräte-Plugin installiert. Dieses Plugin zeigt die Kubernetes-Scheduler-Neuron-Geräte als Systemressourcen an, die von einem Container angefordert werden können. Zusätzlich zu den Standardrichtlinien für Amazon-EKS-Knoten wird die IAM-Richtlinie für den schreibgeschützten Zugriff von Amazon S3 hinzugefügt. So kann die in einem späteren Schritt behandelte Beispielanwendung ein trainiertes Modell aus Amazon S3 laden.

2. Überprüfen Sie, ob alle Pods korrekt gestartet wurden.

```
kubectl get pods -n kube-system
```

Gekürzte Ausgabe:

```
NAME                                  READY   STATUS    RESTARTS   AGE
[...]
neuron-device-plugin-daemonset-6djhp   1/1     Running   0          5m
neuron-device-plugin-daemonset-hwjsj   1/1     Running   0          5m
```

(Optional) Stellen Sie ein TensorFlow Serving-Anwendungs-Image bereit

Ein trainiertes Modell muss für ein Inferentia-Ziel zusammengetragen werden, bevor es auf Inferentia-Instances bereitgestellt werden kann. Um fortzufahren, benötigen Sie ein für Neuron optimiertes TensorFlow Modell, das in Amazon S3 gespeichert ist. Falls Sie noch keines haben SavedModel, folgen Sie bitte dem Tutorial zum Erstellen eines Neuron-kompatiblen ResNet 50-Modells und laden Sie das Ergebnis SavedModel auf S3 hoch. ResNet-50 ist ein beliebtes Modell für maschinelles Lernen, das für Bilderkennungsaufgaben verwendet wird. Weitere Informationen zur Kompilierung von Neuron-Modellen finden Sie unter The AWS Inferentia Chip With DLAMI im Deep Learning Developer Guide. AWS AMIs

Das Beispiel-Deployment-Manifest verwaltet einen vorgefertigten Inferenz-Serving-Container, der von AWS Deep Learning Containers TensorFlow bereitgestellt wird. Im Container befinden sich die AWS Neuron Runtime und die TensorFlow Serving-Anwendung. Eine vollständige Liste der vorgefertigten, für Neuron optimierten Deep Learning Containers finden Sie GitHub unter Verfügbare

Images. Beim Start ruft der DLC Ihr Modell von Amazon S3 ab, startet Neuron TensorFlow Serving mit dem gespeicherten Modell und wartet auf Vorhersageanfragen.

Die Anzahl der Neuron-Geräte, die Ihrer Serving-Anwendung zugewiesen sind, kann angepasst werden, indem Sie die `aws.amazon.com/neuron`-Ressource in der Bereitstellung yaml verändern. Bitte beachten Sie, dass die Kommunikation zwischen TensorFlow Serving und der Neuron-Laufzeit über GRPC erfolgt, weshalb die Fähigkeit an den `IPC_LOCK` Container übergeben werden muss.

1. Fügen Sie die `AmazonS3ReadOnlyAccess`-IAM-Richtlinie der Knoten-Instance-Rolle hinzu, die in Schritt 1 unter Cluster erstellen erstellt wurde. Dies ist notwendig, damit die Beispielanwendung ein trainiertes Modell aus Amazon S3 laden kann.

   ```
   aws iam attach-role-policy \
       --policy-arn arn:aws: iam::aws:policy/AmazonS3ReadOnlyAccess \
       --role-name eksctl-inferentia-nodegroup-ng-in-NodeInstanceRole-FI7HIYS3BS09
   ```

2. Erstellen Sie eine Datei mit dem Namen `rn50_deployment.yaml` und dem folgenden Inhalt. Aktualisieren Sie den Regions-Code und den Modellpfad entsprechend den gewünschten Einstellungen. Der Modellname dient zu Identifikationszwecken, wenn ein Client eine Anfrage an den TensorFlow Server stellt. In diesem Beispiel wird ein Modellname verwendet, der einem Beispiel-Clientskript von ResNet 50 entspricht, das in einem späteren Schritt zum Senden von Vorhersageanforderungen verwendet wird.

   ```
   aws ecr list-images --repository-name neuron-rtd --registry-id 790709498068 --region
    us-west-2
   ```

   ```
   kind: Deployment
   apiVersion: apps/v1
   metadata:
     name: eks-neuron-test
     labels:
       app: eks-neuron-test
       role: master
   spec:
     replicas: 2
     selector:
       matchLabels:
         app: eks-neuron-test
         role: master
     template:
   ```

```
    metadata:
      labels:
        app: eks-neuron-test
        role: master
    spec:
      containers:
        - name: eks-neuron-test
          image: 763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-inference-
neuron:1.15.4-neuron-py37-ubuntu18.04
          command:
            - /usr/local/bin/entrypoint.sh
          args:
            - --port=8500
            - --rest_api_port=9000
            - --model_name=resnet50_neuron
            - --model_base_path=s3://${your-bucket-of-models}/resnet50_neuron/
          ports:
            - containerPort: 8500
            - containerPort: 9000
          imagePullPolicy: IfNotPresent
          env:
            - name: AWS_REGION
              value: "us-east-1"
            - name: S3_USE_HTTPS
              value: "1"
            - name: S3_VERIFY_SSL
              value: "0"
            - name: S3_ENDPOINT
              value: s3.us-east-1.amazonaws.com
            - name: AWS_LOG_LEVEL
              value: "3"
          resources:
            limits:
              cpu: 4
              memory: 4Gi
              aws.amazon.com/neuron: 1
            requests:
              cpu: "1"
              memory: 1Gi
          securityContext:
            capabilities:
              add:
                - IPC_LOCK
```

3. Stellen Sie das Modell bereit.

```
kubectl apply -f rn50_deployment.yaml
```

4. Erstellen Sie eine Datei mit dem Namen `rn50_service.yaml` und dem folgenden Inhalt. Die HTTP- und gRPC-Ports werden für die Annahme von Prognoseanforderungen geöffnet.

```
kind: Service
apiVersion: v1
metadata:
  name: eks-neuron-test
  labels:
    app: eks-neuron-test
spec:
  type: ClusterIP
  ports:
    - name: http-tf-serving
      port: 8500
      targetPort: 8500
    - name: grpc-tf-serving
      port: 9000
      targetPort: 9000
  selector:
    app: eks-neuron-test
    role: master
```

5. Erstellen Sie einen Kubernetes-Service für Ihre TensorFlow Model Serving-Anwendung.

```
kubectl apply -f rn50_service.yaml
```

(Optional) Treffen Sie Prognosen für Ihren Serving-Service TensorFlow

1. Um dies lokal zu testen, leiten Sie den gRPC-Port an den Service `eks-neuron-test` weiter.

```
kubectl port-forward service/eks-neuron-test 8500:8500 &
```

2. Erstellen Sie das Python-Skript namens `tensorflow-model-server-infer.py` mit folgendem Inhalt. Dieses Skript führt Inferenz über gRPC aus, einem Service-Framework.

```
import numpy as np
   import grpc
```

```
   import tensorflow as tf
   from tensorflow.keras.preprocessing import image
   from tensorflow.keras.applications.resnet50 import preprocess_input
   from tensorflow_serving.apis import predict_pb2
   from tensorflow_serving.apis import prediction_service_pb2_grpc
   from tensorflow.keras.applications.resnet50 import decode_predictions

   if __name__ == '__main__':
       channel = grpc.insecure_channel('localhost:8500')
       stub = prediction_service_pb2_grpc.PredictionServiceStub(channel)
       img_file = tf.keras.utils.get_file(
           "./kitten_small.jpg",
           "https://raw.githubusercontent.com/awslabs/mxnet-model-server/master/docs/
images/kitten_small.jpg")
       img = image.load_img(img_file, target_size=(224, 224))
       img_array = preprocess_input(image.img_to_array(img)[None, ...])
       request = predict_pb2.PredictRequest()
       request.model_spec.name = 'resnet50_inf1'
       request.inputs['input'].CopyFrom(
           tf.make_tensor_proto(img_array, shape=img_array.shape))
       result = stub.Predict(request)
       prediction = tf.make_ndarray(result.outputs['output'])
       print(decode_predictions(prediction))
```

3. Führen Sie das Skript aus, um Prognosen an den Service zu senden.

```
python3 tensorflow-model-server-infer.py
```

Eine Beispielausgabe sieht wie folgt aus.

```
[[(u'n02123045', u'tabby', 0.68817204), (u'n02127052', u'lynx', 0.12701613),
 (u'n02123159', u'tiger_cat', 0.08736559), (u'n02124075', u'Egyptian_cat',
 0.063844085), (u'n02128757', u'snow_leopard', 0.009240591)]]
```

Rechenressourcen für AI/ML Workloads auf Amazon EKS verwalten

Dieser Abschnitt soll Ihnen bei der Verwaltung von Rechenressourcen für Machine-Learning-Workloads in Amazon Elastic Kubernetes Service (EKS) behilflich sein. Sie finden Einzelheiten zur Reservierung GPUs mithilfe von Kapazitätsblöcken für verwaltete Knotengruppen und

selbstverwaltete Knoten, einschließlich der Voraussetzungen, der Einrichtung der Startvorlage, der Skalierungskonfigurationen, der Vorbereitung der Arbeitslast und der wichtigsten Überlegungen zum Umgang mit Reservierungszyklen und der ordnungsgemäßen Kündigung von Knoten.

Themen

- Erstellung einer verwalteten Knotengruppe mit Kapazitätsblöcken für ML
- Erstellung selbstverwalteter Knoten mit Kapazitätsblöcken für ML
- Verwenden Sie GB2 P6e-00 UltraServers mit Amazon EKS

Erstellung einer verwalteten Knotengruppe mit Kapazitätsblöcken für ML

Kapazitätsblöcke für Machine Learning (ML) ermöglichen es Ihnen, GPU-Instances zu einem zukünftigen Zeitpunkt zu reservieren, um Ihre ML-Workloads mit kurzer Dauer zu unterstützen. Weitere Informationen finden Sie unter Kapazitätsblöcke für ML im Amazon-EC2-Benutzerhandbuch für Linux-Instances.

Überlegungen

⚠ Important

- Kapazitätsblöcke sind nur für bestimmte Amazon-EC2-Instance-Typen und AWS-Regionen verfügbar. Informationen zur Kompatibilität finden Sie unter Arbeiten mit Kapazitätsblöcken – Voraussetzungen im Amazon-EC2-Benutzerhandbuch für Linux-Instances.
- Weitere Informationen finden Sie unter Verwenden von Kapazitätsblöcken für Machine Learning im Benutzerhandbuch für Amazon EC2 Auto Scaling.
- Verwaltete Knotengruppen mit Kapazitätsblöcken können nur mit benutzerdefinierten Startvorlagen erstellt werden.
- Stellen Sie beim Upgrade verwalteter Knotengruppen mit Kapazitätsblöcken sicher, dass die gewünschte Größe der Knotengruppe auf `0` eingestellt ist.

Erstellung einer verwalteten Knotengruppe mit Amazon-EC2-Kapazitätsblöcken

Sie können Kapazitätsblöcke mit von Amazon EKS verwalteten Knotengruppen zum Bereitstellen und Skalieren von GPU-beschleunigten Worker-Knoten verwenden. Die folgenden Beispiele für AWS-

CloudFormation-Vorlagen decken nicht alle Aspekte ab, die in Produktions-Clustern erforderlich sind. Normalerweise benötigen Sie auch ein Bootstrapping-Skript, um den Knoten mit dem Cluster zu verbinden und ein beschleunigtes Amazon-EKS-AMI anzugeben. Weitere Informationen finden Sie unter the section called “Create”.

1. Erstellen Sie eine Startvorlage, die für Ihre Workloads geeignet ist und mit von Amazon EKS verwalteten Knotengruppen funktioniert. Weitere Informationen finden Sie unter the section called “Startvorlagen”.

   Stellen Sie zusätzlich zu den Anforderungen in den oben genannten Verfahren sicher, dass `LaunchTemplateData` Folgendes enthält:

   - `InstanceMarketOptions` mit der Einstellung `"capacity-block"` für `MarketType`
   - `CapacityReservationSpecification: CapacityReservationTarget` mit `CapacityReservationId` als Kapazitätsblock (zum Beispiel: `cr-`*`02168da1478b509e0`*)
   - `InstanceType` auf einen Instance-Typ eingestellt, der Kapazitätsblöcke unterstützt (zum Beispiel: *`p5.48xlarge`*)

   Nachfolgend finden Sie einen Auszug aus einer CloudFormation-Vorlage, die eine Startvorlage für einen Kapazitätsblock erstellt. Um eine benutzerdefinierte AMI-verwaltete Knotengruppe zu erstellen, können Sie auch `ImageId`- und `UserData`-Parameter hinzufügen.

   ```
   NodeLaunchTemplate:
     Type: "AWS::EC2::LaunchTemplate"
     Properties:
       LaunchTemplateData:
         InstanceMarketOptions:
           MarketType: "capacity-block"
         CapacityReservationSpecification:
           CapacityReservationTarget:
             CapacityReservationId: "cr-02168da1478b509e0"
         InstanceType: p5.48xlarge
   ```

2. Verwenden Sie die Startvorlage, um eine verwaltete Knotengruppe zu erstellen.

   Nachfolgend finden Sie ein Beispiel für den Befehl zum Erstellen einer Knotengruppe für Kapazitätsblöcke. Ersetzen Sie die *`Beispielwerte`* durch die für Ihren Cluster geltenden Werte.

   Gehen Sie beim Erstellen der verwalteten Knotengruppe für Kapazitätsblöcke wie folgt vor:

- Legen Sie den Wert für `capacity-type` auf `"CAPACITY_BLOCK"` fest. Wenn der Kapazitätstyp nicht auf `"CAPACITY_BLOCK"` gesetzt ist oder einer der anderen oben genannten erforderlichen Startvorlagenwerte fehlt, wird die Erstellungsanfrage abgelehnt.
- Achten Sie beim Angeben von `subnets` in der Erstellungsanfrage darauf, nur das Subnetz in derselben Availability Zone wie die Kapazitätsreservierung anzugeben.
- Wenn Sie in der Erstellungsanforderung einen `desiredSize` ungleich Null angeben, berücksichtigt Amazon EKS dies beim Erstellen der Auto Scaling-Gruppe (ASG). Erfolgt die Erstellungsanfrage jedoch vor Aktivierung der Kapazitätsreservierung, kann die ASG keine Amazon-EC2-Instances starten, bis sie aktiviert wird. Dies führt dazu, dass bei ASG-Skalierungsaktivitäten Startfehler auftreten. Sobald die Reservierung aktiv wird, ist der Start der Instances erfolgreich und die ASG wird auf die bei der Erstellung angegebene `desiredSize` skaliert.

```
aws eks create-nodegroup \
    --cluster-name my-cluster \
    --nodegroup-name my-mng \
    --node-role node-role-arn \
    --region region-code \
    --subnets subnet-id \
    --scaling-config minSize=node-group-min-size,maxSize=node-group-max-
size,desiredSize=node-group-desired-size \
    --ami-type "AL2023_x86_64_NVIDIA" \
    --capacity-type "CAPACITY_BLOCK" \
    --launch-template id="lt-id",version=1
```

3. Stellen Sie sicher, dass die Knoten nach der Hochskalierung hinzugefügt werden. Amazon-EKS-Cluster, die verwaltete Knotengruppen mit Kapazitätsblöcken verwenden, führen keine Überprüfungen durch, ob die gestarteten Instances tatsächlich zum Cluster hinzugefügt und dort registriert werden.
4. Wenn Sie `desiredSize` beim Erstellen auf `0` setzen, haben Sie verschiedene Möglichkeiten, die Knotengruppe hoch zu skalieren, wenn die Kapazitätsreservierung aktiv wird:
   - Erstellen Sie eine geplante Skalierungsrichtlinie für die ASG, die mit dem Startzeitpunkt der Kapazitätsblock-Reservierung übereinstimmt. Weitere Informationen finden Sie unter Geplante Skalierung für Amazon EC2 Auto Scaling im Amazon EC2 Auto Scaling-Benutzerhandbuch.
   - Verwenden Sie die Amazon-EKS-Konsole oder `eks update-nodegroup-config`, um die Skalierungskonfiguration zu aktualisieren und die gewünschte Größe der Knotengruppe festzulegen.

- Verwenden Sie den Kubernetes Cluster Autoscaler. Weitere Informationen finden Sie unter Cluster Autoscaler on AWS.

5. Die Knotengruppe ist jetzt bereit für die Planung von Workloads und Pods.
6. Damit Ihre Pods vor Ablauf der Reservierung ordnungsgemäß entleert werden, verwendet Amazon EKS eine geplante Skalierungsrichtlinie, um die Größe der Knotengruppe auf `0` herunterzuskalieren. Diese geplante Skalierung wird mit dem Namen `Amazon EKS Node Group Capacity Scaledown Before Reservation End` festgelegt. Wir empfehlen, diese Aktion nicht zu bearbeiten oder zu löschen.

   Amazon EC2 beginnt 30 Minuten vor Ablauf der Reservierungszeit mit dem Herunterfahren der Instances. Daher richtet Amazon EKS 40 Minuten vor dem Ende der Reservierung eine geplante Herunterskalierung der Knotengruppe ein, um Pods sicher und ordnungsgemäß zu entfernen.

Erstellung selbstverwalteter Knoten mit Kapazitätsblöcken für ML

Kapazitätsblöcke für Machine Learning (ML) ermöglichen es Ihnen, GPU-Instances zu einem zukünftigen Zeitpunkt zu reservieren, um Ihre ML-Workloads mit kurzer Dauer zu unterstützen. Weitere Informationen finden Sie unter Kapazitätsblöcke für ML im Amazon-EC2-Benutzerhandbuch für Linux-Instances.

Überlegungen

Important

- Kapazitätsblöcke sind nur für bestimmte Amazon-EC2-Instance-Typen und AWS-Regionen verfügbar. Informationen zur Kompatibilität finden Sie unter Arbeiten mit Kapazitätsblöcken – Voraussetzungen im Amazon-EC2-Benutzerhandbuch für Linux-Instances.
- Wenn Sie eine selbstverwaltete Knotengruppe erstellen, bevor die Kapazitätsreservierung aktiv wird, legen Sie die gewünschte Kapazität auf `0` fest.
- Um genügend Zeit zu haben, den/die Knoten ordnungsgemäß zu entleeren, empfehlen wir, die Skalierung so zu planen, dass sie mehr als 30 Minuten vor der Endzeit der Kapazitätsblock-Reservierung auf Null skaliert.
- Damit Ihre Pods ordnungsgemäß entladen werden, empfehlen wir Ihnen, den AWS Node Termination Handler wie in den Beispielschritten beschrieben einzurichten.

Verwendung von Kapazitätsblöcken mit selbstverwalteten Knoten

Sie können Kapazitätsblöcke mit Amazon EKS für die Bereitstellung und Skalierung Ihrer selbstverwalteten Knoten verwenden. Die folgenden Schritte geben einen allgemeinen Beispielüberblick. Die Beispiele für AWS CloudFormation-Vorlagen decken nicht alle Aspekte ab, die in einem Produktions-Workload erforderlich sind. Normalerweise benötigen Sie außerdem ein Bootstrapping-Skript, um den Knoten mit dem Cluster zu verbinden, ein beschleunigtes Amazon-EKS-AMI und ein entsprechendes Instance-Profil für den Beitritt zum Cluster anzugeben. Weitere Informationen finden Sie unter the section called “Amazon Linux”.

1. Erstellen Sie eine Startvorlage, die auf Ihre Workload anwendbar ist. Weitere Informationen finden Sie unter Verwenden von Kapazitätsblöcken für Machine Learning im Benutzerhandbuch für Amazon EC2 Auto Scaling.

   Stellen Sie sicher, dass `LaunchTemplateData` Folgendes enthält:

   - `InstanceMarketOptions` mit der Einstellung `"capacity-block"` für `MarketType`
   - `CapacityReservationSpecification: CapacityReservationTarget` mit `CapacityReservationId` als Kapazitätsblock (zum Beispiel: `cr-`*`02168da1478b509e0`*)
   - `IamInstanceProfile` mit `Arn` auf das entsprechende *`iam-instance-profile-arn`* eingestellt
   - `ImageId` auf das entsprechende *`image-id`* eingestellt
   - `InstanceType` auf einen Instance-Typ eingestellt, der Kapazitätsblöcke unterstützt (zum Beispiel: *`p5.48xlarge`*)
   - `SecurityGroupIds` auf die entsprechenden IDs eingestellt (zum Beispiel: *`sg-05b1d815d1EXAMPLE`*)
   - `UserData` auf die entsprechende *`user-data`* für Ihre selbstverwaltete Knotengruppe eingestellt

   Nachfolgend finden Sie einen Auszug aus einer CloudFormation-Vorlage, die eine Startvorlage für einen Kapazitätsblock erstellt.

   ```
   NodeLaunchTemplate:
     Type: "aws::EC2::LaunchTemplate"
     Properties:
       LaunchTemplateData:
         InstanceMarketOptions:
           MarketType: "capacity-block"
   ```

```
      CapacityReservationSpecification:
        CapacityReservationTarget:
          CapacityReservationId: "cr-02168da1478b509e0"
      IamInstanceProfile:
        Arn: iam-instance-profile-arn
      ImageId: image-id
      InstanceType: p5.48xlarge
      KeyName: key-name
      SecurityGroupIds:
      - sg-05b1d815d1EXAMPLE
      UserData: user-data
```

Sie müssen das Subnetz in der Availability Zone übergeben, in der die Reservierung vorgenommen wurde, da Kapazitätsblöcke zonal sind.

2. Verwenden Sie die Startvorlage, um eine selbstverwaltete Knotengruppe zu erstellen. Wenn Sie dies vor der Aktivierung der Kapazitätsreservierung tun, legen Sie die gewünschte Kapazität auf `0` fest. Achten Sie beim Erstellen der Knotengruppe darauf, dass Sie nur das entsprechende Subnetz für die Availability Zone angeben, in der die Kapazität reserviert ist.

   Nachfolgend finden Sie ein Beispiel für eine CloudFormation-Vorlage, die Sie als Referenz verwenden können, wenn Sie eine für Ihre Workload geeignete Vorlage erstellen. In diesem Beispiel werden die `LaunchTemplateId` und das `Version` der im vorherigen Beispiel gezeigten `AWS::Amazon EC2::LaunchTemplate`-Ressource ermittelt. Sie erhält auch die Werte für `DesiredCapacity`, `MaxSize`, `MinSize`, und `VPCZoneIdentifier`, die an anderer Stelle in derselben Vorlage deklariert sind.

```
NodeGroup:
  Type: "AWS::AutoScaling::AutoScalingGroup"
  Properties:
    DesiredCapacity: !Ref NodeAutoScalingGroupDesiredCapacity
    LaunchTemplate:
      LaunchTemplateId: !Ref NodeLaunchTemplate
      Version: !GetAtt NodeLaunchTemplate.LatestVersionNumber
    MaxSize: !Ref NodeAutoScalingGroupMaxSize
    MinSize: !Ref NodeAutoScalingGroupMinSize
    VPCZoneIdentifier: !Ref Subnets
    Tags:
      - Key: Name
        PropagateAtLaunch: true
        Value: !Sub ${ClusterName}-${NodeGroupName}-Node
      - Key: !Sub kubernetes.io/cluster/${ClusterName}
```

```
        PropagateAtLaunch: true
        Value: owned
```

3. Nachdem die Knotengruppe erfolgreich erstellt wurde, stellen Sie sicher, dass Sie die `NodeInstanceRole` für die Knotengruppe aufzeichnen, die erstellt wurde. Dies ist erforderlich, um sicherzustellen, dass bei einer Skalierung der Knotengruppe die neuen Knoten dem Cluster beitreten und Kubernetes die Knoten erkennen kann. Weitere Informationen finden Sie in den AWS-Managementkonsole-Anweisungen unter Erstellen selbstverwalteter Amazon Linux-Knoten.
4. Wir empfehlen Ihnen, eine geplante Skalierungsrichtlinie für die Auto-Scaling-Gruppe zu erstellen, die sich an den Reservierungszeiten für Kapazitätsblöcke orientiert. Weitere Informationen finden Sie unter Geplante Skalierung für Amazon EC2 Auto Scaling im Amazon EC2 Auto Scaling-Benutzerhandbuch.

   Sie können alle von Ihnen reservierten Instances bis 30 Minuten vor Ablauf der Endzeit des Kapazitätsblocks verwenden. Instances, die zu diesem Zeitpunkt noch laufen, werden beendet. Um genügend Zeit zu haben, den/die Knoten ordnungsgemäß zu entleeren, empfehlen wir, die Skalierung so zu planen, dass sie mehr als 30 Minuten vor der Endzeit der Kapazitätsblock-Reservierung auf Null skaliert.

   Wenn Sie stattdessen manuell hochskalieren wollen, wenn die Kapazitätsreservierung `Active` wird, müssen Sie die gewünschte Kapazität der Auto-Scaling-Gruppe zum Startzeitpunkt der Kapazitätsblock-Reservierung aktualisieren. Dann müssten Sie auch mehr als 30 Minuten vor dem Ende der Kapazitätsblock-Reservierung manuell herunterskalieren.
5. Die Knotengruppe ist jetzt bereit für die Planung von Workloads und Pods.
6. Damit Ihre Pods ordnungsgemäß entleert werden kann, empfehlen wir Ihnen, AWS Node Termination Handler einzurichten. Dieser Handler kann mithilfe von EventBridge nach „ASG Scale-in“-Lebenszyklusereignissen von Amazon EC2 Auto Scaling Ausschau halten und es der Kubernetes-Steuerebene ermöglichen, die erforderlichen Maßnahmen zu ergreifen, bevor die Instance nicht verfügbar ist. Andernfalls bleiben Ihre Pods und Kubernetes-Objekte in einem ausstehenden Zustand hängen. Weitere Informationen finden Sie unter AWS Node Termination Handler auf GitHub.

   Wenn Sie keinen Node Termination Handler einrichten, empfehlen wir Ihnen, mit der manuellen Entleerung Ihrer Pods zu beginnen, bevor das 30-Minuten-Fenster erreicht ist, damit sie genügend Zeit haben, um ordnungsgemäß entleert zu werden.

Verwenden Sie GB2 P6e-00 UltraServers mit Amazon EKS

In diesem Thema wird beschrieben, wie Amazon EKS mit P6e-00 konfiguriert und verwendet wird. GB2 UltraServers Der `p6e-gb200.36xlarge` Instance-Typ mit 4 NVIDIA Blackwell GPUs ist nur als P6e-00 verfügbar. GB2 UltraServers Es gibt zwei Typen von P6e- 00. GB2 UltraServers Der `u-p6e-gb200x36` UltraServer hat 9 `p6e-gb200.36xlarge` Instanzen und der `u-p6e-gb200x72` UltraServer hat 18 `p6e-gb200.36xlarge` Instanzen.

Weitere Informationen finden Sie auf der Amazon EC2 GB2 UltraServers P6e-00-Webseite.

Überlegungen

- Amazon EKS unterstützt GB2 P6e-00 UltraServers für Kubernetes-Versionen 1.33 und höher. Diese Version von Kubernetes bietet Unterstützung für Dynamic Resource Allocation (DRA), die standardmäßig in EKS und in der Version 023 EKS-Optimized Accelerated aktiviert ist. AL2 AMIs DRA ist eine Voraussetzung für die Verwendung des P6e-00 mit EKS. GB2 UltraServers DRA wird im automatischen Modus von Karpenter oder EKS nicht unterstützt, und es wird empfohlen, selbstverwaltete EKS-Knotengruppen oder EKS-verwaltete Knotengruppen zu verwenden, wenn Sie den P6e-00 mit EKS verwenden. GB2 UltraServers
- GB2P6e-00 UltraServers werden über Capacity Blocks for ML zur Verfügung gestellt. EC2 Informationen the section called “Kapazitätsverwaltung” zum Starten von EKS-Knoten mit Capacity-Blöcken finden Sie unter.
- Wenn Sie von EKS verwaltete Knotengruppen mit Kapazitätsblöcken verwenden, müssen Sie benutzerdefinierte Startvorlagen verwenden. Wenn Sie von EKS verwaltete Knotengruppen mit GB2 P6e-00 aktualisieren UltraServers, müssen Sie `0` vor dem Upgrade die gewünschte Größe der Knotengruppe auf einstellen.
- Es wird empfohlen, die AL2 023 ARM NVIDIA-Variante des EKS-optimierten Beschleunigers zu verwenden. AMIs Dieses AMI umfasst die erforderlichen Knotenkomponenten und die Konfiguration für die Verwendung mit P6e-00. GB2 UltraServers Wenn Sie sich entscheiden, Ihr eigenes AMI zu erstellen, sind Sie für die Installation und Überprüfung der Kompatibilität der Knoten- und Systemsoftware, einschließlich der Treiber, verantwortlich. Weitere Informationen finden Sie unter the section called “Verwenden Sie die EKS-Linux-GPU AMIs”.
- Es wird empfohlen, die EKS-optimierte AMI-Version `v20251103` oder höher zu verwenden, die die NVIDIA-Treiberversion 580 enthält. Diese NVIDIA-Treiberversion ermöglicht Coherent Driver-Based Memory (CDMM), um potenziellen Speicherüberschüssen entgegenzuwirken. Wenn CDMM aktiviert ist, werden die folgenden Funktionen nicht unterstützt: NVIDIA Multi-Instance-GPU (MIG)

und vGPU. Weitere Informationen zu CDMM finden Sie unter NVIDIA Coherent Driver-based Memory Management (CDMM).

- Wenn Sie den NVIDIA-GPU-Operator mit dem EKS-optimierten AL2 023 NVIDIA AMI verwenden, müssen Sie die Operatorinstallation des Treibers und des Toolkits deaktivieren, da diese bereits im AMI enthalten sind. Die für EKS optimierten AL2 023 NVIDIA enthalten AMIs weder das NVIDIA Kubernetes-Geräte-Plugin noch den NVIDIA DRA-Treiber, und diese müssen separat installiert werden.
- Jede `p6e-gb200.36xlarge` Instanz kann mit bis zu 17 Netzwerkkarten konfiguriert werden und EFA für die Kommunikation zwischen ihnen nutzen. UltraServers Der Netzwerkverkehr kann Workloads durchqueren UltraServers, aber für eine optimale Leistung wird empfohlen, Workloads nach demselben Schema zu planen und dabei IMEX für die GPU-interne Kommunikation zu UltraServer nutzen. UltraServer Weitere Informationen finden Sie unter EFA-Konfiguration für P6e-00-Instances. GB2
- Jede `p6e-gb200.36xlarge` Instance verfügt über 3 x 7,5 TB Instance-Speicher. Standardmäßig formatiert und mountet das EKS-optimierte AMI die Instance-Speicher nicht. Der kurzlebige Speicher des Knotens kann von Pods gemeinsam genutzt werden, die kurzlebigen Speicher anfordern, und von Container-Images, die auf den Knoten heruntergeladen werden. Wenn Sie das AL2 023 EKS-optimierte AMI verwenden, kann dies als Teil des Knoten-Bootstraps in den Benutzerdaten konfiguriert werden, indem die lokale Speicherrichtlinie der Instanz auf RAID0 gesetzt wird. NodeConfig Bei Einstellung auf RAID0 speichert und konfiguriert die Instance die Container-Runtime und das Kubelet, um diesen kurzlebigen Speicher zu nutzen.

Komponenten

Die folgenden Komponenten werden für die Ausführung von Workloads auf EKS mit dem P6e-00 empfohlen. GB2 UltraServers Sie können optional den NVIDIA-GPU-Operator verwenden, um die NVIDIA-Knotenkomponenten zu installieren. Wenn Sie den NVIDIA-GPU-Operator mit dem EKS-optimierten AL2 023 NVIDIA AMI verwenden, müssen Sie die Operatorinstallation des Treibers und des Toolkits deaktivieren, da diese bereits im AMI enthalten sind.

Stack	Komponente
EKS-optimiertes beschleunigtes AMI	Kernel 6.12
	NVIDIA-GPU-Treiber

Stack	Komponente
	NVIDIA CUDA-Benutzermodus-Treiber
	NVIDIA-Container-Toolkit
	NVIDIA-Fabric-Manager
	NVIDIA IMEX-Treiber
	NVLink NVIDIA-Subnetzmanager
	EFA-Treiber
Komponenten, die auf dem Knoten ausgeführt werden	VPC CNI
	EFA-Geräte-Plugin
	NVIDIA K8s-Geräte-Plugin
	NVIDIA DRA-Treiber
	Erkennung von NVIDIA-Knotenfunktionen (NFD)
	Erkennung von NVIDIA-GPU-Features (GFD)

Die Knotenkomponenten in der obigen Tabelle erfüllen die folgenden Funktionen:

- VPC CNI: Weist VPC IPs als primäre Netzwerkschnittstelle für Pods zu, die auf EKS ausgeführt werden
- EFA-Geräte-Plugin: Weist EFA-Geräte als sekundäre Netzwerke für Pods zu, die auf EKS ausgeführt werden. Verantwortlich für den Netzwerkverkehr über P6e-00. GB2 UltraServers Für Workloads mit mehreren Knoten, d. h. GPU-to-GPU innerhalb eines UltraServer Kanals, der über mehrere Knoten fließt. NVLink
- NVIDIA Kubernetes-Geräte-Plugin: Weist Pods, die auf EKS laufen, GPUs als Geräte zu. Es wird empfohlen, das NVIDIA Kubernetes-Geräte-Plugin zu verwenden, bis die GPU-Zuweisungsfunktion des NVIDIA DRA-Treibers den experimentellen Status erreicht hat. Aktuelle Informationen finden Sie in den NVIDIA-DRA-Treiberversionen.

- NVIDIA DRA-Treiber: Ermöglicht ComputeDomain benutzerdefinierte Ressourcen, die die Erstellung von IMEX-Domänen erleichtern, die Workloads folgen, die auf P6e-00 ausgeführt werden. GB2 UltraServers
  - Die ComputeDomain Ressource beschreibt eine IMEX-Domäne (Internode Memory Exchange). Wenn Workloads mit einem ResourceClaim for a auf dem Cluster bereitgestellt ComputeDomain werden, erstellt der NVIDIA DRA-Treiber automatisch ein IMEX, DaemonSet das auf passenden Knoten ausgeführt wird, und richtet die IMEX-Kanäle zwischen den Knoten ein, bevor der Workload gestartet wird. Weitere Informationen zu IMEX finden Sie in der Übersicht über NVIDIA IMEX für Systeme mit mehreren Knoten. NVLink
  - Der NVIDIA-DRA-Treiber verwendet ein von NVIDIA GFD verwendetes Clique-ID-Label (`nvidia.com/gpu.clique`), das Informationen über die Netzwerktopologie und -domäne weitergibt. NVLink
  - Es hat sich bewährt, einen Job pro Workload zu erstellen ComputeDomain .
- NVIDIA Node Feature Discovery (NFD): Erforderliche Abhängigkeit für GFD, um Knotenbezeichnungen auf der Grundlage der erkannten Attribute auf Knotenebene anzuwenden.
- NVIDIA GPU Feature Discovery (GFD): Wendet ein NVIDIA-Standard-Topologie-Label an, das auf die Knoten aufgerufen wird. `nvidia.com/gpu.clique` Knoten innerhalb desselben Systems `nvidia.com/gpu.clique` sind über mehrere Knoten erreichbar NVLink, und Sie können Pod-Affinitäten in Ihrer Anwendung verwenden, um Pods für dieselbe Domain zu planen. NVlink

Verfahren

Im folgenden Abschnitt wird davon ausgegangen, dass Sie über einen EKS-Cluster verfügen, auf dem Kubernetes Version 1.33 oder höher ausgeführt wird, mit einer oder mehreren Knotengruppen mit GB2 P6e-00, auf dem das AL2 023 ARM NVIDIA EKS-optimierte beschleunigte AMI UltraServers ausgeführt wird. Die erforderlichen Schritte für selbstverwaltete EKS-Knoten und the section called “Kapazitätsverwaltung” verwaltete Knotengruppen finden Sie unter den Links unter.

Das folgende Verfahren verwendet die folgenden Komponenten.

Name	Version	Description
NVIDIA-GPU-Betreiber	25.3.4+	Für das Lebenszyklusmanagement erforderlicher Plug-ins wie NVIDIA Kubernetes Device Plugin und NFD/GFD.

Name	Version	Description
NVIDIA DRA-Treiber	25.8.0 oder höher	Für ComputeDomain CRDs und IMEX-Domainverwaltung.
EFA-Geräte-Plugin	0.5.14+	Für die übergreifende Kommunikation. UltraServer

Installieren Sie den NVIDIA GPU-Operator

Der NVIDIA-GPU-Operator vereinfacht die Verwaltung der Komponenten, die für die Verwendung GPUs in Kubernetes-Clustern erforderlich sind. Da der NVIDIA-GPU-Treiber und das Container-Toolkit als Teil des EKS-optimierten beschleunigten AMI installiert werden, müssen diese `false` in der Konfiguration der Helm-Werte auf eingestellt werden.

1. Erstellen Sie eine Helm-Wertedatei `gpu-operator-values.yaml` mit dem Namen der folgenden Konfiguration.

```
devicePlugin:
  enabled: true
nfd:
  enabled: true
gfd:
  enabled: true
driver:
  enabled: false
toolkit:
  enabled: false
migManager:
  enabled: false
```

2. Installieren Sie den NVIDIA-GPU-Operator für Ihren Cluster mithilfe der `gpu-operator-values.yaml` Datei, die Sie im vorherigen Schritt erstellt haben.

```
helm repo add nvidia https://helm.ngc.nvidia.com/nvidia
helm repo update
```

```
helm install gpu-operator nvidia/gpu-operator \
 --namespace gpu-operator \
 --create-namespace \
```

```
 --version v25.3.4 \
 --values gpu-operator-values.yaml
```

Installieren Sie den NVIDIA DRA-Treiber

Ab der NVIDIA-GPU-Operatorversion `v25.3.4` muss der NVIDIA-DRA-Treiber separat installiert werden. Es wird empfohlen, die Versionshinweise zum NVIDIA-GPU-Operator zu lesen, da sich dies in einer future Version ändern kann.

1. Erstellen Sie eine Helm-Wertedatei `dra-values.yaml` mit dem Namen der folgenden Konfiguration. Beachten Sie das `nodeAffinity` und `tolerations` das konfiguriert den DRA-Treiber so, dass er nur auf Knoten mit einer NVIDIA-GPU bereitgestellt wird.

```
resources:
  gpus:
    enabled: false # set to false to disable experimental gpu support
  computeDomains:
    enabled: true

controller:
  nodeSelector: null
  affinity: null
  tolerations: []

kubeletPlugin:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: "nvidia.com/gpu.present"
            operator: In
            values:
            - "true"
  tolerations:
    - key: "nvidia.com/gpu"
      operator: Exists
      effect: NoSchedule
```

2. Installieren Sie den NVIDIA-DRA-Treiber für Ihren Cluster mithilfe der `dra-values.yaml` Datei, die Sie im vorherigen Schritt erstellt haben.

```
helm repo add nvidia https://helm.ngc.nvidia.com/nvidia
helm repo update
```

```
helm install nvidia-dra-driver-gpu nvidia/nvidia-dra-driver-gpu \
  --version="25.8.0" \
  --namespace nvidia-dra-driver-gpu \
  --create-namespace \
  -f dra-values.yaml
```

3. Nach der Installation erstellt der DRA-Treiber `DeviceClass` Ressourcen, die es Kubernetes ermöglichen, Ressourcen zu verstehen und zuzuweisen`ComputeDomain`, sodass das IMEX-Management für verteilte GPU-Workloads auf P6e-00 möglich ist. GB2 UltraServers

   Stellen Sie mit den folgenden Befehlen sicher, dass die DRA-Ressourcen verfügbar sind.

```
kubectl api-resources | grep resource.k8s.io
```

```
deviceclasses           resource.k8s.io/v1  false        DeviceClass
resourceclaims          resource.k8s.io/v1  true         ResourceClaim
resourceclaimtemplates  resource.k8s.io/v1  true         ResourceClaimTemplate
resourceslices          resource.k8s.io/v1  false        ResourceSlice
```

```
kubectl get deviceclasses
```

```
NAME
compute-domain-daemon.nvidia.com
compute-domain-default-channel.nvidia.com
```

Installieren Sie das EFA-Geräte-Plugin

Um die EFA-Kommunikation zwischen nutzen zu können UltraServers, müssen Sie das Kubernetes-Geräte-Plugin für EFA installieren. GB2P6e-00-Instances können mit bis zu 17 Netzwerkkarten konfiguriert werden, und das primäre NCI (Index 0) muss vom Typ sein und bis zu 100 Gbit/s `interface` ENA-Bandbreite unterstützen. Konfigurieren Sie Ihre EFA- und ENA-Schnittstellen bei der Knotenbereitstellung gemäß Ihren Anforderungen. Weitere Informationen zur EFA-Konfiguration finden Sie in der AWS Dokumentation zur EFA-Konfiguration für GB2 P6e-00-Instanzen.

1. Erstellen Sie eine Helm-Wertedatei `efa-values.yaml` mit dem Namen der folgenden Konfiguration.

```
tolerations:
  - key: nvidia.com/gpu
    operator: Exists
    effect: NoSchedule
```

2. Installieren Sie den NVIDIA DRA-Operator für Ihren Cluster mithilfe der `dra-values.yaml` Datei, die Sie im vorherigen Schritt erstellt haben.

```
helm repo add eks https://aws.github.io/eks-charts
helm repo update
```

```
helm install efa eks/aws-efa-k8s-device-plugin -n kube-system \
  --version="0.5.14" \
  -f efa-values.yaml
```

   Wenn Sie Ihre Instances beispielsweise mit einer reinen EFA-Schnittstelle in jeder NCI-Gruppe konfiguriert haben, wird bei der Beschreibung eines Knotens davon ausgegangen, dass pro Knoten 4 zuweisbare EFA-Geräte vorhanden sind.

```
kubectl describe node/<gb200-node-name>
```

```
Capacity:
  ...
  vpc.amazonaws.com/efa:  4
Allocatable:
  ...
  vpc.amazonaws.com/efa:  4
```

Validieren Sie IMEX über Multi-Node NVLink

Einen NVLINK-NCCL-Test mit mehreren Knoten und andere Mikro-Benchmarks finden Sie im Repository. awesome-distributed-training GitHub Die folgenden Schritte zeigen, wie Sie einen Test mit mehreren Knoten mit nvbandwidth durchführen. NVLink

1. Um einen Bandbreitentest mit mehreren Knoten auf zwei Knoten in der NVL72 Domäne durchzuführen, installieren Sie zunächst den MPI-Operator:

```
kubectl create -f https://github.com/kubeflow/mpi-operator/releases/download/v0.7.0/
mpi-operator.yaml
```

2. Erstellen Sie eine Helm-Wertedatei mit dem Namen`nvbandwidth-test-job.yaml`, die das Testmanifest definiert. Beachten Sie die `nvidia.com/gpu.clique` Pod-Affinität, die Worker in derselben NVLink Domäne einzuplanen, die über mehrere Knoten erreichbar NVLink ist.

   Ab der NVIDIA DRA-Treiberversion `v25.8.0` ComputeDomains sind sie elastisch und `.spec.numNodes` können in der Definition auf `0` eingestellt werden. ComputeDomain Informationen zu Updates finden Sie in den neuesten Versionshinweisen zum NVIDIA DRA-Treiber.

```
---
apiVersion: resource.nvidia.com/v1beta1
kind: ComputeDomain
metadata:
  name: nvbandwidth-test-compute-domain
spec:
  numNodes: 0 # This can be set to 0 from NVIDIA DRA Driver version v25.8.0+
  channel:
    resourceClaimTemplate:
      name: nvbandwidth-test-compute-domain-channel

---
apiVersion: kubeflow.org/v2beta1
kind: MPIJob
metadata:
  name: nvbandwidth-test
spec:
  slotsPerWorker: 4 # 4 GPUs per worker node
  launcherCreationPolicy: WaitForWorkersReady
  runPolicy:
    cleanPodPolicy: Running
  sshAuthMountPath: /home/mpiuser/.ssh
  mpiReplicaSpecs:
    Launcher:
      replicas: 1
      template:
        metadata:
```

```
        labels:
          nvbandwidth-test-replica: mpi-launcher
      spec:
        containers:
        - image: ghcr.io/nvidia/k8s-samples:nvbandwidth-v0.7-8d103163
          name: mpi-launcher
          securityContext:
            runAsUser: 1000
          command:
          - mpirun
          args:
          - --bind-to
          - core
          - --map-by
          - ppr:4:node
          - -np
          - "8"
          - --report-bindings
          - -q
          - nvbandwidth
          - -t
          - multinode_device_to_device_memcpy_read_ce
  Worker:
    replicas: 2 # 2 worker nodes
    template:
      metadata:
        labels:
          nvbandwidth-test-replica: mpi-worker
      spec:
        affinity:
          podAffinity:
            requiredDuringSchedulingIgnoredDuringExecution:
            - labelSelector:
                matchExpressions:
                - key: nvbandwidth-test-replica
                  operator: In
                  values:
                  - mpi-worker
              topologyKey: nvidia.com/gpu.clique
        containers:
        - image: ghcr.io/nvidia/k8s-samples:nvbandwidth-v0.7-8d103163
          name: mpi-worker
          securityContext:
            runAsUser: 1000
```

```
            env:
            command:
            - /usr/sbin/sshd
            args:
            - -De
            - -f
            - /home/mpiuser/.sshd_config
            resources:
              limits:
                nvidia.com/gpu: 4  # Request 4 GPUs per worker
              claims:
              - name: compute-domain-channel # Link to IMEX channel
          resourceClaims:
          - name: compute-domain-channel
            resourceClaimTemplateName: nvbandwidth-test-compute-domain-channel
```

3. Erstellen Sie den Job ComputeDomain und starten Sie ihn mit dem folgenden Befehl.

```
kubectl apply -f nvbandwidth-test-job.yaml
```

4. ComputeDomain Bei der Erstellung können Sie sehen, dass der ComputeDomain Workload aus zwei Knoten besteht:

```
kubectl get computedomains.resource.nvidia.com -o yaml
```

```
status:
  nodes:
  - cliqueID: <ClusterUUID>.<Clique ID>
    ipAddress: <node-ip>
    name: <node-hostname>
  - cliqueID: <ClusterUUID>.<Clique ID>
    ipAddress: <node-ip>
    name: <node-hostname>
  status: Ready
```

5. Überprüfen Sie die Ergebnisse des Jobs mit dem folgenden Befehl.

```
kubectl logs --tail=-1 -l job-name=nvbandwidth-test-launcher
```

6. Wenn der Test abgeschlossen ist, löschen Sie ihn mit dem folgenden Befehl.

```
kubectl delete -f nvbandwidth-test-job.yaml
```

Rezepte zur Optimierung Ihres Amazon EKS-Clusters für AI/ML Workloads

> **Tip**
>
> Melden Sie sich für bevorstehende Amazon AI/ML EKS-Workshops an.

In diesem Abschnitt finden Sie kurze Rezepte für die Optimierung Ihres Amazon EKS-Clusters, insbesondere für AI/ML Workloads, die spezielle Hardware erfordern. Sie finden Anleitungen, wie Sie verhindern können, dass Pods auf bestimmten Knoten geplant werden, indem Sie den verwalteten Knotengruppen Taints hinzufügen, einschließlich Voraussetzungen, step-by-step Verfahren und Überlegungen zur Bereitstellung.

Themen

- Anleitung: Verhindern, dass Pods auf bestimmten Knoten geplant werden

Anleitung: Verhindern, dass Pods auf bestimmten Knoten geplant werden

Übersicht

Der Betrieb von Knoten mit spezialisierten Prozessoren, beispielsweise GPUs, kann teurer sein als der von Knoten auf Standard-Rechnern. Um diese Knoten vor Workloads zu schützen, die keine spezielle Hardware erfordern, können Sie Kubernetes-Taints verwenden. Taints markieren Knoten, um Pods ohne übereinstimmende Toleranzen abzulehnen und so sicherzustellen, dass nur kompatible Workloads geplant werden. Weitere Informationen finden Sie unter Taints und Toleranzen in der Kubernetes-Dokumentation.

Kubernetes-Knoten-Taints können auf neue und vorhandene verwaltete Knotengruppen angewendet werden, indem Sie dieAWS-Managementkonsoleoder über die Amazon EKS API. Diese Anleitung zeigt, wie Sie mithilfe der AWS CLI Taints auf von Amazon EKS verwaltete Knotengruppen anwenden. Hinweise zum Erstellen einer Knotengruppe mit einem Taint mithilfe der AWS-Managementkonsole finden Sie unter the section called “Create”.

Voraussetzungen

- Ein vorhandener Amazon-EKS-Cluster.

- AWS CLI installiert und mit entsprechenden Berechtigungen konfiguriert.

Schritte

Schritt 1: Knotengruppe mit Taints erstellen

Verwenden Sie den `aws eks create-nodegroup`-Befehl, um eine neue verwaltete Knotengruppe mit Taints zu erstellen. Dieses Beispiel wendet einen Taint mit Schlüssel `dedicated`, Wert `gpuGroup` und Effekt `NO_SCHEDULE` an.

```
aws eks create-nodegroup \
 --cli-input-json '
{
  "clusterName": "my-cluster",
  "nodegroupName": "node-taints-example",
  "subnets": [
     "subnet-1234567890abcdef0",
     "subnet-abcdef01234567890",
     "subnet-021345abcdef67890"
   ],
  "nodeRole": "arn:aws:iam::111122223333:role/AmazonEKSNodeRole",
  "taints": [
     {
         "key": "dedicated",
         "value": "gpuGroup",
         "effect": "NO_SCHEDULE"
     }
   ]
}'
```

Weitere Informationen und Beispiele finden Sie unter Taint in der Kubernetes-Referenzdokumentation.

Schritt 2: Taints in einer vorhandenen Knotengruppe aktualisieren

Verwenden Sie den AWS-CLI-Befehl aws eks update-nodegroup-config, um Taints für verwaltete Knotengruppen hinzuzufügen, zu entfernen oder zu ersetzen.

```
aws eks update-nodegroup-config
  --cluster-name my-cluster
  --nodegroup-name node-taints-example
```

```
  --taints 'removeTaints=[{key=dedicated,value=gpuGroup,effect=NO_SCHEDULE}]'
```

Hinweise

- Taints können aktualisiert werden, nachdem Sie die Knotengruppe mit dem `UpdateNodegroupConfig`-API erstellen.
- Der Taint-Schlüssel muss mit einem Buchstaben oder einer Zahl beginnen. Es kann Buchstaben, Zahlen, Bindestriche (`-`), Punkte (`.`) und Unterstriche (`_`) enthalten. Er kann bis zu 63 Zeichen lang sein.
- Optional kann der Taint-Schlüssel mit einem DNS-Subdomänenpräfix und einem einzelnen `/` beginnen. Wenn er mit einem DNS-Subdomänenpräfix beginnt, kann er 253 Zeichen lang sein.
- Der Wert ist optional und muss mit einem Buchstaben oder einer Zahl beginnen. Es kann Buchstaben, Zahlen, Bindestriche (`-`), Punkte (`.`) und Unterstriche (`_`) enthalten. Er kann bis zu 63 Zeichen lang sein.
- Bei direkter Verwendung von Kubernetes oder AWS-Managementkonsole muss der Taint-Effekt `NoSchedule`, `PreferNoSchedule`, oder `NoExecute` sein. Bei Verwendung der AWS-CLI oder der API muss der Taint-Effekt jedoch `NO_SCHEDULE`, `PREFER_NO_SCHEDULE` oder `NO_EXECUTE` sein.
- Für eine Knotengruppe sind maximal 50 Taints zulässig.
- Wenn Taints, die mithilfe einer verwalteten Knotengruppe erstellt wurden, manuell von einem Knoten entfernt werden, fügt Amazon EKS die Taints nicht wieder zum Knoten hinzu. Dies gilt auch dann, wenn die Taints in der Konfiguration der verwalteten Knotengruppe angegeben sind.

Ressourcen für den Einstieg in KI/ML in Amazon EKS

Für den Einstieg mit Machine Learning in EKS wählen Sie zunächst aus diesen präskriptiven Mustern aus, um schnell einen EKS-Cluster sowie ML-Software und -Hardware für die Ausführung von ML-Workloads bereitzustellen.

Workshops

Generative KI in Amazon EKS Workshop

Erfahren Sie mehr über die ersten Schritte mit Anwendungen großer Sprachmodelle (LLM) und Inferenz in Amazon EKS. Erfahren Sie, wie Sie LLM-Workloads in Produktionsqualität bereitstellen

und verwalten können. In praktischen Übungen lernen Sie, wie Sie Amazon EKS mit AWS-Services und Open-Source-Tools nutzen können, um zuverlässige LLM-Lösungen zu erstellen. Die Workshop-Umgebung stellt alle erforderlichen Infrastrukturen und Tools zur Verfügung, sodass Sie sich ganz auf das Lernen und die Umsetzung konzentrieren können.

Generative KI in Amazon EKS unter Verwendung von Neuron

Erfahren Sie mehr über die ersten Schritte mit Anwendungen großer Sprachmodelle (LLM) und Inferenz in Amazon EKS. Erfahren Sie, wie Sie produktionsreife LLM-Workloads bereitstellen und verwalten, fortschrittliche RAG-Muster mit Vektordatenbanken implementieren und datengestützte LLM-Anwendungen mit Open-Source-Frameworks erstellen. In praktischen Übungen lernen Sie, wie Sie Amazon EKS mit AWS-Services und Open-Source-Tools nutzen können, um zuverlässige LLM-Lösungen zu erstellen. Die Workshop-Umgebung stellt alle erforderlichen Infrastrukturen und Tools zur Verfügung, sodass Sie sich ganz auf das Lernen und die Umsetzung konzentrieren können.

Bewährte Methoden

Die KI/ML-bezogenen Themen im Leitfaden für bewährte Methoden für Amazon EKS bieten detaillierte Empfehlungen zu den folgenden Bereichen zur Optimierung Ihrer KI/ML-Workloads in Amazon EKS.

KI/ML-Rechenleistung und Autoscaling

Dieser Abschnitt beschreibt bewährte Methoden zur Optimierung von KI/ML-Rechenleistung und automatischer Skalierung in Amazon EKS, wobei der Schwerpunkt auf GPU-Ressourcenmanagement, Knotenausfallsicherheit und Anwendungsskalierung liegt. Er bietet Strategien wie die Planung von Workloads mit bekannten Kennzeichnungen und Knoten-Affinität, die Verwendung von ML-Kapazitätsblöcken oder On-Demand-Kapazitätsreservierungen sowie die Implementierung von Knoten-Zustandsprüfungen mit Tools wie dem EKS-Knoten-Überwachungsagent.

KI/ML-Netzwerke

In diesem Abschnitt werden bewährte Methoden zur Optimierung von KI/ML-Netzwerken in Amazon EKS beschrieben, um die Leistung und Skalierbarkeit zu verbessern. Dazu gehören Strategien wie die Auswahl von Instances mit höherer Netzwerkbandbreite oder Elastic Fabric Adapter (EFA) für verteiltes Training, die Installation von Tools wie MPI und NCCL sowie die Aktivierung der Präfix-Delegierung, um die Anzahl der IP-Adressen zu erhöhen und die Startzeiten von Pods zu verbessern.

KI/ML-Sicherheit

Dieser Abschnitt konzentriert sich auf die Sicherung der Datenspeicherung und die Gewährleistung der Compliance für KI-/ML-Workloads in Amazon EKS. Dies beinhaltet Praktiken wie die Verwendung von Amazon S3 mit AWS Key Management Service (KMS) für die serverseitige Verschlüsselung (SSE-KMS), die Konfiguration von Buckets mit regionalen KMS-Schlüsseln und S3-Bucket-Schlüsseln zur Kostensenkung, die Gewährung von IAM-Berechtigungen für KMS-Aktionen wie die Entschlüsselung für EKS-Pods und die Überprüfung mit AWS-CloudTrail-Protokollen.

KI/ML-Speicher

Dieser Abschnitt enthält bewährte Methoden zur Optimierung der Speicherung in KI-/ML-Workloads in Amazon EKS. Hierzu zählen beispielsweise die Bereitstellung von Modellen mithilfe von CSI-Treibern zum Einbinden von Services wie S3, FSx für Lustre oder EFS als persistente Volumes, die Auswahl des Speichers basierend auf den Anforderungen der Workload (z.BB. FSx für Lustre für verteiltes Training mit Optionen wie Scratch-SSD oder Persistent-SSD) und die Aktivierung von Features wie Datenkomprimierung und Striping.

KI/ML-Beobachtbarkeit

Dieser Abschnitt befasst sich mit der Überwachung und Optimierung der GPU-Auslastung für KI-/ML-Workloads in Amazon EKS, um die Effizienz zu steigern und Kosten zu senken. Dazu gehören Strategien wie die Ermittlung hoher GPU-Auslastung mit Tools wie CloudWatch Container Insights und dem in Prometheus und Grafana integrierten DCGM-Exporter von NVIDIA sowie Metriken, deren Analyse wir für Ihre KI-/ML-Workloads empfehlen.

KI/ML-Leistung

Dieser Abschnitt konzentriert sich auf die Verbesserung der Anwendungsskalierung und -leistung für KI-/ML-Workloads in Amazon EKS durch Container-Image-Verwaltung und Startup-Optimierung. Dazu gehören Verfahren wie die Verwendung kleiner, leichtgewichtiger Basis-Images oder AWS-Deep-Learning-Container mit mehrstufigen Entwicklungen, das Vorladen von Images über EBS-Snapshots oder das vorab Abrufen in den Laufzeit-Cache mithilfe von DaemonSets oder Bereitstellungen.

Referenzarchitekturen

Erkunden Sie diese GitHub-Repositorys, um Referenzarchitekturen, Beispiel-Code und Service-Programme für die Implementierung von verteiltem Training und Inferenz für KI-/ML-Workloads in Amazon EKS und anderen AWS-Services zu finden.

Verteiltes AWSome-Training

Dieses Repository bietet eine Sammlung von bewährten Methoden, Referenzarchitekturen, Beispielen für das Modelltraining und Service-Programmen für das Training großer Modelle in AWS. Es unterstützt verteiltes Training mit Amazon EKS, einschließlich CloudFormation-Vorlagen für EKS-Cluster, benutzerdefinierte AMI- und Container-Entwicklungen, Testfälle für Frameworks wie PyTorch (DDP/FSDP, MegatronLM, NeMo) und JAX sowie Tools für Validierung, Beobachtbarkeit und Leistungsüberwachung wie EFA Prometheus Exporter und Nvidia Nsight Systems.

AWSome-Inferenz

Dieses Repository bietet Referenzarchitekturen und Testfälle zur Optimierung von Inferenzlösungen in AWS, mit Schwerpunkt auf Amazon EKS und beschleunigten EC2-Instances. Es beinhaltet Infrastruktur-Einrichtungen für VPC- und EKS-Cluster, Projekte für Frameworks wie NVIDIA NIMs, TensorRT-LLM, Triton Inference Server und RayService, mit Beispielen für Modelle wie Llama3-8B und Llama 3.1 405B. Bietet Multi-Node-Bereitstellungen mit K8s LeaderWorkerSet, EKS-Autoscaling, Multi-Instance-GPUs (MIG) und Anwendungsfälle aus der Praxis, wie beispielsweise einen Audio-Bot für ASR, Inferenz und TTS.

Tutorials

Wenn Sie an der Einrichtung von Machine-Learning-Plattformen und -Frameworks in EKS interessiert sind, sehen Sie sich die in diesem Abschnitt beschriebenen Tutorials an. Diese Tutorials decken alles ab, von Mustern für die optimale Nutzung von GPU-Prozessoren über die Auswahl von Modellierungs-Tools bis hin zur Entwicklung von Frameworks für spezialisierte Branchen.

Entwicklung generativer KI-Plattformen in EKS

- Generative KI-Modelle in Amazon EKS bereitstellen
- Entwicklung von Multi-Tenant-JupyterHub-Plattformen in Amazon EKS

Spezialisierte generative KI-Frameworks in EKS ausführen

- Beschleunigung Ihrer generativen KI-Workloads für verteiltes Training mit dem NVIDIA NeMo Framework in Amazon EKS
- Ausführung von TorchServe in Amazon Elastic Kubernetes Service

NVIDIA-GPU-Leistung für ML in EKS maximieren

- Implementieren Sie GPU-Freigabe, um NVIDIA-GPUs für Ihre EKS-Cluster effizient zu nutzen:

  GPU-Freigabe in Amazon EKS mit NVIDIA-Zeitscheiben und beschleunigten EC2-Instances
- Verwenden Sie Multi-Instance-GPUs (MIGs) und NIM-Microservices, um mehr Pods pro GPU in Ihren EKS-Clustern auszuführen:

  Maximierung der GPU-Auslastung mit NVIDIA Multi-Instance GPU (MIG) in Amazon EKS: Ausführung von mehr Pods pro GPU für verbesserte Leistung
- Entwicklung und Bereitstellung eines skalierbaren Machine-Learning-Systems auf Kubernetes mit Kubeflow in AWS

Video-Codierungs-Workloads in EKS ausführen

- Bereitstellung von Videoinhalten mit fraktionierten GPUs in Containern in Amazon EKS

Beschleunigtes Laden von Images für Inferenz-Workloads

- So optimierte und sicherte H2O.ai seine KI-/ML-Infrastruktur mit Karpenter und Bottlerocket

Überwachung von ML-Workloads

- Überwachung der GPU-Auslastung in Amazon EKS mithilfe von AWS verwalteten Open-Source-Services
- Aktivierung von Pod-basierten GPU-Metriken in Amazon CloudWatch

Amazon EKS-Tools

Amazon EKS bietet eine Reihe von AWS Tools, mit denen Sie Ihre EKS-Cluster effizienter verwalten, Fehler beheben und mit ihnen interagieren können. Diese Tools nutzen KI-gestützte Funktionen und standardisierte Schnittstellen, um den Clusterbetrieb zu vereinfachen, die Fehlerbehebung zu beschleunigen und Interaktionen mit Ihren Kubernetes-Ressourcen in natürlicher Sprache zu ermöglichen. Zu den verfügbaren Tools gehören:

- Amazon EKS Model Context Protocol (MCP) -Server — Ein vollständig verwalteter Service, der KI-gestützte Erlebnisse für das EKS-Clustermanagement ermöglicht. Der EKS MCP Server ist in MCP-kompatible KI-Codierungsassistenten integriert, um in Echtzeit kontextuelles Wissen über Ihre Cluster und Kubernetes-Ressourcen bereitzustellen. Verwenden Sie ihn, um Cluster zu verwalten, Anwendungen bereitzustellen, Probleme zu beheben und die EKS-Dokumentation über Interaktionen in natürlicher Sprache abzufragen.
- Amazon Q auf der Amazon EKS-Konsole — Eine KI-gestützte Integration zur Fehlerbehebung, die direkt in der AWS Management Console verfügbar ist. Amazon Q analysiert automatisch Clusterprobleme und stellt Untersuchungsergebnisse, Ursachenanalysen und empfohlene Maßnahmen zur Schadensbegrenzung bereit, wenn Sie in der EKS-Konsole auf Fehler oder Warnungen stoßen.

Diese Tools arbeiten zusammen, um während des gesamten Lebenszyklus Ihres EKS-Clusters umfassenden Support zu bieten, von der Ersteinrichtung und Konfiguration bis hin zum laufenden Betrieb und zur Fehlerbehebung. Ganz gleich, ob Sie Anwendungen entwickeln, die Infrastruktur verwalten oder Produktionsprobleme lösen, diese AWS Tools helfen Ihnen, effizienter mit Ihren EKS-Clustern zu arbeiten.

Themen

- Amazon EKS Model Context Protocol (MCP) -Server
- Verwenden Sie Amazon Q Developer auf der Amazon EKS-Konsole

Amazon EKS Model Context Protocol (MCP) -Server

Der Amazon EKS MCP-Server ist ein vollständig verwalteter Service, der KI-gestützte Erlebnisse für Entwicklung und Betrieb ermöglicht. Das Model Context Protocol (MCP) bietet eine standardisierte Schnittstelle, die KI-Agenten und -Anwendungen mit kontextuellem Echtzeitwissen über Ihre EKS-

Cluster und Kubernetes-Ressourcen erweitert und so genauere, kontextsensitive Antworten und KI-gestützte Workflows während des gesamten Anwendungslebenszyklus ermöglicht, von der Ersteinrichtung über die Produktionsoptimierung bis hin zur Fehlerbehebung.

> **Note**
>
> Der Amazon EKS MCP Server befindet sich in der Vorschauversion für Amazon EKS und kann sich ändern.

-Übersicht

Der EKS MCP-Server kann problemlos in jeden MCP-kompatiblen KI-Codierungsassistenten integriert werden, um Ihren Entwicklungsworkflow wie Kiro, Amazon Q Developer CLI oder Tools von Drittanbietern wie Cursor oder Cline zu verbessern. Zu Beginn führt Sie der EKS MCP-Server durch die Clustererstellung, die automatische Bereitstellung der Voraussetzungen und die Anwendung von Best Practices. AWS Während der Entwicklung vereinfacht er den Betrieb von EKS und Kubernetes, indem er hochwertige Workflows für die Anwendungsbereitstellung und das Clustermanagement bereitstellt. Für Debugging und Problembehebung beschleunigt der Server die Problemlösung durch integrierte Fehlerbehebungshilfen und den Zugriff auf die Wissensdatenbank, die über die EKS-Konsole und Ihre bevorzugten KI-Assistenten verfügbar sind. Auf diese Funktionen kann über Interaktionen in natürlicher Sprache zugegriffen werden, sodass Sie komplexe Kubernetes-Operationen intuitiver und effizienter ausführen können.

Der vollständig verwaltete EKS MCP-Server wird in der AWS Cloud gehostet, sodass keine lokale Installation und Wartung erforderlich ist. Er bietet Funktionen der Enterprise-Klasse wie automatische Updates und Patches, zentrale Sicherheit durch AWS IAM-Integration, umfassende Audit-Protokollierung und die bewährte Skalierbarkeit AWS CloudTrail, Zuverlässigkeit und Unterstützung von. AWS Der vollständig verwaltete EKS MCP-Server, der in der AWS Cloud gehostet wird, bietet die folgenden Hauptvorteile:

- Eliminieren Sie Installation und Wartung. Da der EKS MCP-Server in der AWS Cloud gehostet wird, müssen Sie keine Versionsupdates mehr verwalten oder lokale Serverprobleme beheben. Konfigurieren Sie einfach Ihren KI-Assistenten so, dass er eine Verbindung zum gehosteten EKS-MCP-Serverendpunkt herstellt, und schon können Sie mit der Arbeit mit Ihren EKS-Clustern beginnen.
- Zentralisiertes Zugriffsmanagement. Der EKS MCP-Server ist in IAM integriert und bietet so eine zentrale, sichere Möglichkeit, den Zugriff auf den Server zu kontrollieren. Alle Anfragen werden

mit AWS SigV4 über einen schlanken Proxy signiert, was eine nahtlose Integration mit Ihren vorhandenen AWS Anmeldeinformationen und IAM-Richtlinien ermöglicht.

- Verbesserte Überwachung und Transparenz. AWS CloudTrail Die Integration erfasst die Aufrufe von Tools für Initialisierung und Vollzugriff, die über den gehosteten Service getätigt werden, und ermöglicht so detaillierte Prüfprotokolle und Compliance-Berichte.
- Immer up-to-date. Erhalten Sie automatisch neue Tools, Funktionen und Bugfixes, ohne dass lokale Installationen aktualisiert werden müssen. Der gehostete Service wird auf der Grundlage Ihres Feedbacks und Ihrer AWS Best Practices kontinuierlich verbessert.

Beispiele für die Integration

Der EKS MCP Server bietet mehrere Tools, mit denen Sie:

- Ihren Cluster verwalten Erstellen, konfigurieren und verwalten Sie EKS-Cluster mit automatisierten Best Practices.
- Kubernetes-Ressourcen verwalten Stellen Sie Anwendungen bereit, verwalten Sie Kuberentes-Ressourcen und überprüfen Sie den Clusterstatus.
- Beheben Sie Ihren Cluster. Diagnostizieren Sie Probleme mithilfe integrierter Tools zur Problembehebung und der Knowledge-Base von Runbooks
- Dokumentation abfragen Suchen Sie die relevante EKS-Dokumentation kontextbezogen und rufen Sie sie ab.

Erkunden Sie Ihre Cluster

```
Show me all EKS clusters and their status
What insights does EKS have about my production-cluster?
Show me the VPC configuration for my staging cluster
```

Schauen Sie sich die Kubernetes-Ressourcen an

```
List all deployments in the production namespace
Show me pods that are not in Running state
Get the logs from the api-server pod in the last 30 minutes
```

Beheben von Problemen mit

```
Why is my nginx-ingress-controller pod failing to start?
Search the EKS troubleshooting guide for pod networking issues
Show me events related to the failed deployment in the staging namespace
```

Ressourcen erstellen (wenn der Schreibmodus aktiviert ist)

```
Create a new EKS cluster named demo-cluster with VPC and Auto Mode
Deploy my containerized app from ECR to the production namespace with 3 replicas
Generate a Kubernetes deployment YAML for my Node.js app running on port 3000
```

Erste Schritte

Um zu beginnen, sehen Sie sich the section called “Erste Schritte” an.

Erste Schritte mit dem Amazon EKS MCP Server

Dieses Handbuch führt Sie durch die Schritte zur Einrichtung und Verwendung des EKS MCP-Servers mit Ihren KI-Codeassistenten. Sie erfahren, wie Sie Ihre Umgebung konfigurieren, eine Verbindung zum Server herstellen und mit der Verwaltung Ihrer EKS-Cluster durch Interaktionen in natürlicher Sprache beginnen.

> **Note**
>
> Der Amazon EKS MCP Server befindet sich in der Vorschauversion für Amazon EKS und kann sich ändern.

Voraussetzungen

Stellen Sie zunächst sicher, dass Sie die folgenden Aufgaben durchgeführt haben:

- Erstellen Sie ein AWS Konto mit Zugriff auf Amazon EKS
- Installieren und konfigurieren Sie die AWS CLI mit Anmeldeinformationen
- Installieren Sie Python 3.10+
- Installieren uv

Einrichtung

1. Überprüfen Sie die -Voraussetzungen.

```
# Check that your Python version is 3.10 or higher
python3 --version

# Check uv installation
uv --version

# Verify CLI configuration
aws configure list
```

2. IAM-Berechtigungen einrichten

Um eine Verbindung zum EKS MCP-Server herzustellen, müssen Ihrer IAM-Rolle die folgenden Richtlinien zugeordnet sein: **`eks-mcp:InvokeMcp`**(erforderliche Berechtigungen für die Initialisierung und das Abrufen von Informationen über verfügbare Tools), **`eks-mcp:CallReadOnlyTool`**(erforderliche Berechtigungen für die Verwendung schreibgeschützter Tools) und **`eks-mcp:CallPrivilegedTool`**(erforderliche Berechtigungen für die Verwendung von Tools mit Vollzugriff (Schreibzugriff)). Diese `eks-mcp` Berechtigungen sind in den unten aufgeführten verwalteten Richtlinien für Schreibschutz und vollen AWS Zugriff enthalten.

- Öffnen Sie die IAM-Konsole.
- Wählen Sie im linken Navigationsbereich je nach Identität, der Sie die Richtlinie zuordnen möchten, Benutzer, Benutzergruppen oder Rollen und anschließend den Namen des jeweiligen Benutzers, der Gruppe oder Rolle aus.
- Wählen Sie die Registerkarte Berechtigungen.
- Wählen Sie Richtlinien anhängen (oder Berechtigungen hinzufügen, wenn es das erste Mal ist).
- Suchen Sie in der Richtlinienliste nach der verwalteten Richtlinie, die Sie anhängen möchten, und wählen Sie sie aus:
- Nur-Lese-Operationen: Amazon EKSMCPRead OnlyAccess
- Wählen Sie Richtlinien anhängen (oder Weiter und dann zur Bestätigung Berechtigungen hinzufügen).

Dadurch wird die Richtlinie angehängt, und die Berechtigungen werden sofort wirksam. Sie können derselben Identität mehrere Richtlinien zuordnen, und jede kann verschiedene Berechtigungen

enthalten. Weitere Informationen zu diesen Richtlinien finden Sie unter AWS Verwaltete Richtlinien für Amazon Elastic Kubernetes Service.

3. Wählen Sie einen KI-Assistenten

Wählen Sie einen der folgenden MCP-kompatiblen KI-Assistenten oder ein anderes MCP-kompatibles Tool:

- Installieren Sie Amazon Q Developer CLI
- Installieren Sie Kiro
- Installieren Sie Cursor
- Installieren Sie die Cline VS Code-Erweiterung

Schritt 1: Konfigurieren Sie Ihren KI-Assistenten

Wählen Sie eine der folgenden Optionen, um Ihren KI-Code-Assistenten einzurichten. Wenn Sie diesen Schritt abschließen, wird Ihr KI-Code-Assistent für die Verwendung des MCP-Proxys eingerichtet AWS, der für einen sicheren, authentifizierten Zugriff auf den Amazon EKS MCP-Server erforderlich ist. Dazu gehört das Hinzufügen oder Bearbeiten der MCP-Konfigurationsdatei (z. B. `~/.aws/amazonq/mcp.json` für Amazon Q Developer CLI). Der Proxy fungiert als clientseitige Bridge, übernimmt die AWS SigV4-Authentifizierung mithilfe Ihrer lokalen AWS Anmeldeinformationen und ermöglicht die dynamische Tool-Erkennung für die Interaktion mit AWS Backend-MCP-Servern wie dem EKS MCP-Server. Weitere Informationen finden Sie unter MCP Proxy for Repository. AWS

Option A: Amazon Q Developer CLI

Die Q Developer CLI bietet die am besten integrierte Erfahrung mit dem EKS MCP Server.

1. Suchen Sie die MCP-Konfigurationsdatei

- MacOS/Linux: `~/.aws/q/mcp.json`
- Windows: `%USERPROFILE%\.aws\q\mcp.json`

2. MCP-Serverkonfiguration hinzufügen

Erstellen Sie die Konfigurationsdatei, falls sie nicht existiert. Achten Sie darauf, den Platzhalter region (`{region}`) durch die gewünschte Region zu ersetzen.

Für Mac/Linux:

```
{
  "mcpServers": {
    "eks-mcp": {
      "disabled": false,
      "type": "stdio",
      "command": "uvx",
      "args": [
        "mcp-proxy-for-aws@latest",
        "https://eks-mcp.{region}.api.aws/mcp",
        "--service",
        "eks-mcp",
        "--profile",
        "default",
        "--region",
        "{region}"
      ]
    }
  }
}
```

Für Windows:

```
{
  "mcpServers": {
    "eks-mcp": {
      "disabled": false,
      "type": "stdio",
      "command": "uvx",
      "args": [
        "--from",
        "mcp-proxy-for-aws@latest",
        "mcp-proxy-for-aws.exe",
        "https://eks-mcp.{region}.api.aws/mcp",
        "--service",
        "eks-mcp",
        "--profile",
        "default",
        "--region",
        "{region}"
      ]
    }
```

```
  }
}
```

Sicherheitshinweis: `--read-only` kann verwendet werden, um nur Leseoperationen für Tools zuzulassen.

3. Konfiguration überprüfen

Starten Sie Q Developer CLI neu und überprüfen Sie dann die verfügbaren Tools:

```
q /tools
```

Option B: Kiro IDE

Kiro ist ein Codierungs-Workspace mit integrierter MCP-Unterstützung, bei dem KI an erster Stelle steht.

1. Öffnen Sie die Kiro Settings

- Öffne Kiro
- Gehe zu Kiro → Einstellungen und suche nach „MCP Config“
- Oder drücken Sie `Cmd+Shift+P,` (Mac) oder `Ctrl+Shift+P,` (Windows/Linux) und suchen Sie nach „MCP Config“

2. Fügen Sie die MCP-Serverkonfiguration hinzu

- Klicken Sie auf „Workspace MCP Config öffnen“ oder „User MCP Config öffnen“, um die MCP-Konfigurationsdatei direkt zu bearbeiten.

Achten Sie darauf, den Platzhalter Region (`{region}`) durch die gewünschte Region zu ersetzen.

Für Mac/Linux:

```
{
  "mcpServers": {
    "eks-mcp": {
      "disabled": false,
      "type": "stdio",
      "command": "uvx",
      "args": [
```

```
        "mcp-proxy-for-aws@latest",
        "https://eks-mcp.{region}.api.aws/mcp",
        "--service",
        "eks-mcp",
        "--profile",
        "default",
        "--region",
        "{region}"
      ]
    }
  }
}
```

Für Windows:

```
{
  "mcpServers": {
    "eks-mcp": {
      "disabled": false,
      "type": "stdio",
      "command": "uvx",
      "args": [
        "--from",
        "mcp-proxy-for-aws@latest",
        "mcp-proxy-for-aws.exe",
        "https://eks-mcp.{region}.api.aws/mcp",
        "--service",
        "eks-mcp",
        "--profile",
        "default",
        "--region",
        "{region}"
      ]
    }
  }
}
```

Sicherheitshinweis: `--read-only` kann verwendet werden, um nur Leseoperationen für Tools zuzulassen.

Option C: Cursor-IDE

Cursor bietet integrierte MCP-Unterstützung mit einer grafischen Konfigurationsoberfläche.

1. Öffnen Sie die Cursoreinstellungen

- Cursor öffnen
- Gehen Sie zu Einstellungen → Cursoreinstellungen → Tools & MCP
- Oder drücken Sie `Cmd+Shift+P` (Mac)/`Ctrl+Shift+P`(Windows) und suchen Sie nach „MCP“

2. MCP-Serverkonfiguration hinzufügen

- Klicken Sie auf „Neuer MCP-Server“

Erstellen Sie die Konfigurationsdatei, falls sie nicht existiert. Achten Sie darauf, den Platzhalter region (`{region}`) durch die gewünschte Region zu ersetzen.

Für Mac/Linux:

```
{
  "mcpServers": {
    "eks-mcp": {
      "disabled": false,
      "type": "stdio",
      "command": "uvx",
      "args": [
        "mcp-proxy-for-aws@latest",
        "https://eks-mcp.{region}.api.aws/mcp",
        "--service",
        "eks-mcp",
        "--profile",
        "default",
        "--region",
        "{region}"
      ]
    }
  }
}
```

Für Windows:

```
{
  "mcpServers": {
    "eks-mcp": {
```

```
      "disabled": false,
      "type": "stdio",
      "command": "uvx",
      "args": [
        "--from",
        "mcp-proxy-for-aws@latest",
        "mcp-proxy-for-aws.exe",
        "https://eks-mcp.{region}.api.aws/mcp",
        "--service",
        "eks-mcp",
        "--profile",
        "default",
        "--region",
        "{region}"
      ]
    }
  }
}
```

Sicherheitshinweis: `--read-only` kann verwendet werden, um nur Leseoperationen für Tools zuzulassen.

3. Cursor neu starten

Schließen Sie den Cursor und öffnen Sie ihn erneut, damit die Änderungen wirksam werden.

4. Überprüfen Sie im Cursor-Chat

Öffne das Chat-Panel und versuche:

```
What EKS MCP tools are available?
```

Sie sollten eine Liste der verfügbaren EKS-Verwaltungstools sehen.

Option D: Cline (VS-Code-Erweiterung)

Cline ist eine beliebte VS Code-Erweiterung, die KI-Unterstützung direkt in Ihren Editor bringt.

1. Öffnen Sie die Cline-Einstellungen

- Öffnen Sie Cline
- Drücken Sie `Cmd+Shift+P` (Mac)/`Ctrl+Shift+P`(Windows) und suchen Sie nach „MCP“

2. MCP-Serverkonfiguration hinzufügen

- Klicken Sie auf „Server hinzufügen“
- Klicken Sie auf „Benutzerkonfiguration öffnen“

Erstellen Sie die Konfigurationsdatei, falls sie nicht existiert. Achten Sie darauf, den Platzhalter region (`{region}`) durch die gewünschte Region zu ersetzen.

Für Mac/Linux:

```
{
  "mcpServers": {
    "eks-mcp": {
      "disabled": false,
      "type": "stdio",
      "command": "uvx",
      "args": [
        "mcp-proxy-for-aws@latest",
        "https://eks-mcp.{region}.api.aws/mcp",
        "--service",
        "eks-mcp",
        "--profile",
        "default",
        "--region",
        "{region}"
      ]
    }
  }
}
```

Für Windows:

```
{
  "mcpServers": {
    "eks-mcp": {
      "disabled": false,
      "type": "stdio",
      "command": "uvx",
      "args": [
        "--from",
        "mcp-proxy-for-aws@latest",
```

```
        "mcp-proxy-for-aws.exe",
        "https://eks-mcp.{region}.api.aws/mcp",
        "--service",
        "eks-mcp",
        "--profile",
        "default",
        "--region",
        "{region}"
      ]
    }
  }
}
```

Sicherheitshinweis: `--read-only` kann verwendet werden, um nur Leseoperationen für Tools zuzulassen.

2. Laden Sie den VS-Code neu

Drücken Sie`Cmd+Shift+P`/`Ctrl+Shift+P`und wählen Sie „Entwickler: Fenster neu laden“

3. Überprüfen Sie die Konfiguration

Öffne Cline und frage:

```
List the available MCP tools for EKS
```

Schritt 2: (Optional) Erstellen Sie eine „Write“ -Richtlinie

Optional können Sie eine vom Kunden verwaltete IAM-Richtlinie erstellen, die vollen Zugriff auf den Amazon EKS MCP-Server bietet. Diese Richtlinie gewährt Berechtigungen zur Verwendung aller Tools auf dem EKS MCP-Server, einschließlich sowohl privilegierter Tools, die Schreibvorgänge beinhalten können, als auch schreibgeschützte Tools. Beachten Sie, dass Berechtigungen mit hohem Risiko (alles mit Delete* oder uneingeschränkte IAM-Ressourcen) in dieser Richtlinie enthalten sind, da sie für setup/teardown die Clusterressourcen im Tool manage_eks_stacks erforderlich sind.

```
aws iam create-policy \
 --policy-name EKSMcpWriteManagementPolicy \
 --policy-document "{\"Version\": \"2012-10-17\", \"Statement\": [{\"Effect
\": \"Allow\", \"Action\": [\"eks:DescribeCluster\", \"eks:ListClusters\",
 \"eks:DescribeNodegroup\", \"eks:ListNodegroups\", \"eks:DescribeAddon\",
 \"eks:ListAddons\", \"eks:DescribeAccessEntry\", \"eks:ListAccessEntries\",
```

```
 \"eks:DescribeInsight\", \"eks:ListInsights\", \"eks:AccessKubernetesApi\"],
 \"Resource\": \"*\"}, {\"Effect\": \"Allow\", \"Action\": [\"eks:CreateCluster
\", \"eks:DeleteCluster\", \"eks:CreateAccessEntry\", \"eks:TagResource\"],
 \"Resource\": \"*\"}, {\"Effect\": \"Allow\", \"Action\": [\"iam:GetRole\",
 \"iam:ListRolePolicies\", \"iam:ListAttachedRolePolicies\", \"iam:GetRolePolicy\",
 \"iam:GetPolicy\", \"iam:GetPolicyVersion\"], \"Resource\": \"*\"}, {\"Effect\":
 \"Allow\", \"Action\": [\"iam:TagRole\", \"iam:CreateRole\", \"iam:AttachRolePolicy
\", \"iam:PutRolePolicy\", \"iam:DetachRolePolicy\", \"iam:DeleteRole\"],
 \"Resource\": \"*\"}, {\"Effect\": \"Allow\", \"Action\": [\"iam:PassRole\"],
 \"Resource\": \"*\", \"Condition\": {\"StringEquals\": {\"iam:PassedToService
\": [\"eks.amazonaws.com\", \"ec2.amazonaws.com\"]}}}, {\"Effect\": \"Allow\",
 \"Action\": [\"ec2:CreateVpc\", \"ec2:CreateSubnet\", \"ec2:CreateRouteTable\",
 \"ec2:CreateRoute\", \"ec2:CreateInternetGateway\", \"ec2:CreateNatGateway\",
 \"ec2:CreateSecurityGroup\", \"ec2:AttachInternetGateway\", \"ec2:AssociateRouteTable
\", \"ec2:ModifyVpcAttribute\", \"ec2:ModifySubnetAttribute\", \"ec2:AllocateAddress
\", \"ec2:CreateTags\"], \"Resource\": \"*\"}, {\"Effect\": \"Allow\", \"Action
\": [\"ec2:DeleteVpc\", \"ec2:DeleteSubnet\", \"ec2:DisassociateRouteTable\",
 \"ec2:DeleteRouteTable\", \"ec2:DeleteRoute\", \"ec2:DetachInternetGateway\",
 \"ec2:DeleteInternetGateway\", \"ec2:DeleteNatGateway\", \"ec2:ReleaseAddress\",
 \"ec2:DeleteSecurityGroup\"], \"Resource\": \"*\"}, {\"Effect\": \"Allow\", \"Action
\": [\"ec2:DescribeVpcs\", \"ec2:DescribeSubnets\", \"ec2:DescribeRouteTables\",
 \"ec2:DescribeInternetGateways\", \"ec2:DescribeNatGateways\", \"ec2:DescribeAddresses
\", \"ec2:DescribeSecurityGroups\", \"ec2:DescribeAvailabilityZones\"], \"Resource
\": \"*\"}, {\"Effect\": \"Allow\", \"Action\": [\"cloudformation:CreateStack
\", \"cloudformation:UpdateStack\", \"cloudformation:DeleteStack\",
 \"cloudformation:DescribeStacks\", \"cloudformation:TagResource\"], \"Resource
\": \"*\"}, {\"Effect\": \"Allow\", \"Action\": [\"sts:GetCallerIdentity\"],
 \"Resource\": \"*\"}, {\"Effect\": \"Allow\", \"Action\": [\"logs:StartQuery\",
 \"logs:GetQueryResults\"], \"Resource\": \"*\"}, {\"Effect\": \"Allow\", \"Action\":
 [\"cloudwatch:GetMetricData\"], \"Resource\": \"*\"}, {\"Effect\": \"Allow\", \"Action
\": [\"eks-mcp:*\"], \"Resource\": \"*\"}]}"
```

Schritt 3: Überprüfen Sie Ihr Setup

Verbindung testen

Stellen Sie Ihrem KI-Assistenten eine einfache Frage, um die Verbindung zu überprüfen:

```
List all EKS clusters in my {aws} account
```

Sie sollten eine Liste Ihrer EKS-Cluster sehen.

Schritt 4: Führen Sie Ihre ersten Aufgaben aus

Beispiel 1: Erkunden Sie Ihre Cluster

```
Show me all EKS clusters and their status
What insights does EKS have about my production-cluster?
Show me the VPC configuration for my staging cluster
```

Beispiel 2: Überprüfen Sie die Kubernetes-Ressourcen

```
Get the details of all the kubernetes resources deployed in my EKS cluster
Show me pods that are not in Running state or pods with any restarts
Get the logs from the aws-node daemonset in the last 30 minutes
```

Beispiel 3: Probleme beheben

```
Why is my nginx-ingress-controller pod failing to start?
Search the EKS troubleshooting guide for pod networking issues
Show me events related to the failed deployment in the staging namespace
```

Beispiel 4: Ressourcen erstellen (wenn der Schreibmodus aktiviert ist)

```
Create a new EKS cluster named demo-cluster with VPC and Auto Mode
Deploy my containerized app from ECR to the production namespace with 3 replicas
Generate a Kubernetes deployment YAML for my Node.js app running on port 3000
```

Allgemeine Konfigurationen

Szenario 1: Mehrere AWS Profile

Wenn Sie mit mehreren AWS Konten arbeiten, erstellen Sie separate MCP-Serverkonfigurationen.

Für Mac/Linux:

```
{
  "mcpServers": {
    "eks-mcp-prod": {
      "disabled": false,
      "type": "stdio",
      "command": "uvx",
```

```
      "args": [
        "mcp-proxy-for-aws@latest",
        "https://eks-mcp.{region}.api.aws/mcp",
        "--service",
        "eks-mcp",
        "--profile",
        "production",
        "--region",
        "us-west-2"
      ]
    },
    "eks-mcp-dev": {
      "disabled": false,
      "type": "stdio",
      "command": "uvx",
      "args": [
        "mcp-proxy-for-aws@latest",
        "https://eks-mcp.{region}.api.aws/mcp",
        "--service",
        "eks-mcp",
        "--profile",
        "development",
        "--region",
        "us-east-1"
      ]
    }
  }
}
```

Für Windows:

```
{
  "mcpServers": {
    "eks-mcp-prod": {
      "disabled": false,
      "type": "stdio",
      "command": "uvx",
      "args": [
        "--from",
        "mcp-proxy-for-aws@latest",
        "mcp-proxy-for-aws.exe",
        "https://eks-mcp.{region}.api.aws/mcp",
        "--service",
```

```
        "eks-mcp",
        "--profile",
        "production",
        "--region",
        "us-west-2"
      ]
    },
    "eks-mcp-dev": {
      "disabled": false,
      "type": "stdio",
      "command": "uvx",
      "args": [
        "--from",
        "mcp-proxy-for-aws@latest",
        "mcp-proxy-for-aws.exe",
        "https://eks-mcp.{region}.api.aws/mcp",
        "--service",
        "eks-mcp",
        "--profile",
        "development",
        "--region",
        "us-east-1"
      ]
    }
  }
}
```

Szenario 2: Schreibgeschützt für die Produktion

Erstellen Sie eine schreibgeschützte Konfiguration für Produktionsumgebungen.

Für Mac/Linux:

```
{
  "mcpServers": {
    "eks-mcp-prod-readonly": {
      "command": "uvx",
      "args": [
        "mcp-proxy-for-aws@latest",
        "https://eks-mcp.{region}.api.aws/mcp",
        "--service",
        "eks-mcp",
        "--profile",
```

```
        "production",
        "--region",
        "us-west-2",
        "--read-only"
      ],
      "autoApprove": [
        "list_k8s_resources",
        "get_pod_logs",
        "get_k8s_events"
      ]
    }
  }
}
```

Für Windows:

```
{
  "mcpServers": {
    "eks-mcp-prod-readonly": {
      "command": "uvx",
      "args": [
        "--from",
        "mcp-proxy-for-aws@latest",
        "mcp-proxy-for-aws.exe",
        "https://eks-mcp.{region}.api.aws/mcp",
        "--service",
        "eks-mcp",
        "--profile",
        "production",
        "--region",
        "us-west-2",
        "--read-only"
      ],
      "autoApprove": [
        "list_k8s_resources",
        "get_pod_logs",
        "get_k8s_events"
      ]
    }
  }
}
```

Szenario 3: Entwicklung mit Vollzugriff

Für Entwicklungsumgebungen mit vollem Schreibzugriff.

Für Mac/Linux:

```
{
  "mcpServers": {
    "eks-mcp-dev-full": {
      "command": "uvx",
      "args": [
        "mcp-proxy-for-aws@latest",
        "https://eks-mcp.{region}.api.aws/mcp",
        "--service",
        "eks-mcp",
        "--profile",
        "development",
        "--region",
        "us-east-1"
      ]
    }
  }
}
```

Für Windows:

```
{
  "mcpServers": {
    "eks-mcp-dev-full": {
      "command": "uvx",
      "args": [
        "--from",
        "mcp-proxy-for-aws@latest",
        "mcp-proxy-for-aws.exe",
        "https://eks-mcp.{region}.api.aws/mcp",
        "--service",
        "eks-mcp",
        "--profile",
        "development",
        "--region",
        "us-east-1"
      ]
    }
```

```
  }
}
```

Überlegungen

Sicherheit

Geben Sie keine Geheimnisse oder vertraulichen Informationen über zulässige Eingabemechanismen weiter:

- Schließen Sie keine Geheimnisse oder Anmeldeinformationen in YAML-Dateien ein, die mit apply_yaml angewendet wurden.
- Übergeben Sie vertrauliche Informationen nicht direkt in der Aufforderung an das Modell.
- Nehmen Sie keine Geheimnisse in CloudFormation Vorlagen oder Anwendungsmanifeste auf.
- Vermeiden Sie die Verwendung von MCP-Tools zur Erstellung von Kubernetes-Geheimnissen, da dies die Bereitstellung der geheimen Daten für das Modell erfordern würde.
- Vermeiden Sie es, vertrauliche Informationen in Anwendungsprotokollen innerhalb von Kubernetes-Pods zu protokollieren.

Sicherheit von YAML-Inhalten:

- Verwenden Sie nur YAML-Dateien aus vertrauenswürdigen Quellen.
- Der Server stützt sich auf die Kubernetes-API-Validierung für YAML-Inhalte und führt keine eigene Validierung durch.
- Prüfen Sie YAML-Dateien, bevor Sie sie auf Ihren Cluster anwenden.

Anstatt Geheimnisse über MCP weiterzugeben:

- Verwenden Sie AWS Secrets Manager oder Parameter Store, um vertrauliche Informationen zu speichern.
- Konfigurieren Sie das richtige Kubernetes-RBAC für Dienstkonten.
- Verwenden Sie IAM-Rollen für Dienstkonten (IRSA) für den Dienstzugriff von Pods aus. AWS

Redaktion sensibler Daten:

- Der EKS MCP Server redigiert automatisch allgemeine Muster für Sicherheitstoken, Zertifikate und andere vertrauliche Informationen in Tool-Antworten.
- Geschwärzte Werte werden durch ersetzt, `HIDDEN_FOR_SECURITY_REASONS` um zu verhindern, dass Daten versehentlich dem Modell zugänglich gemacht werden.
- Diese Schwärzung gilt für alle Antworten des Tools, einschließlich Protokollen, Ressourcenbeschreibungen und Konfigurationsdaten.

Als Nächstes

Informationen zu den Konfigurationsoptionen finden Sie unterthe section called “Konfiguration”. Eine vollständige Liste der Tools finden Sie unterthe section called “Tools”.

Referenz zur Amazon EKS MCP Serverkonfiguration

Dieses Handbuch zeigt alle verfügbaren Konfigurationen für das mcp-proxy-for-awsclientseitige Tool, mit dem Sie von Ihrer IDE aus eine Verbindung zum vollständig verwalteten Amazon EKS MCP-Server herstellen können.

> **Note**
>
> Der Amazon EKS MCP Server befindet sich in der Vorschauversion für Amazon EKS und kann sich ändern.

Beispiel

```
{
  "mcpServers": {
    "eks-mcp": {
      "disabled": false,
      "type": "stdio",
      "command": "uvx",
      "args": [
        "mcp-proxy-for-aws@latest",
        "https://eks-mcp.us-west-2.api.aws/mcp",
        "--service",
        "eks-mcp",
        "--profile",
        "default",
```

```
        "--region",
        "us-east-1",
        "--read-only"
      ]
    }
  }
}
```

IAM-Berechtigungen

Die Rolle, die für die Verbindung mit dem MCP-Server verwendet wird, erfordert **`eks-mcp:InvokeMcp`**Berechtigungen für die Initialisierung und das Abrufen von Informationen über verfügbare Tools. **`eks-mcp:CallReadOnlyTool`**ist für die Verwendung von schreibgeschützten Tools und **`eks-mcp:CallPrivilegedTool`**für die Verwendung von Tools mit vollem Zugriff (Schreibzugriff) erforderlich.

Umgebungsvariablen

`AWS_PROFILE`(optional) Name des zu verwendenden AWS Anmeldeinformationsprofils; kann durch das `--profile` Befehlszeilenargument außer Kraft gesetzt werden.

- Beispiel: `export AWS_PROFILE=production`

`AWS_REGION`(optional) AWS Region für die Sigv4-Signierung; standardmäßig auf, falls nicht festgelegt. `us-west-2`

- Beispiel: `export AWS_REGION=us-east-1`

Argumente

SigV4-MCP-Endpunkt-URL (erforderlich) Die MCP-Endpunkt-URL, zu der eine Verbindung hergestellt werden soll.

- Beispiel: `https://eks-mcp.us-west-2.api.aws/mcp`

`--service`(optional) AWS Dienstname für die SigV4-Signatur; wird automatisch anhand des Endpunkt-Hostnamens erkannt, falls nicht angegeben.

- Beispiel: `--service eks-mcp`

`--profile`(optional) Zu verwendendes AWS Anmeldeinformationsprofil. Standardmäßig wird die `AWS_PROFILE` Umgebungsvariable verwendet, wenn sie nicht angegeben ist.

- Beispiel: `--profile production`

`--region` AWS zu verwendende Region. Verwendet die `AWS_REGION` Umgebungsvariable, wenn sie nicht gesetzt ist, standardmäßig. `us-east-1`

- Beispiel: `--region us-west-2`

`--read-only`(optional) Deaktivieren Sie Tools, für die möglicherweise Schreibberechtigungen erforderlich sind (Tools, die KEINE Schreibberechtigungen benötigen, sind mit readOnlyHint =true versehen). Standardmäßig sind alle Tools aktiviert.

- Beispiel: `--read-only`

Weitere Konfigurationsoptionen finden Sie unter Konfigurationsparameter.

Tools

Ausführliche Informationen zu allen verfügbaren Tools, einschließlich Parametern und Anwendungsbeispielen, finden Sie unterthe section called “Tools”.

Referenz zu den Amazon EKS MCP Servertools

Der Server stellt die folgenden MCP-Tools zur Verfügung.

> **Note**
>
> Der Amazon EKS MCP Server befindet sich in der Vorschauversion für Amazon EKS und kann sich ändern.

Tools, die nur lesbar sind

In diesem Abschnitt werden die schreibgeschützten Tools beschrieben, die für den EKS MCP-Server verfügbar sind. Beachten Sie, dass alle schreibgeschützten Kubernetes-API-Operationen auf beide zugreifen können:

- Private Cluster (siehe Privater Cluster-Endpunkt)
- Öffentliche Cluster

search_eks_documentation

Suchen Sie in der EKS-Dokumentation nach Informationen und Anleitungen up-to-date. Dieses Tool bietet Zugriff auf die neueste EKS-Dokumentation, einschließlich neuer Funktionen und aktueller Verbesserungen, von denen Agenten möglicherweise nichts wissen.

Parameter:

- Anfrage (erforderlich): Ihre spezifische Frage oder Suchanfrage im Zusammenhang mit der EKS-Dokumentation, den Funktionen oder bewährten Methoden.
- limit (optional): Maximale Anzahl zurückzugebender Dokumentationsergebnisse (1—10). Standard: 5.

search_eks_troubleshooting_guide

Durchsucht den EKS-Leitfaden zur Problembehandlung auf der Grundlage einer Abfrage nach Informationen zur Problembehandlung. Es hilft bei der Identifizierung häufiger Probleme und bietet step-by-step Lösungen.

Parameter:

- Anfrage (erforderlich): Ihre spezifische Frage oder Problembeschreibung im Zusammenhang mit der EKS-Fehlerbehebung.

describe_eks_resource

Ruft detaillierte Informationen zu einer bestimmten EKS-Clusterressource ab, einschließlich Konfiguration, Status und Metadaten.

Parameter:

- cluster_name (erforderlich): Name des EKS-Clusters (erforderlich für Ressourcen im Clusterbereich).
- resource_type (erforderlich): Der zu beschreibende EKS-Ressourcentyp. Zulässige Werte:
- accessentry (erfordert cluster_name und resource_name als PrincipalArn)

- Addon (benötigt cluster_name und resource_name als Addon-Namen)
- cluster (erfordert cluster_name), nodegroup (erfordert cluster_name und resource_name als Knotengruppennamen).
- resource_name (optional): Name der spezifischen Ressource, die beschrieben werden soll (für die meisten Ressourcentypen erforderlich).

list_eks_resources

Listet EKS-Ressourcen eines bestimmten Typs auf und gibt eine Zusammenfassung aller Ressourcen des angegebenen Typs zurück, auf die zugegriffen werden kann.

Parameter:

- resource_type (erforderlich): Der aufzulistende EKS-Ressourcentyp. Zulässige Werte:
- accessentry (erfordert cluster_name)
- Addon (benötigt cluster_name)
- Cluster (keine zusätzlichen Parameter erforderlich)
- nodegroup (erfordert cluster_name).
- cluster_name (optional): Name des EKS-Clusters (erforderlich für Ressourcen im Clusterbereich).

get_eks_insights

Ruft Einblicke in den EKS-Cluster und Empfehlungen zur Optimierung ab. Bietet umsetzbare Erkenntnisse zur Sicherheits-, Leistungs- und Kostenoptimierung auf der Grundlage von AWS Best Practices und Clusteranalysen.

Parameter:

- cluster_name (erforderlich): Name des EKS-Clusters.
- Kategorie (optional): Optionale Kategorie zum Filtern von Erkenntnissen (z. B. „MISCONFIGURATION“ oder „UPGRADE_READINESS“).
- insight_id (optional): Optionale ID einer bestimmten Erkenntnis, für die detaillierte Informationen abgerufen werden sollen.
- next_token (optional): Optionales Token für die Paginierung, um die nächsten Ergebnisse zu erhalten.

get_eks_vpc_config

Ruft die VPC-Konfiguration für einen EKS-Cluster ab, einschließlich Subnetzen, Routentabellen und Netzwerkkonnektivität.

Parameter:

- cluster_name (erforderlich): Name des EKS-Clusters, für den die VPC-Konfiguration abgerufen werden soll.
- vpc_id (optional): ID der spezifischen VPC, die abgefragt werden soll (optional, verwendet Cluster-VPC, falls nicht angegeben).

get_k8s_events

Ruft Kubernetes-Ereignisse ab, die sich auf bestimmte Ressourcen beziehen, zur Fehlerbehebung und Überwachung.

Parameter:

- cluster_name (erforderlich): Name des EKS-Clusters, in dem sich die Ressource befindet.
- kind (erforderlich): Art des beteiligten Objekts (z. B. „Pod“, „Deployment“, „Service“). Muss genau mit der Art der Ressource übereinstimmen.
- name (erforderlich): Name des beteiligten Objekts, für das Ereignisse abgerufen werden sollen.
- namespace (optional): Namespace des beteiligten Objekts. Erforderlich für Ressourcen mit Namespace (wie Pods, Deployments). Nicht erforderlich für Ressourcen im Clusterbereich (wie Nodes,). PersistentVolumes

get_pod_logs

Ruft Protokolle von Pods in einem EKS-Cluster mit Filteroptionen ab.

Parameter:

- cluster_name (erforderlich): Name des EKS-Clusters, auf dem der Pod ausgeführt wird.
- Namespace (erforderlich): Kubernetes-Namespace, in dem sich der Pod befindet.
- pod_name (erforderlich): Name des Pods, von dem Logs abgerufen werden sollen.

- container_name (optional): Name des spezifischen Containers, von dem Logs abgerufen werden sollen. Nur erforderlich, wenn der Pod mehrere Container enthält.
- limit_bytes (optional): Maximale Anzahl von Byte, die zurückgegeben werden sollen. Standard: 10 KB (10240 Byte).
- previous (optional): Gibt zuvor beendete Container-Logs zurück (standardmäßig falsch). Nützlich, um Protokolle für Pods abzurufen, die neu gestartet werden.
- since_seconds (optional): Gibt nur Protokolle zurück, die jünger als diese Anzahl von Sekunden sind. Nützlich, um aktuelle Protokolle abzurufen, ohne den gesamten Verlauf abzurufen.
- tail_lines (optional): Anzahl der Zeilen, die vom Ende der Protokolle zurückgegeben werden sollen. Standard: 100.

list_api_versions

Listet alle verfügbaren API-Versionen im angegebenen Kubernetes-Cluster auf.

Parameter:

- cluster_name (erforderlich): Name des EKS-Clusters.

list_k8s_resources

Listet Kubernetes-Ressourcen einer bestimmten Art in einem EKS-Cluster auf.

Parameter:

- cluster_name (erforderlich): Name des EKS-Clusters, in dem sich die Ressourcen befinden.
- kind (erforderlich): Art der aufzulistenden Kubernetes-Ressourcen (z. B. 'Pod', 'Service', 'Deployment'). Verwenden Sie das `list_api_versions` Tool, um verfügbare Ressourcentypen zu finden.
- api_version (erforderlich): API-Version der Kubernetes-Ressourcen (z. B. 'v1', '1'). apps/v1', 'networking.k8s.io/v Verwenden Sie das `list_api_versions` Tool, um verfügbare API-Versionen zu finden.
- field_selector (optional): Feldauswahl zum Filtern von Ressourcen (z. B. 'metadata.name=my-pod, status.phase=Running'). Verwendet dieselbe Syntax wie das Flag --field-selector von kubectl.
- label_selector (optional): Label-Selektor zum Filtern von Ressourcen (z. B. 'app=nginx, tier=frontend'). Verwendet dieselbe Syntax wie `--selector` das Flag von kubectl.

- namespace (optional): Namespace der aufzulistenden Kubernetes-Ressourcen. Falls nicht angegeben, werden Ressourcen in allen Namespaces aufgelistet (für Ressourcen mit Namespaces).

read_k8s_resource

Ruft detaillierte Informationen zu einer bestimmten Kubernetes-Ressource in einem EKS-Cluster ab.

Parameter:

- api_version (erforderlich): API-Version der Kubernetes-Ressource (z. B. „v1"," 1"). apps/v1", "networking.k8s.io/v
- cluster_name (erforderlich): Name des EKS-Clusters, in dem sich die Ressource befindet.
- kind (erforderlich): Art der Kubernetes-Ressource (z. B. „Pod“, „Service“, „Deployment“).
- name (erforderlich): Name der zu lesenden Kubernetes-Ressource.
- namespace (optional): Namespace der Kubernetes-Ressource. Erforderlich für Ressourcen mit Namespace. Nicht erforderlich für Ressourcen im Clusterbereich (wie Nodes,). PersistentVolumes

generate_app_manifest

Generiert standardisierte Kubernetes-Bereitstellungs- und Dienstmanifeste für containerisierte Anwendungen.

Parameter:

- app_name (erforderlich): Name der Anwendung. Wird für Bereitstellungs- und Dienstnamen sowie für Labels verwendet.
- image_uri (erforderlich): Vollständige ECR-Image-URI mit Tag (z. B. 123456789012.dkr.ecr.region.amazonaws.com/repo:tag). Muss den vollständigen Repository-Pfad und das Tag enthalten.
- load_balancer_scheme (optional): Load Balancer-Schema. AWS Zulässige Werte:
- „intern“ (nur private VPC)
- „mit dem Internet verbunden“ (öffentlicher Zugang).
- Standard: „intern“.
- cpu (optional): CPU-Anfrage für jeden Container (z. B. „100 m“ für 0,1 CPU-Kerne, „500 m“ für einen halben Kern). Standard: „100 m“.

- memory (optional): Speicheranforderung für jeden Container (z. B. „128Mi“ für 128 MiB, „1Gi“ für 1 GiB). Standard: „128Mi“
- Namespace (optional): Kubernetes-Namespace, in dem die Anwendung bereitgestellt werden soll. Standard: „Standard“.
- port (optional): Container-Port, auf dem die Anwendung lauscht. Standard: 80
- Replikate (optional): Anzahl der bereitzustellenden Replikate. Standard: 2

get_cloudwatch_logs

Fragt CloudWatch Protokolle ab, wobei die Filterung auf der Grundlage der Eingabeparameter erfolgt und Standardprotokollgruppen unterstützt werden, die für die EKS-Cluster-Observability verwendet werden.

Parameter:

- cluster_name (erforderlich): Name des EKS-Clusters, in dem sich die Ressource befindet. Wird verwendet, um den Namen der CloudWatch Protokollgruppe zu erstellen.
- resource_type (erforderlich): Ressourcentyp, nach dem Logs durchsucht werden sollen. Gültige Werte: +"pod“, „node“, „container“, „cluster“. Dies bestimmt, wie Protokolle gefiltert werden.
- log_type (erforderlich): Der abzufragende Protokolltyp. Zulässige Werte:
- „Anwendung“: Protokolle Container/application
- „host“: Systemprotokolle auf Knotenebene
- „Leistung“: Protokolle zu Leistungsmetriken
- „Steuerebene“: Protokolle der EKS-Steuerungsebene
- "your-log-group-name„: Geben Sie direkt einen benutzerdefinierten CloudWatch Protokollgruppennamen an.
- resource_name (optional): Ressourcenname, nach dem in Protokollnachrichten gesucht werden soll (z. B. Pod-Name, Knotenname, Containername). Wird verwendet, um Logs nach der spezifischen Ressource zu filtern.
- Minuten (optional): Anzahl der Minuten, für die nach Protokollen gesucht werden soll. Standard: 15. Wird ignoriert, wenn start_time angegeben ist. Verwenden Sie kleinere Werte für aktuelle Probleme und größere Werte für historische Analysen.
- start_time (optional): Startzeit im ISO-Format (z. B. „2023-01-01T 00:00:00 Z“). Falls angegeben, überschreibt er den Minuten-Parameter.

- end_time (optional): Endzeit im ISO-Format (z. B. „2023-01-01T 01:00:00 Z“). Falls nicht angegeben, wird standardmäßig die aktuelle Uhrzeit verwendet.
- Felder (optional): Benutzerdefinierte Felder, die in die Abfrageergebnisse aufgenommen werden sollen (standardmäßig "@timestamp, @message „). Verwenden Sie die CloudWatch Logs Insights-Feldsyntax.
- filter_pattern (optional): Zusätzliches anzuwendendes CloudWatch Logs-Filtermuster. Verwendet die CloudWatch Logs Insights-Syntax (z. B. „ERROR“, „field=value“).
- limit (optional): Maximale Anzahl zurückzugebender Protokolleinträge. Verwenden Sie niedrigere Werte (10-50) für schnellere Abfragen, höhere Werte (100-1000) für umfassendere Ergebnisse. Höhere Werte wirken sich negativ auf die Performance aus.

get_cloudwatch_metrics

Ruft CloudWatch Metriken und Datenpunkte für die EKS-Clusterüberwachung und Leistungsanalyse ab. Verarbeitet Container Insights-Metriken, benutzerdefinierte Metriken und konfigurierbare Zeiträume und Dimensionen.

Parameter:

- cluster_name (erforderlich): Name des EKS-Clusters, für den Metriken abgerufen werden sollen.
- dimensions (erforderlich): Dimensionen, die für die CloudWatch Metrikabfrage als JSON-Zeichenfolge verwendet werden sollen. Muss geeignete Dimensionen für den Ressourcentyp und die Metrik enthalten (z. B. '{“ ClusterName „: „my-cluster“, "PodName„: „my-pod“, „Namespace“: „default"}').
- metric_name (erforderlich): Name der Metrik, die abgerufen werden soll. Allgemeine Beispiele:
- `cpu_usage_total`: CPU-Auslastung insgesamt
- `memory_rss`: Speichernutzung von Resident Set Size
- `network_rx_bytes`: Empfangene Netzwerk-Bytes
- `network_tx_bytes`: Übertragene Netzwerk-Bytes
- Namespace (erforderlich): CloudWatch Namespace, in dem die Metrik gespeichert ist. Allgemeine Werte:
- `"ContainerInsights"`: Für Container-Metriken
- `"AWS/EC2"`: EC2 Zum Beispiel Metriken
- `"AWS/EKS"`: Für Metriken der EKS-Kontrollebene

- Minuten (optional): Anzahl der Minuten, für die nach Metriken gesucht werden soll. Standard: 15. Wird ignoriert, wenn start_time angegeben ist.
- start_time (optional): Startzeit im ISO-Format (z. B. „2023-01-01T 00:00:00 Z“). Falls angegeben, überschreibt er den Minuten-Parameter.
- end_time (optional): Endzeit im ISO-Format (z. B. „2023-01-01T 01:00:00 Z“). Falls nicht angegeben, wird standardmäßig die aktuelle Uhrzeit verwendet.
- limit (optional): Maximale Anzahl zurückzugebender Datenpunkte. Höhere Werte (100-1000) liefern detailliertere Daten, können sich jedoch auf die Leistung auswirken. Standard: 50.
- Zeitraum (optional): Zeitraum in Sekunden für die metrischen Datenpunkte. Standard: 60 (1 Minute). Niedrigere Werte (1—60) bieten eine höhere Auflösung, sind aber möglicherweise weniger verfügbar.
- stat (optional): Statistik, die für die Metrikaggregation verwendet werden soll. Standard: „Durchschnitt“. Zulässige Werte:
- `Average`: Mittelwert während des Zeitraums
- `Sum`: Gesamtwert während des Zeitraums
- `Maximum`: Höchster Wert während des Zeitraums
- `Minimum`: Niedrigster Wert während des Zeitraums
- `SampleCount`: Anzahl der Proben während des Zeitraums.

get_eks_metrics_guidance

Hier finden Sie Anleitungen zu CloudWatch Metriken für bestimmte Ressourcentypen in EKS-Clustern. Hilfreich für den Agenten bei der Bestimmung der richtigen Dimensionen für die Verwendung mit dem Tool get_cloudwatch_metrics.

Parameter:

- resource_type (erforderlich): Art der Ressource, für die Metriken abgerufen werden sollen (Cluster, Knoten, Pod, Namespace, Dienst).

get_policies_for_role

Ruft alle Richtlinien ab, die einer bestimmten IAM-Rolle zugeordnet sind, einschließlich Richtlinien zur Übernahme von Rollen, verwalteten Richtlinien und Inline-Richtlinien.

Parameter:

- role_name (erforderlich): Name der IAM-Rolle, für die Richtlinien abgerufen werden sollen. Die Rolle muss in Ihrem Konto vorhanden sein. AWS

Tools mit vollem Zugriff (Schreib-)

In diesem Abschnitt werden die schreibgeschützten Tools beschrieben, die für den EKS MCP-Server verfügbar sind. Beachten Sie, dass (Stand „heute“) alle Kubernetes-API-Schreiboperationen nur auf Folgendes zugreifen können:

- Öffentliche Cluster (=true) endpointPublicAccess

manage_k8s_resource

Verwaltet eine einzelne Kubernetes-Ressource mit Schreiboperationen (erstellen, aktualisieren, patchen oder löschen).

Parameter:

- Operation (erforderlich): Vorgang, der für die Ressource ausgeführt werden soll. Zulässige Werte:
- `create:`Eine neue Ressource erstellen
- `replace`: Ersetzt eine bestehende Ressource
- `patch`: Aktualisiere bestimmte Felder einer vorhandenen Ressource
- `delete`: Löscht eine bestehende Ressource
- Hinweis: Verwenden Sie read_k8s_resource zum Lesen von Ressourcen und list_k8s_resources zum Auflisten mehrerer Ressourcen.
- cluster_name (erforderlich): Name des EKS-Clusters, in dem sich die Ressource befindet oder erstellt wird.
- kind (erforderlich): Art der Kubernetes-Ressource (z. B. „Pod“, „Service“, „Deployment“).
- api_version (erforderlich): API-Version der Kubernetes-Ressource (z. B. „v1"," 1"). apps/v1", "networking.k8s.io/v
- body (optional): Ressourcendefinition als JSON-Zeichenfolge. Erforderlich für Erstellungs-, Ersetzungs- und Patch-Operationen. Für Erstellen und Ersetzen sollte dies eine vollständige Ressourcendefinition sein. Bei Patches sollte dies nur die zu aktualisierenden Felder enthalten.
- name (optional): Name der Kubernetes-Ressource. Erforderlich für alle Operationen außer create (wo es im Hauptteil angegeben werden kann).

- namespace (optional): Namespace der Kubernetes-Ressource. Erforderlich für Ressourcen mit Namespace. Nicht erforderlich für Ressourcen im Clusterbereich (wie Nodes,). PersistentVolumes

apply_yaml

Wendet Kubernetes-YAML-Manifeste auf einen EKS-Cluster an.

Parameter:

- cluster_name (erforderlich): Name des EKS-Clusters, in dem die Ressourcen erstellt oder aktualisiert werden.
- Namespace (erforderlich): Kubernetes-Namespace, auf den Ressourcen angewendet werden sollen. Wird für Ressourcen mit Namespace verwendet, die keinen Namespace angeben.
- yaml_content (erforderlich): YAML-Inhalt, der auf den Cluster angewendet werden soll. Kann mehrere Dokumente enthalten, die durch '---' getrennt sind.
- force (optional): Ob Ressourcen aktualisiert werden sollen, falls sie bereits existieren (ähnlich wie bei kubectl apply). Auf false setzen, um nur neue Ressourcen zu erstellen.

manage_eks_stacks

Verwaltet CloudFormation EKS-Stacks mit Vorgängen zum Generieren von Vorlagen, zum Bereitstellen, Beschreiben und Löschen von EKS-Clustern und der zugrunde liegenden Infrastruktur. Die Clustererstellung dauert in der Regel 15 bis 20 Minuten. Für Bereitstellungs- und Löschvorgänge muss der Stack mit diesem Tool erstellt worden sein (d. h. mit CreatedBy = gekennzeichnetEksMcpServer).

Parameter:

- cluster_name (erforderlich): Name des EKS-Clusters (für Generierungs-, Bereitstellungs-, Beschreib- und Löschvorgänge). Dieser Name wird zur Ableitung des CloudFormation Stack-Namens verwendet und in die Clusterressourcen eingebettet.
- Operation (erforderlich): Durchzuführender Vorgang. Zulässige Werte:
- `generate`: Generieren Sie eine CloudFormation Vorlage
- `deploy`: Stellen Sie einen CloudFormation Stack bereit (erforderlich`template_content`)
- `describe`: Describe/read ein CloudFormation Stapel (schreibgeschützt)
- `delete`: Löscht einen Stapel CloudFormation

- template_content (optional): Inhalt der CloudFormation Vorlage (für Bereitstellungsvorgänge). Dies sollte der vollständige Inhalt der YAML- oder JSON-Vorlage sein. Unterstützt sowohl einzelne Ressourcen als auch YAML-Inhalte mit mehreren Dokumenten, getrennt durch „---“.

add_inline_policy

Fügt einer IAM-Rolle eine neue Inline-Richtlinie hinzu.

Parameter:

- Berechtigungen (erforderlich): Berechtigungen, die als JSON-Zeichenfolgen, die IAM-Richtlinienanweisungen darstellen, in die Richtlinie aufgenommen werden sollen. Dabei kann es sich entweder um eine einzelne JSON-Zeichenfolge oder um ein Array von JSON-Zeichenfolgen handeln.
- policy_name (erforderlich): Name der zu erstellenden Inline-Richtlinie. Muss innerhalb der Rolle eindeutig sein.
- role_name (erforderlich): Name der IAM-Rolle, zu der die Richtlinie hinzugefügt werden soll. Die Rolle muss existieren.

Verwenden Sie Amazon Q Developer auf der Amazon EKS-Konsole

Amazon Elastic Kubernetes Service (EKS) ist in Amazon Q integriert, um KI-gestützte Problemlösungen direkt in der AWS Management Console bereitzustellen. Diese Integration hilft Ihnen, Cluster-, Steuerungsebene-, Knoten- und Workload-Probleme mit KI-Unterstützung von Amazon Q schnell zu untersuchen und zu lösen.

Funktionsweise

In der Amazon EKS-Konsole werden die Schaltflächen „Inspect with Amazon Q“ kontextbezogen neben Fehlern oder Problemen in der gesamten Konsole angezeigt. Wenn Sie auf diese Schaltfläche klicken, analysiert Amazon Q das Problem automatisch und öffnet auf der rechten Seite der Konsole ein Chat-Panel mit Untersuchungsergebnissen, Ursachenanalysen und vorgeschlagenen Maßnahmen zur Schadensbegrenzung.

Die Integration wird an den folgenden Stellen in der Amazon EKS-Konsole angezeigt:

- Clusterstatus — Untersuchen Sie Probleme mit dem Clusterstatus und Statusmeldungen auf der Registerkarte Clusterstatus im Observability-Dashboard
- Kontrollebene — Beheben Sie Fehler und Warnungen von Komponenten der Kontrollebene bei der Überwachung der Kontrollebene im Observability-Dashboard
- Einblicke in das Upgrade — Analysieren Sie potenzielle Upgrade-Blocker und Kompatibilitätsprobleme unter Upgrade-Einblicke im Observability-Dashboard
- Zustand des Knotens — Untersuchen Sie Probleme auf Knotenebene, die sich auf die Clusterkapazität auswirken, unter Probleme mit dem Knotenstatus im Observability-Dashboard
- Workloads — Analysieren Sie Kubernetes-Ereignisse auf Pods, die auf Ausfälle oder Probleme hinweisen

Verwendung von Amazon Q zur Fehlerbehebung

Um ein Problem mit Amazon Q zu untersuchen

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Wählen Sie den Namen des Clusters, den Sie untersuchen möchten.
3. Wenn Sie auf eine Fehlermeldung oder einen Problemindikator stoßen, suchen Sie nach der Schaltfläche Mit Amazon Q überprüfen. Die Schaltfläche wird kontextbezogen neben dem Problem oder in der Detailansicht zum Fehlerstatus angezeigt.
4. Wählen Sie Inspect with Amazon Q.
5. Amazon Q untersucht das Problem automatisch und zeigt die Analyse in einem Chat-Panel auf der rechten Seite der Konsole an.
6. Überprüfen Sie die Untersuchungsergebnisse, einschließlich der Ursachenanalyse und der vorgeschlagenen Maßnahmen zur Schadensbegrenzung.
7. Sie können das Gespräch fortsetzen, indem Sie Amazon Q weitere Fragen zu dem Problem stellen.

Hinweis Die Amazon Q-Integration wird nur angezeigt, wenn ein Fehler, eine Warnung oder ein Problem vorliegt, das untersucht werden muss. Sie wird nicht angezeigt, wenn die Ressourcen intakt sind.

Überlegungen

Beachten Sie Folgendes, wenn Sie Amazon Q mit Amazon EKS verwenden:

- Schreibgeschützte Operationen — Die Amazon Q-Integration führt nur Lesevorgänge auf Ihren Cluster-Ressourcen durch. Sie führt keine Mutations- oder Schreibaktionen in Ihrer Cluster-Konfiguration oder Ihren Workloads durch.
- Regionsübergreifende Verarbeitung — Amazon Q kann Daten AWS regionsübergreifend verarbeiten, um KI-gestützte Analysen bereitzustellen. Weitere Informationen zur regionsübergreifenden Verarbeitung finden Sie unter Regionsübergreifende Verarbeitung im Amazon Q Developer User Guide.
- AWS Nur Management Console — Diese Integration ist nur über die AWS Management Console verfügbar. Es ist nicht über die AWS CLI oder die Infrastruktur als Code-Tools verfügbar. AWS APIs

Weitere Informationen

Weitere Informationen zur Verwendung von Amazon Q finden Sie unter Chat mit Amazon Q im Amazon Q Developer User Guide.

Versionsverwaltung in Amazon EKS

Dieser Abschnitt soll Ihnen dabei helfen, die Funktionsweise der Kubernetes- Versionsverwaltung innerhalb des Amazon Elastic Kubernetes Service (EKS) zu verstehen. Sie finden Details zu unterstützten Versionen, unseren Veraltungs- und Supportrichtlinien, Upgrade-Prozessen und wichtigen Überlegungen zur Verwaltung Ihrer Cluster-Versionen.

Themen

- Verständnis des Kubernetes-Versionslebenszyklus auf EKS
- Anzeigen der Amazon-EKS-Plattformversionen für jede Kubernetes-Version

Verständnis des Kubernetes-Versionslebenszyklus auf EKS

> **Tip**
>
> Melden Sie sich für bevorstehende Amazon EKS-Workshops an.

Kubernetes entwickelt sich durch neue Features, Design-Aktualisierungen und Fehlerbehebungen schnell weiter. Die Community veröffentlicht im Durchschnitt alle vier Monate neue Kubernetes-Nebenversionen (z. B. `1.33`). Amazon EKS folgt für Nebenversionen dem Upstream-Release- und Einstellungszyklus. Sobald neue Kubernetes-Versionen in Amazon EKS verfügbar werden, empfehlen wir Ihnen, Ihre Cluster proaktiv auf die neueste verfügbare Version zu aktualisieren.

Eine Nebenversion wird in den ersten 14 Monaten nach ihrer Veröffentlichung standardmäßig in Amazon EKS unterstützt. Sobald eine Version das Ende des Standard-Supportdatums überschritten hat, wird für die folgenden 12 Monate ein erweiterter Support angeboten. Mit dem erweiterten Support können Sie gegen Aufpreis pro Cluster-Stunde länger bei einer bestimmten Kubernetes-Version bleiben. Wenn Sie Ihren Cluster vor Ablauf des verlängerten Supportzeitraums nicht aktualisiert haben, wird Ihr Cluster automatisch auf die älteste derzeit unterstützte erweiterte Version aktualisiert.

Der erweiterte Support ist standardmäßig aktiviert. Informationen zum Deaktivieren finden Sie unter Erweiterten EKS-Support deaktivieren.

Wir empfehlen, dass Sie Ihren Cluster mit der neuesten verfügbaren Kubernetes-Version erstellen, die von Amazon EKS unterstützt wird. Wenn Ihre Anwendung eine bestimmte Version von

Kubernetes benötigt, können Sie auch ältere Versionen auswählen. Sie können neue Amazon-EKS-Cluster auf jeder Version erstellen, die mit Standard- oder verlängertem Support angeboten wird.

Verfügbare Versionen mit Standard-Support

Die folgenden Kubernetes-Versionen sind derzeit mit Amazon-EKS-Standard-Support verfügbar:

- `1.34`
- `1.33`
- `1.32`

Wichtige Änderungen, die Sie für jede Version des Standard-Supports beachten sollten, finden Sie unter Standard-Support für Kubernetes-Versionen.

Verfügbare Versionen mit verlängertem Support

Die folgenden Kubernetes-Versionen sind derzeit mit erweitertem Support für Amazon EKS verfügbar:

- `1.31`
- `1.30`
- `1.29`

Wichtige Änderungen, die Sie bei jeder Version im erweiterten Support beachten sollten, finden Sie unter Erweiterter Support für Kubernetes-Versionen.

Amazon-EKS-Kubernetes-Veröffentlichungskalender

Die folgende Tabelle zeigt wichtige Veröffentlichungs- und Support-Daten, die für jede -Version zu berücksichtigen sind. Fakturierung für den erweiterten Support beginnt zu Beginn des Tages, an dem die Version das Ende des Standard-Supports erreicht, in der Zeitzone UTC+0. Die Daten in der folgenden Tabelle basieren auf der Zeitzone UTC+0.

> **Note**
>
> Daten mit nur einem Monat und einem Jahr sind ungefähre Angaben und werden mit einem genauen Datum aktualisiert, wenn es bekannt ist.

Um Benachrichtigungen über alle Änderungen an den Quelldateien dieser spezifischen Dokumentationsseite zu erhalten, können Sie die folgende URL mit einem RSS-Reader abonnieren:

```
https://github.com/awsdocs/amazon-eks-user-guide/commits/mainline/latest/ug/versioning/kubernetes-versions.adoc.atom
```

Kubernetes-Version	Upstream-Release	Amazon-EKS-Version	Ende des Standard-Supports	Ende des verlängerten Supports
1.34	27. August 2025	2. Oktober 2025	2. Dezember 2026	2. Dezember 2027
1.33	23. April 2025	29. Mai 2025	29. Juli 2026	29. Juli 2027
1.32	11. Dezember 2024	23. Januar 2025	23. März 2026	23. März 2027
1.31	13. August 2024	26. September 202	26. November 202	26. November 2026
1.30	17. April 2024	23. Mai 2024	23. Juli 2025	23. Juli 2026
1.29	13. Dezember 2023	23. Januar 2024	23. März 2025	23. März 2026

Versionsinformationen mit AWS CLI abrufen

Sie können die AWS CLI verwenden, um Informationen zu den auf EKS verfügbaren Kubernetes-Versionen abzurufen, z. B. das Enddatum des Standardsupports.

So rufen Sie mithilfe der CLI Informationen zu verfügbaren Kubernetes-Versionen auf EKS ab AWS

1. Öffnen Sie Ihr Terminal.
2. Stellen Sie sicher, dass Sie die AWS CLI installiert und konfiguriert haben. Weitere Informationen finden Sie unter Installation oder Aktualisierung auf die neueste Version der CLI.
3. Führen Sie den folgenden Befehl aus:

```
aws eks describe-cluster-versions
```

4. Der Befehl gibt eine JSON-Ausgabe mit Details zu den verfügbaren Cluster-Versionen zurück. Nachfolgend finden Sie ein Beispiel für die Ausgabe:

```
{
    "clusterVersions": [
        {
            "clusterVersion": "1.31",
            "clusterType": "eks",
            "defaultPlatformVersion": "eks.21",
            "defaultVersion": true,
            "releaseDate": "2024-09-25T17:00:00-07:00",
            "endOfStandardSupportDate": "2025-11-25T16:00:00-08:00",
            "endOfExtendedSupportDate": "2026-11-25T16:00:00-08:00",
            "status": "STANDARD_SUPPORT",
            "kubernetesPatchVersion": "1.31.3"
        }
    ]
}
```

Die Ausgabe liefert für jede Cluster-Version die folgenden Informationen:

- `clusterVersion`: Die Kubernetes-Version des EKS-Clusters
- `clusterType`: Der Cluster-Typ (z. B. „eks“)
- `defaultPlatformVersion`: Die Standardversion der EKS-Plattform
- `defaultVersion`: Ob es sich um die Standardversion handelt
- `releaseDate`: Das Datum, an dem diese Version veröffentlicht wurde
- `endOfStandardSupportDate`: Das Datum, an dem der Standard-Support endet

- `endOfExtendedSupportDate`: Das Datum, an dem der erweiterte Support endet
- `status`: Der aktuelle Support-Status der Version, z. B. `STANDARD_SUPPORT` oder `EXTENDED_SUPPORT`
- `kubernetesPatchVersion`: Die spezifische Kubernetes-Patch-Version

Amazon EKS-Version FAQs

Wie viele Kubernetes-Versionen sind mit Standard-Support verfügbar?

Im Einklang mit der Unterstützung der Kubernetes-Community für Kubernetes-Versionen verpflichtet sich Amazon EKS, jederzeit Unterstützung für mindestens drei Kubernetes-Versionen anzubieten. Wir geben das Ende des Standard-Supports für eine bestimmte Kubernetes-Nebenversion mindestens 60 Tage im Voraus bekannt. Aufgrund des Amazon-EKS-Qualifizierungs- und Freigabeprozesses für neue Kubernetes-Versionen endet der Standard-Support für eine Kubernetes-Version in Amazon EKS nach dem Datum, an dem das Kubernetes-Projekt den Upstream-Support für die Version einstellt.

Wie lange erhält ein Kubernetes-Standard-Support von Amazon EKS?

Eine Kubernetes-Version wird ab der ersten Verfügbarkeit in Amazon EKS 14 Monate lang unterstützt. Dies gilt auch dann, wenn Kubernetes eine in Amazon EKS verfügbare Version nicht mehr unterstützt. Wir portieren Sicherheits-Patches, die für die in Amazon EKS unterstützten Kubernetes-Versionen relevant sind, zurück.

Werde ich benachrichtigt, wenn der Standard-Support für eine Kubernetes-Version in Amazon EKS endet?

Ja. Wenn auf Clustern in Ihrem Konto die Version ausgeführt wird, die sich dem Ende des Supports nähert, sendet Amazon EKS etwa 12 Monate nach der Veröffentlichung der Kubernetes-Version auf Amazon EKS eine Benachrichtigung über das AWS Health Dashboard. Die Mitteilung enthält das Datum des Support-Laufzeitendes. Dies ist mindestens 60 Tage ab dem Datum der Benachrichtigung.

Welche Kubernetes-Feature werden von Amazon EKS unterstützt?

Amazon EKS unterstützt alle allgemein verfügbaren (GA) Feature der Kubernetes-API. Neue Betaversionen APIs sind in Clustern standardmäßig nicht aktiviert. Zuvor vorhandene Betaversionen APIs und neue Versionen vorhandener Betaversionen sind jedoch APIs weiterhin standardmäßig aktiviert. Alpha-Features werden nicht unterstützt.

Werden von Amazon EKS verwaltete Knotengruppen automatisch zusammen mit der Version der Cluster-Kontrollebene aktualisiert?

Nein. Eine verwaltete Knotengruppe erstellt EC2 Amazon-Instances in Ihrem Konto. Diese Instances werden nicht automatisch aktualisiert, wenn Sie oder Amazon EKS Ihre Steuerebene aktualisieren. Weitere Informationen finden Sie unter the section called “Aktualisierung”. Wir empfehlen, dieselbe Kubernetes-Version auf Ihrer Steuerungsebene und Ihren Knoten beizubehalten.

Werden selbstverwaltete Knotengruppen automatisch zusammen mit der Version der Cluster-Kontrollebene aktualisiert?

Nein. Eine selbstverwaltete Knotengruppe umfasst EC2 Amazon-Instances in Ihrem Konto. Diese Instances werden nicht automatisch aktualisiert, wenn Sie oder Amazon EKS die Version der Steuerebene in Ihrem Namen aktualisieren. Für eine selbstverwaltete Knotengruppe gibt es in der Konsole keinen Hinweis darauf, dass sie aktualisiert werden muss. Sie können die auf einem Knoten installierte `kubelet`-Version anzeigen, indem Sie den Knoten in der Liste Knoten auf der Registerkarte Übersicht Ihres Clusters auswählen, um zu bestimmen, welche Knoten aktualisiert werden müssen. Sie müssen die Knoten manuell aktualisieren. Weitere Informationen finden Sie unter the section called “Update-Methoden”.

Das Kubernetes-Projekt überprüft die Kompatibilität zwischen der Steuerebene und den Knoten für bis zu drei Nebenversionen. `1.30`-Knoten funktionieren beispielsweise weiterhin, wenn sie von einer `1.33`-Steuerebene orchestriert werden. Es wird jedoch nicht empfohlen, einen Cluster mit Knoten auszuführen, die sich dauerhaft drei Nebenversionen hinter der Steuerebene befinden. Weitere Informationen finden Sie unter Kubernetes Version and Version Skew Support Policy in der Kubernetes-Dokumentation. Wir empfehlen, dieselbe Kubernetes-Version auf Ihrer Steuerungsebene und Ihren Knoten beizubehalten.

Werden Pods, die in Fargate ausgeführt werden, automatisch mit einem automatischen Upgrade der Cluster-Steuerebenen-Version aktualisiert?

Nein. Wir empfehlen dringend, Fargate-Pods als Teil eines Replikations-Controllers wie einer Kubernetes-Bereitstellung auszuführen. Führen Sie anschließend einen fortlaufenden Neustart aller Fargate-Pods durch. Die neue Version des Fargate-Pods wird mit einer `kubelet`-Version bereitgestellt, die dieselbe Version wie Ihre aktualisierte Version der Cluster-Steuerebene ist. Weitere Informationen finden Sie unter Bereitstellungen in der Kubernetes-Dokumentation.

> **Important**
>
> Wenn Sie die Steuerebene aktualisieren, müssen Sie die Fargate-Knoten nach wie vor selbst aktualisieren. Um Fargate-Knoten zu aktualisieren, löschen Sie den Fargate-Pod, der durch den Knoten repräsentiert wird, und stellen Sie den Pod erneut bereit. Der neue Pod wird mit einer `kubelet`-Version bereitgestellt, die der Version Ihres Clusters entspricht.

Welche Kubernetes-Versionen werden für Hybridknoten unterstützt?

Amazon EKS Hybrid Nodes unterstützt dieselben Kubernetes-Versionen wie Amazon-EKS-Cluster mit anderen Knoten-Rechentypen, einschließlich Support für Standard- und erweiterte Kubernetes-Versionen. Hybridknoten werden bei einer Aktualisierung Ihrer Steuerebenen-Version nicht automatisch aktualisiert. Sie sind für die Aktualisierung Ihrer Hybridknoten verantwortlich. Weitere Informationen finden Sie unter the section called “Hybridknoten aktualisieren”.

Erweiterter Support für Amazon EKS FAQs

Die Begriffe Standard-Support und erweiterter Support sind mir neu. Was bedeuten diese Begriffe?

Der Standard-Support für eine Kubernetes-Version in Amazon EKS beginnt mit der Veröffentlichung einer Kubernetes-Version in Amazon EKS und endet 14 Monate nach dem Veröffentlichungsdatum. Der erweiterte Support für eine Kubernetes-Version beginnt unmittelbar nach dem Ende des Standard-Supports und endet nach 12 Monaten. Beispielsweise endet der Standard-Support für die Version `1.23` in Amazon EKS am 11. Oktober 2023. Der erweiterte Support für die Version `1.23` beginnt am 12. Oktober 2023 und endet am 11. Oktober 2024.

Was muss ich tun, um erweiterten Support für Amazon-EKS-Cluster zu erhalten?

Sie müssen den erweiterten Support (siehe Erweiterter EKS-Support) für Ihren Cluster aktivieren, indem Sie die Cluster-Upgrade-Richtlinie auf EXTENDED ändern. Standardmäßig ist die Upgrade-Richtlinie für alle neuen und vorhandenen Cluster auf EXTENDED festgelegt, sofern nicht anders angegeben. Die Upgrade-Richtlinie für Ihren Cluster finden Sie unter Cluster-Upgrade-Richtlinie. Der Standard-Support beginnt mit der Veröffentlichung einer Kubernetes-Version in Amazon EKS und endet 14 Monate nach dem Veröffentlichungsdatum. Der erweiterte Support für eine Kubernetes-Version beginnt unmittelbar nach dem Ende des Standard-Supports und endet nach 12 Monaten.

Für welche Kubernetes-Versionen kann ich erweiterten Support erhalten?

Sie können Cluster auf jeder Version bis zu 12 Monate nach dem Ende des Standard-Supports für diese Version ausführen. Das bedeutet, dass jede Version 26 Monate lang in Amazon EKS unterstützt wird (14 Monate Standard-Support plus 12 Monate verlängerter Support).

Was ist, wenn ich den erweiterten Support nicht nutzen möchte?

Wenn Sie nicht automatisch für den erweiterten Support angemeldet werden möchten, können Sie Ihren Cluster auf eine Kubernetes-Version aktualisieren, die im Standard-Amazon-EKS-Support enthalten ist. Informationen zum Deaktivieren des erweiterten Supports finden Sie unter Erweiterten EKS-Support deaktivieren. Hinweis: Wenn Sie den erweiterten Support deaktivieren, wird Ihr Cluster am Ende des Standard-Supports automatisch aktualisiert.

Was passiert nach Ablauf des 12-monatigen erweiterten Supports?

Cluster, die in einer Kubernetes-Version ausgeführt werden, deren 26-monatiger Lebenszyklus (14 Monate Standard-Support plus 12 Monate erweiterter Support) abgelaufen ist, werden automatisch auf die nächste Version aktualisiert. Die automatische Aktualisierung umfasst ausschließlich die Kubernetes-Steuerebene. Wenn Sie über Knoten von EKS Auto Mode verfügen, werden diese möglicherweise automatisch aktualisiert. Selbstverwaltete Knoten und von EKS verwaltete Knotengruppen bleiben auf der vorherigen Version.

Nach Ablauf des verlängerten Supports können Sie mit der nicht unterstützten Version keine neuen Amazon-EKS-Cluster mehr erstellen. Vorhandene Steuerebenen werden von Amazon EKS durch einen schrittweisen Bereitstellungsprozess nach dem Datum des Support-Laufzeitendes automatisch auf die älteste unterstützte Version aktualisiert. Stellen Sie nach dem automatischen Update der Steuerungsebene sicher, dass Sie die Cluster-Add-Ons und EC2 Amazon-Knoten manuell aktualisieren. Weitere Informationen finden Sie unter the section called “Kubernetes-Version aktualisieren”.

Wann genau wird meine Steuerebene nach Ablauf des erweiterten Supports automatisch aktualisiert?

Amazon EKS kann keine spezifischen Zeitrahmen bereitstellen. Automatische Updates können jederzeit nach Ablauf des verlängerten Supports erfolgen. Sie erhalten vor der Aktualisierung keine Benachrichtigung. Wir empfehlen Ihnen, dass Sie Ihre Steuerebene proaktiv aktualisieren, ohne sich auf den automatischen Aktualisierungsprozess von Amazon EKS zu verlassen. Weitere Informationen finden Sie unter the section called “Kubernetes-Version aktualisieren”.

Kann ich meine Steuerebene auf unbestimmte Zeit auf einer Kubernetes-Version belassen?

Nein. Cloud-Sicherheit AWS hat höchste Priorität. Ab einem bestimmten Zeitpunkt (normalerweise nach einem Jahr) veröffentlicht die Kubernetes-Community keine Patches mehr für allgemeine Schwachstellen und Risiken (CVE) und rät von der Einreichung von CVEs für nicht unterstützte Versionen ab. Dies bedeutet, dass Schwachstellen, die speziell für eine ältere Version von Kubernetes gelten, möglicherweise nicht gemeldet werden. Daher können Cluster im Falle einer Schwachstelle ohne Vorankündigung und ohne Behebungsoptionen offengelegt werden. Amazon EKS lässt daher nicht zu, dass Steuerebenen auf einer Version verbleiben, die das Ende des erweiterten Supports erreicht hat.

Fallen zusätzliche Kosten für den erweiterten Support an?

Ja, für Amazon-EKS-Cluster, die im erweiterten Support ausgeführt werden, fallen zusätzliche Kosten an. Preisinformationen finden Sie im AWS Blog oder auf unserer Preisseite unter Preise für die erweiterte Unterstützung für Kubernetes-Versionen von Amazon EKS.

Was ist im erweiterten Support enthalten?

Amazon-EKS-Cluster im verlängerten Support erhalten fortlaufend Sicherheits-Patches für die Kubernetes-Steuerebene. Darüber hinaus wird Amazon EKS Patches für Amazon VPC CNI, `kube-proxy` und CoreDNS-Add-Ons für erweiterte Support-Versionen veröffentlichen. Amazon EKS wird auch Patches für AWS veröffentlichte Amazon EKS veröffentlichen, die AMIs für Amazon Linux, Bottlerocket und Windows optimiert sind, sowie für Amazon EKS Fargate-Knoten für diese Versionen. Alle Cluster im erweiterten Support erhalten weiterhin Zugriff auf technischen Support von AWS.

Gibt es im Rahmen des verlängerten Supports Einschränkungen bei Patches für Komponenten, die keine Kubernetes-Komponenten sind?

Extended Support deckt zwar alle Kubernetes-spezifischen Komponenten von ab AWS, bietet aber jederzeit nur Support für Amazon EKS, AWS das AMIs für Amazon Linux, Bottlerocket und Windows optimiert ist. Das bedeutet, dass Sie möglicherweise neuere Komponenten (wie Betriebssystem oder Kernel) auf Ihrem für Amazon EKS optimierten AMI haben werden, während Sie den verlängerten Support verwenden. Sobald Amazon Linux 2 beispielsweise 2025 das Ende seines Lebenszyklus erreicht, AMIs wird das für Amazon EKS optimierte Amazon Linux mit einem neueren Amazon Linux-Betriebssystem erstellt. Amazon EKS wird wichtige Abweichungen im Support-Lebenszyklus wie diese für jede Kubernetes-Version bekannt geben und dokumentieren.

Kann ich mit einer Version mit erweitertem Support neue Cluster erstellen?

Ja.

Versionshinweise für Kubernetes-Versionen mit Standard-Support anzeigen

> **Tip**
>
> Melden Sie sich für bevorstehende Amazon EKS-Workshops an.

Dieses Thema enthält wichtige Änderungen, die Sie für jede Kubernetes-Version im Standard-Support beachten sollten. Überprüfen Sie beim Upgrade sorgfältig die Änderungen, die zwischen der alten und der neuen Version für Ihren Cluster vorgenommen wurden.

Kubernetes 1.34

Kubernetes `1.34` ist jetzt in Amazon EKS verfügbar. Weitere Informationen zu Kubernetes `1.34` finden Sie in der offiziellen Versionsankündigung.

> **Important**
>
> - Containerd wurde in Version 1.34 zum Start auf 2.1 aktualisiert.
>   - Wenn nach dem Upgrade Probleme auftreten, lesen Sie die containerd Versionshinweise zu 2.1.
> - AWS veröffentlicht kein EKS-optimiertes Amazon Linux 2-AMI für Kubernetes 1.34.
>   - AWS fordert Sie auf, zu Amazon Linux 2023 zu migrieren. Weitere Informationen erhalten Sie unter the section called “Upgrade auf AL2 023”.
>   - Weitere Informationen finden Sie unter the section called “Veraltung von Amazon Linux 2 AMI”.
> - AppArmor ist in Kubernetes 1.34 veraltet.
>   - Wir empfehlen die Migration zu alternativen Container-Sicherheitslösungen wie seccomp oder Pod Security Standards.
> - VolumeAttributesClass (VAC) wechselt in Kubernetes 1.34 zu GA und migriert von der Beta-API (`storage.k8s.io/v1beta1`) zur stabilen API (). `storage.k8s.io/v1`
>   - Wenn Sie den EBS-CSI-Treiber mit AWS-verwalteten Sidecar-Containern (von CSI Components in der ECR-Galerie) verwenden, funktioniert die Volumenänderung weiterhin problemlos auf EKS 1.31-1.33-Clustern. AWS wird die Sidecars so patchen, dass sie APIs bis zum Ende der Standardunterstützung für EKS 1.33 (29. Juli 2026) Beta VAC unterstützen.

    - Wenn Sie Ihre CSI-Sidecar-Container selbst verwalten, müssen Sie möglicherweise ältere Sidecar-Versionen auf Clustern vor 1.34 verwenden, um die VAC-Funktionalität aufrechtzuerhalten.
    - Um VolumeAttributesClass GA-Funktionen (z. B. Modifikations-Rollback) zu verwenden, führen Sie ein Upgrade auf EKS 1.34 oder höher durch.
  - External JWT Signer for Service Account Tokens wird in die Beta-Version hochgestuft. Bei der Verwendung externer Unterzeichner wird das service-account-extend-token Ablaufkennzeichen -- nicht mehr vollständig eingehalten. Der API-Server erzwingt den Mindestablauf zwischen der gewünschten Verlängerung (1 Jahr) und dem Limit für externe Unterzeichner (24 Stunden).
    - Wir empfehlen die Verwendung gebundener Dienstkonto-Tokens, die automatisch von Kubernetes bereitgestellt und rotiert werden.

- Dynamic Resource Allocation (DRA) Core APIs (GA): Die dynamische Ressourcenzuweisung ist inzwischen stabil und ermöglicht eine effiziente Verwaltung spezialisierter Hardware, z. B. GPUs über standardisierte Zuweisungsschnittstellen. Dadurch wird das Ressourcenmanagement für Hardwarebeschleuniger vereinfacht und die Nutzung spezialisierter Ressourcen verbessert.
- Projizierte ServiceAccount Tokens für Kubelet (Beta): Diese Erweiterung verbessert die Sicherheit, indem kurzlebige Anmeldeinformationen für Container-Image-Pulls anstelle von langlebigen Geheimnissen verwendet werden. Dadurch wird das Risiko der Offenlegung von Anmeldeinformationen verringert und die allgemeine Sicherheitslage Ihrer Cluster gestärkt.
- Ressourcenanforderungen und Grenzwerte auf Pod-Ebene (Beta): Diese Funktion vereinfacht die Ressourcenverwaltung, indem sie gemeinsam genutzte Ressourcenpools für Pods mit mehreren Containern ermöglicht und so eine effizientere Ressourcenzuweisung und -nutzung für komplexe Anwendungen mit mehreren Containern ermöglicht.
- Mutable CSI Node Allocatable Count (Beta): Das `MutableCSINodeAllocatableCount` Feature-Gate ist in EKS 1.34 standardmäßig aktiviert, wodurch das Attribut CSINode Max Attachable Volume Count veränderbar ist und ein Mechanismus eingeführt wird, mit dem es dynamisch auf der Grundlage der Benutzerkonfiguration auf CSI-Treiberebene aktualisiert werden kann. Diese Updates können entweder durch regelmäßige Intervalle oder durch Fehlererkennung ausgelöst werden. Dadurch wird die Zuverlässigkeit des Stateful-Pod-Schedulings erhöht, indem Diskrepanzen zwischen der gemeldeten und der tatsächlichen Verbindungskapazität auf den Knoten behoben werden.

  - Weitere Informationen finden Sie unter Kubernetes v1.34: Mutable CSI Node Allocatable Count im Kubernetes-Blog.
- Veraltungshinweis — Cgroup-Treiberkonfiguration: Die manuelle Konfiguration des Cgroup-Treibers wird zugunsten der automatischen Erkennung eingestellt.
  - Auswirkung auf Kunden: Wenn Sie das `--cgroup-driver` Flag derzeit in Ihrer Kubelet-Konfiguration manuell setzen, sollten Sie sich darauf vorbereiten, diese Konfiguration zu entfernen.
  - Erforderliche Maßnahme: Planen Sie die Aktualisierung von Knoten-Bootstrap-Skripten und benutzerdefinierten AMI-Konfigurationen, um manuelle Cgroup-Treibereinstellungen zu entfernen, bevor die Funktion in einer future Kubernetes-Version entfernt wird.
  - Weitere Informationen finden Sie in der Dokumentation zum Cgroup-Treiber.

Das vollständige `1.34` Kubernetes-Changelog finden Sie unter -1.34.md https://github.com/kubernetes/ kubernetes/blob/master/CHANGELOG/CHANGELOG

Kubernetes 1.33

Kubernetes `1.33` ist jetzt in Amazon EKS verfügbar. Weitere Informationen zu Kubernetes `1.33` finden Sie in der offiziellen Versionsankündigung.

> **Important**
>
> - Die beta-Version der Kubernetes-API für die dynamische Ressourcenzuweisung ist aktiviert.
>   - Diese Beta-API verbessert die Erfahrung bei der Planung und Überwachung von Workloads, die Ressourcen erfordern, wie z. GPUs
>   - Die Beta-API wird von der Kubernetes-Community definiert und kann sich in zukünftigen Versionen von Kubernetes ändern.
>   - Lesen Sie die Feature-Phasen in der Kubernetes-Dokumentation sorgfältig durch, um zu verstehen, welche Auswirkungen die Verwendung der Beta-Version hat. APIs
> - AWS veröffentlicht kein EKS-optimiertes Amazon Linux 2-AMI für Kubernetes 1.33.
>   - AWS fordert Sie auf, zu Amazon Linux 2023 zu migrieren. Weitere Informationen erhalten Sie unter the section called “Upgrade auf AL2 023”.

  - Weitere Informationen finden Sie unter the section called “Veraltung von Amazon Linux 2 AMI”.

- Pod-Ressourcenanpassung vor Ort (Beta): Die Ressourcenanpassung vor Ort wurde in die Beta-Phase hochgestuft, sodass CPU- und Speicherressourcen für vorhandene Pods ohne Neustart dynamisch aktualisiert werden können. Dies ermöglicht die vertikale Skalierung von zustandsbehafteten Workloads ohne Ausfallzeiten und eine nahtlose Ressourcenanpassung basierend auf Datenverkehrsmustern.
- Sidecar-Container jetzt stabil: Sidecar-Container sind nun stabil und implementieren Sidecars als spezielle Init-Container mit `restartPolicy: Always`, die vor Anwendungs-Containern gestartet werden, während des gesamten Pod-Lebenszyklus ausgeführt werden und Muster zur Signalisierung des Betriebszustands unterstützen.
  - Weitere Informationen finden Sie unter Sidecar-Container in der Kubernetes-Dokumentation.
- Veraltete Endpunkt-API: Die Endpoint-API ist jetzt offiziell veraltet und gibt beim Zugriff Warnungen zurück. Migrieren Sie Workloads und Skripts, um stattdessen die EndpointSlices API zu verwenden, die moderne Funktionen wie Dual-Stack-Netzwerke unterstützt und mehrere pro Service verarbeitet. EndpointSlices
  - Weitere Informationen finden Sie im Kubernetes-Blog unter Kubernetes v1.33: Fortsetzung des Übergangs von Endpoints zu. EndpointSlice
- Support für Elastic Fabric Adapter: Die Standardsicherheitsgruppe für Amazon-EKS-Cluster unterstützt nun Datenverkehr für Elastic Fabric Adapter (EFA). Die Standardsicherheitsgruppe verfügt über eine neue Ausgangsregel, die EFA-Datenverkehr mit dem Ziel derselben Sicherheitsgruppe zulässt. Dadurch wird EFA-Datenverkehr innerhalb des Clusters ermöglicht.
  - Weitere Informationen finden Sie unter Elastic Fabric Adapter für AI/ML und HPC-Workloads auf Amazon EC2 im Amazon Elastic Compute Cloud-Benutzerhandbuch.

Das vollständige `1.33` Kubernetes-Changelog finden Sie unter -1.33.md https://github.com/kubernetes/ kubernetes/blob/master/CHANGELOG/CHANGELOG

Kubernetes 1.32

Kubernetes `1.32` ist jetzt in Amazon EKS verfügbar. Weitere Informationen zu Kubernetes `1.32` finden Sie in der offiziellen Versionsankündigung.

⚠ Important

- Die `flowcontrol.apiserver.k8s.io/v1beta3` API-Version von FlowSchema und PriorityLevelConfiguration wurde in Version entfernt. `1.32` Wenn Sie diese verwenden APIs, müssen Sie Ihre Konfigurationen aktualisieren, um die neueste unterstützte Version zu verwenden, bevor Sie das Upgrade durchführen.
- ServiceAccount `metadata.annotations[kubernetes.io/enforce-mountable-secrets]`ist in der Version veraltet `1.32` und wird in einer future Kubernetes-Nebenversion entfernt. Es wird empfohlen, separate Namespaces zu verwenden, um den Zugriff auf eingebundene Geheimnisse zu isolieren.
- Die Kubernetes-Version `1.32` ist die letzte Version, für die Amazon EKS Amazon Linux 2 () AL2 veröffentlichen wird. AMIs `1.33`Ab Version 2 wird Amazon EKS weiterhin Amazon Linux 2023 (AL2023) und Bottlerocket auf Basis von Amazon veröffentlichen. AMIs

- Das Memory-Manager-Feature hat in Kubernetes Version `1.32` den Status „Generally Available“ (GA) erreicht. Diese Verbesserung ermöglicht eine effizientere und besser vorhersehbare Speicherzuweisung für containerisierte Anwendungen. Dies ist besonders vorteilhaft für Workloads mit spezifischen Speicheranforderungen.
- PersistentVolumeClaims (PVCs), die StatefulSets inzwischen erstellt wurden, beinhalten automatische Bereinigungsfunktionen. Wenn sie nicht mehr benötigt PVCs werden, werden sie automatisch gelöscht, wobei die Datenpersistenz bei StatefulSet Updates und Knotenwartungsvorgängen erhalten bleibt. Diese Funktion vereinfacht die Speicherverwaltung und verhindert, dass PVCs in Ihrem Cluster verwaiste Speicher entstehen.
- Es wurde die Funktion „Benutzerdefinierte Ressourcenfeldauswahl“ eingeführt, mit der Entwickler Feldauswahlfunktionen zu benutzerdefinierten Ressourcen hinzufügen können. Dieses Feature bietet dieselben Filterfunktionen, die für integrierte Kubernetes-Objekte verfügbar sind, auch für benutzerdefinierte Ressourcen. Dadurch wird eine präzisere und effizientere Ressourcenfilterung ermöglicht und ein besseres API-Design gefördert.

Das vollständige `1.32` Kubernetes-Changelog finden Sie unter -1.32.md https://github.com/kubernetes/ kubernetes/blob/master/CHANGELOG/CHANGELOG

Änderungen bei der anonymen Authentifizierung

Beim Einstieg mit Amazon EKS `1.32` ist die anonyme Authentifizierung auf die folgenden Endpunkte für die Zustandsprüfung des API-Serverstatus beschränkt:

- `/healthz`
- `/livez`
- `/readyz`

Anfragen an andere Endpunkte, die den `system:unauthenticated`-Benutzer verwenden, erhalten eine `401 Unauthorized`-HTTP-Antwort. Diese Sicherheitsverbesserung trägt dazu bei, unbeabsichtigten Cluster-Zugriff zu verhindern, der aufgrund falsch konfigurierter RBAC-Richtlinien auftreten könnte.

> **Note**
>
> Die `public-info-viewer`-RBAC-Rolle gilt weiterhin für die oben aufgeführten Endpunkte der Zustandsprüfung.

Veraltung von Amazon Linux 2 AMI

Die Kubernetes-Version `1.32` ist die letzte Version, für die Amazon EKS veröffentlicht wurde. AL2 AMIs `1.33`Ab Version 2 wird Amazon EKS weiterhin Version AL2 023 veröffentlichen und auf Bottlerocket basieren. AMIs Weitere Informationen finden Sie unter the section called “AL2 AMI-Abwertung”.

Versionshinweise für Kubernetes-Versionen mit erweitertem Support anzeigen

Amazon EKS unterstützt Kubernetes-Versionen länger als Upstream-Versionen, mit Standard-Support für Kubernetes-Nebenversionen für 14 Monate ab dem Zeitpunkt ihrer Veröffentlichung in Amazon EKS und erweitertem Support für Kubernetes-Nebenversionen für weitere 12 Monate Support (insgesamt 26 Monate pro Version).

Dieses Thema erläutert wichtige Änderungen, die Sie für jede Kubernetes-Version des verlängerten Supports beachten sollten. Überprüfen Sie beim Upgrade sorgfältig die Änderungen, die zwischen der alten und der neuen Version für Ihren Cluster vorgenommen wurden.

Kubernetes 1.31

Kubernetes `1.31` ist jetzt in Amazon EKS verfügbar. Weitere Informationen zu Kubernetes `1.31` finden Sie in der offiziellen Versionsankündigung.

> **Important**
>
> - Das seit 2017 veraltete Kubelet-Flag `--keep-terminated-pod-volumes` wurde im Rahmen der Veröffentlichung der Version `1.31` entfernt. Diese Änderung wirkt sich darauf aus, wie gekündigte Pod-Volumes vom Kubelet verarbeitet werden. Wenn Sie dieses Flag in Ihren Knotenkonfigurationen verwenden, müssen Sie Ihre Bootstrap-Skripte und Startvorlagen aktualisieren, um es vor der Aktualisierung zu entfernen.

- Das Beta-Feature Gate `VolumeAttributesClass` und die API-Ressource sind in der Amazon-EKS-Version `1.31` aktiviert. Mit dieser Funktion können Cluster-Betreiber veränderbare Eigenschaften von Persistent Volumes (PVs) ändern, die von kompatiblen CSI-Treibern verwaltet werden, einschließlich des Amazon EBS CSI-Treibers. Um dieses Feature nutzen zu können, stellen Sie sicher, dass Ihr CSI-Treiber das `VolumeAttributesClass`-Feature unterstützt (für den Amazon-EBS-CSI-Treiber führen Sie ein Upgrade auf Version `1.35.0` oder höher durch, um das Feature automatisch zu aktivieren). Sie können `VolumeAttributesClass` Objekte erstellen, um die gewünschten Volumenattribute wie Volume-Typ und Durchsatz zu definieren und sie Ihren Persistent Volume Claims () PVCs zuzuordnen. Weitere Informationen finden Sie in der offiziellen Kubernetes-Dokumentation sowie in der Dokumentation Ihres CSI-Treibers.
  - Weitere Informationen zum Amazon-EBS-CSI-Treiber finden Sie unter the section called “Amazon EBS”.
- Die Kubernetes-Unterstützung für AppArmorwurde auf Stable umgestellt und ist nun allgemein für die öffentliche Nutzung verfügbar. Mit dieser Funktion können Sie Ihre Container schützen, AppArmor indem Sie das `appArmorProfile.type` Feld in den Containern festlegen. `securityContext` Vor der Kubernetes-Version AppArmor wurde `1.30` es durch Anmerkungen gesteuert. Ab Version `1.30` erfolgt die Steuerung über Felder. Um dieses Feature optimal zu nutzen, empfehlen wir, Annotationen zu migrieren und das `appArmorProfile.type`-Feld zu verwenden, um die Kompatibilität Ihrer Workloads sicherzustellen.
- Die Funktion zur Übergangszeit in der PersistentVolume letzten Phase wurde auf Stable umgestellt und ist nun in der Kubernetes-Version allgemein für den öffentlichen Gebrauch verfügbar. `1.31` Mit dieser Funktion wird ein neues Feld eingeführt,`.status.lastTransitionTime`, in dem ein

Zeitstempel angegeben wird PersistentVolumeStatus, wann das PersistentVolume letzte Mal in eine andere Phase übergegangen ist. Diese Erweiterung ermöglicht eine bessere Nachverfolgung und Verwaltung von PersistentVolumes, insbesondere in Szenarien, in denen es wichtig ist, den Lebenszyklus von Volumen zu verstehen.

Das vollständige `1.31` Kubernetes-Changelog finden Sie unter -1.31.md https://github.com/kubernetes/ kubernetes/blob/master/CHANGELOG/CHANGELOG

Kubernetes 1.30

Kubernetes `1.30` ist jetzt in Amazon EKS verfügbar. Weitere Informationen zu Kubernetes `1.30` finden Sie in der offiziellen Versionsankündigung.

- Ab der Amazon EKS-Version `1.30` oder neuer verwenden alle neu erstellten verwalteten Knotengruppen automatisch standardmäßig Amazon Linux 2023 (AL2023) als Knotenbetriebssystem. Weitere Informationen zum Festlegen des Betriebssystems für eine verwaltete Knotengruppe finden Sie unter the section called “Create”.
- Mit Amazon EKS `1.30` wird das `topology.k8s.aws/zone-id`-Label den Worker-Knoten hinzugefügt. Sie können die Availability Zone IDs (AZ IDs) verwenden, um den Standort von Ressourcen in einem Konto im Verhältnis zu den Ressourcen in einem anderen Konto zu bestimmen. Weitere Informationen finden Sie im AWS RAM-Benutzerhandbuch unter Availability Zone IDs für Ihre AWS Ressourcen.
- Ab Version `1.30` enthält Amazon EKS keine `default`-Annotation mehr zu den `gp2` `StorageClass`-Ressourcen, die auf neu erstellte Cluster angewendet werden. Dies hat keine Auswirkungen, wenn Sie namentlich auf diese Speicherklasse verweisen. Sie müssen Maßnahmen ergreifen, wenn Sie sich darauf verlassen, dass im Cluster ein Standard-`StorageClass` vorhanden ist. Sie sollten auf `StorageClass` mit dem Namen `gp2` verweisen. Alternativ können Sie die von Amazon EBS empfohlene Standardspeicherklasse bereitstellen, indem Sie den `defaultStorageClass.enabled`-Parameter bei der Installation von Version `1.31.0` oder höher von `aws-ebs-csi-driver add-on` auf „true“ setzen.
- Die erforderliche Mindest-IAM-Richtlinie für die IAM-Rolle des Amazon-EKS-Clusters wurde geändert. Die Aktion `ec2:DescribeAvailabilityZones` ist erforderlich. Weitere Informationen finden Sie unter the section called “Cluster-IAM-Rolle”.

Das vollständige `1.30` Kubernetes-Changelog finden Sie unter -1.30.md. https://github.com/kubernetes/ kubernetes/blob/master/CHANGELOG/CHANGELOG

Kubernetes 1.29

Kubernetes `1.29` ist jetzt in Amazon EKS verfügbar. Weitere Informationen zu Kubernetes `1.29` finden Sie in der offiziellen Versionsankündigung.

> **Important**
>
> - Die veralteten `flowcontrol.apiserver.k8s.io/v1beta2`-API-Versionen `FlowSchema` und `PriorityLevelConfiguration` werden in Kubernetes Version `1.29` nicht mehr bereitgestellt. Falls Sie über Manifeste oder Clientsoftware verfügen, welche die veraltete Beta-API-Gruppe verwenden, sollten Sie diese vor der Aktualisierung auf Version `1.29` ändern.

- Das `.status.kubeProxyVersion`-Feld für Knotenobjekte ist nun veraltet, und das Kubernetes-Projekt schlägt vor, dieses Feld in einer zukünftigen Version zu entfernen. Das veraltete Feld ist nicht korrekt und wurde in der Vergangenheit von `kubelet` verwaltet, das weder die `kube-proxy`-Version kennt noch weiß, ob `kube-proxy` ausgeführt wird. Wenn Sie dieses Feld in Ihrer Clientsoftware verwendet haben, sollten Sie dies unterlassen, da die Informationen nicht zuverlässig sind und das Feld nun veraltet ist.
- Um die potenzielle Angriffsfläche in Kubernetes `1.29` zu reduzieren, kennzeichnet das `LegacyServiceAccountTokenCleanUp`-Feature automatisch generierte geheime Tokens als ungültig, wenn sie längere Zeit (standardmäßig 1 Jahr) nicht verwendet wurden, und entfernt sie automatisch, wenn nach der Kennzeichnung als ungültig (standardmäßig 1 weiteres Jahr) längere Zeit keine Verwendung erfolgt. Um solche Tokens zu identifizieren, können Sie folgenden Befehl ausführen:

```
kubectl get cm kube-apiserver-legacy-service-account-token-tracking -n kube-system
```

Das vollständige `1.29` Kubernetes-Changelog finden Sie unter -1.29.md# 1280. https://github.com/ kubernetes/ kubernetes/blob/master/CHANGELOG/CHANGELOG changelog-since-v

Aktuellen Cluster-Supportzeitraum anzeigen

Der Abschnitt Cluster-Supportzeitraum der AWS-Konsole zeigt an, ob Ihr Cluster derzeit Standard- oder erweiterten Support erhält. Wenn es sich bei Ihrem Cluster-Supportzeitraum um Erweiterter Support handelt, wird Ihnen der erweiterte EKS-Support in Rechnung gestellt.

Weitere Informationen zu Standard- und erweitertem Support finden Sie unter Informationen zum Lebenszyklus der Kubernetes-Version in EKS.

1. Navigieren Sie zur Seite Cluster im Abschnitt „EKS“ der AWS-Konsole. Vergewissern Sie sich, dass die AWS-Konsole auf dieselbe Region eingestellt ist wie der Cluster, den Sie überprüfen möchten.
2. Überprüfen Sie die Spalte Supportzeitraum. Wenn der Wert Standard-Support bis … lautet, fallen für Sie derzeit keine Kosten für den erweiterten Support an. Sie befinden sich innerhalb des Standard-Supportzeitraums. Wenn der Wert Erweiterter Support … lautet, fallen für diesen Cluster derzeit Kosten für den erweiterten Support an.

> **Note**
>
> Der Supportzeitraum kann nicht über die AWS-API oder CLI abgerufen werden.

Aktuelle Cluster-Upgrade-Richtlinie anzeigen

Die Cluster-Upgrade-Richtlinie bestimmt, was mit Ihrem Cluster passiert, wenn der Standard-Supportzeitraum endet. Wenn Ihre Upgrade-Richtlinie `EXTENDED` lautet, wird der Cluster nicht automatisch aktualisiert und erhält erweiterten Support. Wenn Ihre Upgrade-Richtlinie `STANDARD` lautet, wird er automatisch aktualisiert.

Mit den Amazon-EKS-Steuerelementen für die Kubernetes-Versionsrichtlinie können Sie das Ende des Standard-Supports für Ihre EKS-Cluster festlegen. Mit diesen Steuerelementen können Sie entscheiden, welche Cluster in den erweiterten Support übergehen und welche Cluster am Ende des Standard-Supports für eine Kubernetes-Version automatisch aktualisiert werden sollen.

Eine Nebenversion wird in den ersten 14 Monaten nach ihrer Veröffentlichung standardmäßig in Amazon EKS unterstützt. Sobald eine Version das Ende des Standard-Supportdatums überschritten hat, wird für die folgenden 12 Monate ein erweiterter Support angeboten. Mit dem erweiterten Support können Sie gegen Aufpreis pro Cluster-Stunde länger bei einer bestimmten Kubernetes-Version bleiben. Sie können den erweiterten Support für einen EKS-Cluster aktivieren oder deaktivieren. Wenn Sie den erweiterten Support deaktivieren, aktualisiert AWS Ihren Cluster am Ende des Standard-Supports automatisch auf die nächste Version. Wenn Sie den erweiterten Support aktivieren, können Sie gegen eine zusätzliche Gebühr für einen begrenzten Zeitraum bei der

aktuellen Version bleiben. Planen Sie regelmäßige Upgrades Ihres Kubernetes-Clusters ein, auch wenn Sie den erweiterten Support nutzen.

Sie können die Versionsrichtlinie für neue und bestehende Cluster mithilfe der `supportType`-Eigenschaft festlegen. Zur Festlegung der Richtlinie für die Versionsunterstützung stehen zwei Optionen zur Verfügung:

- **`STANDARD`** – Ihr EKS-Cluster ist am Ende des Standard-Supports für ein automatisches Upgrade berechtigt. Mit dieser Einstellung fallen keine zusätzlichen Supportkosten an, jedoch wird Ihr EKS-Cluster automatisch auf die nächste unterstützte Kubernetes-Version im Standard-Support aktualisiert.
- **`EXTENDED`** – Ihr EKS-Cluster wird in den erweiterten Support übergehen, sobald die Kubernetes-Version das Ende des Standard-Supports erreicht. Bei dieser Einstellung fallen zusätzliche Support-Gebühren an. Sie können Ihren Cluster auf eine standardmäßig unterstützte Kubernetes-Version aktualisieren, um die erweiterten Support-Gebühren zu vermeiden. Cluster, die mit erweitertem Support ausgeführt werden, können am Ende des erweiterten Supports automatisch aktualisiert werden.

Der erweiterte Support ist standardmäßig für neue Cluster und vorhandene Cluster aktiviert. Sie können überprüfen, ob der erweiterte Support für ein Cluster aktiviert ist, indem Sie die AWS-Managementkonsole aufrufen oder die AWS CLI verwenden.

> **Important**
>
> Wenn Sie möchten, dass Ihr Cluster die aktuelle Kubernetes-Version beibehält, um die Vorteile des erweiterten Supportzeitraums zu nutzen, müssen Sie die Upgrade-Richtlinie für den erweiterten Support vor dem Ende des Standard-Supportzeitraums aktivieren.

Sie können die Richtlinie zur Versionsunterstützung für Ihre Cluster nur festlegen, wenn diese unter Kubernetes-Version im Standard-Support ausgeführt werden. Sie diese Einstellung nicht mehr ändern, bis Sie wieder unter einer Version im Standard-Support ausgeführt werden.

Wenn Sie beispielsweise Ihre Richtlinie zur Versionsunterstützung auf `standard` festgelegt haben, können Sie diese Einstellung nicht mehr ändern, nachdem die auf Ihrem Cluster ausgeführte Kubernetes-Version das Ende des Standard-Supports erreicht hat. Wenn Sie Ihre Richtlinie zur Versionsunterstützung auf `extended` festgelegt haben, können Sie diese Einstellung nicht mehr ändern, nachdem die auf Ihrem Cluster ausgeführte Kubernetes-Version das Ende des Standard-

Supports erreicht hat. Um die Einstellung für die Richtlinie zur Versionsunterstützung zu ändern, muss Ihr Cluster in einer standardmäßig unterstützten Kubernetes-Version ausgeführt werden.

Cluster-Upgrade-Richtlinie anzeigen (AWS-Konsole)

1. Navigieren Sie zur Seite Cluster im Abschnitt „EKS“ der AWS-Konsole. Vergewissern Sie sich, dass die AWS-Konsole auf dieselbe Region eingestellt ist wie der Cluster, den Sie überprüfen möchten.
2. Überprüfen Sie die Spalte Upgrade-Richtlinie. Wenn der Wert Standard-Support lautet, wird Ihr Cluster nicht in den erweiterten Support aufgenommen. Wenn der Wert Erweiterter Support lautet, wird Ihr Cluster in den erweiterten Support übernommen.

Cluster-Upgrade-Richtlinie anzeigen (AWS-CLI)

1. Überprüfen Sie, ob die AWS-CLI installiert ist und Sie angemeldet sind. Erfahren Sie, wie Sie die AWS-CLI aktualisieren und installieren.
2. Bestimmen Sie den Namen Ihres EKS-Clusters. Legen Sie die CLI auf dieselbe AWS-Region wie Ihren EKS-Cluster fest.
3. Führen Sie den folgenden Befehl aus:

```
aws eks describe-cluster \
--name <cluster-name> \
--query "cluster.upgradePolicy.supportType"
```

4. Wenn der Wert `STANDARD` lautet, wird Ihr Cluster nicht in den erweiterten Support aufgenommen. Wenn der Wert `EXTENDED` lautet, wird Ihr Cluster in den erweiterten Support versetzt.

Erhöhen der Flexibilität bei der Planung von Kubernetes-Versions-Upgrades durch die Aktivierung der erweiterten EKS-Unterstützung

In diesem Thema wird beschrieben, wie Sie die Upgrade-Richtlinie eines EKS-Clusters festlegen, um erweiterten Support zu aktivieren. Die Upgrade-Richtlinie eines EKS-Clusters bestimmt, was geschieht, wenn ein Cluster das Ende der Standard-Supportzeitraum erreicht. Wenn für eine Cluster-Upgrade-Richtlinie erweiterter Support aktiviert ist, beginnt die erweiterte Support-Phase nach Ablauf der Standard-Support-Phase. Der Cluster wird nach Ablauf der Standard-Support-Phase nicht automatisch aktualisiert.

Cluster, die sich tatsächlich in der erweiterten Support-Phase befinden, verursachen höhere Kosten. Wenn für einen Cluster lediglich die Upgrade-Richtlinie zur Aktivierung des erweiterten Supports festgelegt ist und er sich ansonsten im Standard-Support-Phase befindet, fallen Standardkosten an.

Wenn Sie einen Cluster in der AWS-Konsole erstellen, ist die Upgrade-Richtlinie für diesen Cluster so festgelegt, dass der erweiterte Support deaktiviert wird. Wenn Sie einen Cluster auf andere Weise erstellen, ist die Upgrade-Richtlinie so festgelegt, dass der erweiterte Support aktiviert ist. Beispielsweise ist für mit der AWS API erstellte Cluster der erweiterte Support aktiviert.

Weitere Informationen zu Upgrade-Richtlinien finden Sie unter Cluster-Upgrade-Richtlinie.

> **Important**
>
> Wenn Sie möchten, dass Ihr Cluster die aktuelle Kubernetes-Version beibehält, um die Vorteile des erweiterten Supportzeitraums zu nutzen, müssen Sie die Upgrade-Richtlinie für den erweiterten Support vor dem Ende des Standard-Supportzeitraums aktivieren.
> Wenn Sie den erweiterten Support nicht aktivieren, wird Ihr Cluster automatisch aktualisiert.

Erweiterten Support für EKS aktivieren (AWS-Konsole)

1. Navigieren Sie in der AWS-Konsole zu Ihrem EKS-Cluster. Wählen Sie auf der Seite Cluster-Informationen die Registerkarte Übersicht aus.
2. Wählen Sie im Abschnitt Kubernetes-Versionseinstellungen die Option Verwalten aus.
3. Wählen Sie Erweiterter Support und dann Änderungen speichern aus.

Erweiterten Support für EKS aktivieren (AWS-CLI)

1. Überprüfen Sie, ob die AWS-CLI installiert ist und Sie angemeldet sind. Erfahren Sie, wie Sie die AWS-CLI aktualisieren und installieren.
2. Bestimmen Sie den Namen Ihres EKS-Clusters.
3. Führen Sie den folgenden Befehl aus:

```
aws eks update-cluster-config \
--name <cluster-name> \
--upgrade-policy supportType=EXTENDED
```

Verhindern Sie erhöhte Cluster-Kosten, indem Sie den erweiterten Support für EKS deaktivieren.

In diesem Thema wird beschrieben, wie Sie die Upgrade-Richtlinie eines EKS-Clusters festlegen, um den erweiterten Support zu deaktivieren. Die Upgrade-Richtlinie eines EKS-Clusters bestimmt, was geschieht, wenn ein Cluster das Ende der Standard-Supportzeitraum erreicht. Wenn für eine Cluster-Upgrade-Richtlinie der erweiterte Support deaktiviert ist, wird automatisch ein Upgrade auf die nächste Kubernetes-Version durchgeführt.

Weitere Informationen zu Upgrade-Richtlinien finden Sie unter Cluster-Upgrade-Richtlinie.

> **Important**
>
> Sobald Ihr Cluster in den erweiterten Support eingetreten ist, können Sie diesen nicht mehr deaktivieren. Sie können den erweiterten Support nur für Cluster deaktivieren, die sich im Standard-Support befinden.
> AWS empfiehlt, Ihren Cluster auf eine Version innerhalb des Standard-Supportzeitraums zu aktualisieren.

Erweiterten EKS-Support deaktivieren (AWS-Konsole)

1. Navigieren Sie in der AWS-Konsole zu Ihrem EKS-Cluster. Wählen Sie auf der Seite Cluster-Informationen die Registerkarte Übersicht aus.
2. Wählen Sie im Abschnitt Kubernetes-Versionseinstellungen die Option Verwalten.
3. Wählen Sie Standard-Support und anschließend Änderungen speichern.

Erweiterten Support für EKS deaktivieren (AWS-CLI)

1. Überprüfen Sie, ob die AWS-CLI installiert ist und Sie angemeldet sind. Erfahren Sie, wie Sie die AWS-CLI aktualisieren und installieren.
2. Bestimmen Sie den Namen Ihres EKS-Clusters.
3. Führen Sie den folgenden Befehl aus:

```
aws eks update-cluster-config \
--name <cluster-name> \
--upgrade-policy supportType=STANDARD
```

Anzeigen der Amazon-EKS-Plattformversionen für jede Kubernetes-Version

Amazon-EKS-Plattformversionen definieren die Funktionalitäten der Amazon-EKS-Cluster-Steuerebene (z. B. welche Kubernetes-API-Server-Flags aktiviert sind) sowie die aktuelle Kubernetes-Patch-Version. Jede Kubernetes-Minor-Version hat eine oder mehrere zugehörige Amazon-EKS-Plattformversionen. Die Plattformversionen für verschiedene Kubernetes-Nebenversionen sind unabhängig. Sie können die aktuelle Plattformversion Ihres Clusters mit der AWS CLI oder abrufen AWS-Managementkonsole. Wenn Sie einen lokalen Cluster auf AWS Outposts haben, lesen Sie the section called “EKS-Plattformversionen” statt dieses Themas weiter.

Wenn eine neue Kubernetes-Nebenversion in Amazon EKS verfügbar ist, z. B. 1.33, beginnt die erste Amazon EKS-Plattformversion für diese Kubernetes-Nebenversion bei. `eks.1` Allerdings veröffentlicht Amazon EKS regelmäßig neue Plattformversionen, um neue Einstellungen für die Kubernetes-Steuerebene und Sicherheitsfixes bereitzustellen.

Wenn neue Amazon-EKS-Plattformversionen für eine Minor-Version verfügbar werden:

- Die Versionsnummer der Amazon-EKS-Plattform wird erhöht (`eks.<n+1>`).
- Amazon EKS aktualisiert automatisch alle bestehenden Cluster auf die neueste Amazon-EKS-Plattformversion für die entsprechende Kubernetes-Minor-Version. Automatische Upgrades bestehender Amazon-EKS-Plattformversionen werden schrittweise durchgeführt. Der Rollout-Prozess kann einige Zeit in Anspruch nehmen. Wenn Sie die neuesten Funktionen der Amazon-EKS-Plattformversion sofort benötigen, sollten Sie einen neuen Amazon-EKS-Cluster erstellen.

  Wenn Ihr Cluster mehr als zwei Plattformversionen hinter der aktuellen Plattformversion zurückliegt, ist es möglich, dass Amazon EKS Ihren Cluster nicht automatisch aktualisieren konnte. Einzelheiten dazu, was dies verursachen kann, finden Sie unter the section called “Die Amazon-EKS-Plattformversion liegt mehr als zwei Versionen hinter der aktuellen Plattformversion”.
- Amazon EKS veröffentlicht möglicherweise ein neues Knoten-AMI mit einer entsprechenden Patch-Version. Alle Patch-Versionen sind jedoch zwischen der EKS-Steuerebene und dem Knoten AMIs für eine bestimmte Kubernetes-Nebenversion kompatibel.

Neue Amazon-EKS-Plattformversionen führen keine kritischen Änderungen ein. Sie führen nicht zu Service-Unterbrechungen.

Cluster werden immer mit der neuesten verfügbaren Amazon-EKS-Plattformversion (`eks.<n>`) für die angegebene Kubernetes-Version erstellt. Wenn Sie Ihren Cluster auf eine neue Kubernetes-Minor-Version aktualisieren, erhält Ihr Cluster die aktuelle Amazon-EKS-Plattformversion für die Kubernetes-Minor-Version, auf die Sie aktualisiert haben.

Die aktuellen und kürzlichen Amazon-EKS-Plattformversionen sind in den folgenden Tabellen beschrieben.

> **Note**
>
> AWS hat kürzlich einige Plattformversionen deaktiviert, die im Juni 2024 veröffentlicht wurden. Die Plattformversionen hatten Stabilitätsprobleme. Keine Aktion erforderlich.

Um Benachrichtigungen über alle Änderungen an den Quelldateien dieser spezifischen Dokumentationsseite zu erhalten, können Sie die folgende URL mit einem RSS-Reader abonnieren:

```
https://github.com/awsdocs/amazon-eks-user-guide/commits/mainline/latest/ug/versioning/platform-versions.adoc.atom
```

Kubernetes-Version **`1.34`**

Die folgenden Zugangssteuerungen sind für alle aktiviert `1.34`-Plattformversionen: `NodeRestriction`, `ExtendedResourceToleration`, `NamespaceLifecycle`, `LimitRanger`, `ServiceAccount`, `TaintNodesByCondition`, `PodSecurity`, `Priority`, `DefaultTolerationSeconds`, `DefaultStorageClass`, `StorageObjectInUseProtection`, `PersistentVolumeClaimResize`, `RuntimeClass`, `CertificateApproval`, `CertificateSigning`, `CertificateSubjectRestriction`, `DefaultIngressClass`, `MutatingAdmissionWebhook`, `ValidatingAdmissionWebhook`, `ResourceQuota`.

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
`1.34.2`	`eks.10`	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	19. Dezember 2025

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.34.1	eks.9	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	18. November 2025
1.34.1	eks.8	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	17. November 2025
1.34.1	eks.7	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	6. November 2025
1.34.1	eks.6	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	30. Oktober 2025
1.34.1	eks.5	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	15. Oktober 2025
1.34.1	eks.4	Erstversion von der Kubernetes-Version 1.34 für EKS. Weitere Informationen finden Sie unter the section called “Kubernetes 1.34”.	2. Oktober 2025

Kubernetes-Version **1.33**

Die folgenden Zugangssteuerungen sind für alle aktiviert 1.33-Plattformversionen: NodeRestriction, ExtendedResourceToleration, NamespaceLifecycle, LimitRanger, ServiceAccount, TaintNodesByCondition, PodSecurity, Priority, DefaultTolerationSeconds, DefaultStorageClass, StorageObjectInUseProtection,

PersistentVolumeClaimResize, RuntimeClass, CertificateApproval, CertificateSigning, CertificateSubjectRestriction, DefaultIngressClass, MutatingAdmissionWebhook, ValidatingAdmissionWebhook, ResourceQuota.

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.33.6	eks.24	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	19. Dezember 2025
1.33.5	eks.23	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	18. November 2025
1.33.5	eks.22	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	17. November 2025
1.33.5	eks.21	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	6. November 2025
1.33.5	eks.20	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	30. Oktober 2025
1.33.5	eks.19	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	28. Oktober 2025
1.33.5	eks.18	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	15. Oktober 2025
1.33.5	eks.17	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	15. Oktober 2025

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.33.5	eks.16	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	2. Oktober 2025
1.33.4	eks.15	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	29. September 2025
1.33.4	eks.14	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	29. September 2025
1.33.4	eks.13	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	16. September 2025
1.33.4	eks.12	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	5. September 2025
1.33.3	eks.11	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	25. August 2025
1.33.3	eks.10	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	17. August 2025

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.33.2	eks.9	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. Die Versionen 1.33 eks.7 und 1.33 eks.8 wurden intern verworfen und nie veröffentlicht.	30. Juli 2025
1.33.1	eks.6	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	26. Juni 2025
1.33.1	eks.5	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	11. Juni 2025
1.33.1	eks.4	Erstversion von der Kubernetes-Version 1.33 für EKS. Weitere Informationen finden Sie unter the section called “Kubernetes 1.33”.	28. Mai 2025

Kubernetes-Version **1.32**

Die folgenden Zugangssteuerungen sind für alle aktiviert 1.32-Plattformversionen: NodeRestriction, ExtendedResourceToleration, NamespaceLifecycle, LimitRanger, ServiceAccount, TaintNodesByCondition, PodSecurity, Priority, DefaultTolerationSeconds, DefaultStorageClass, StorageObjectInUseProtection, PersistentVolumeClaimResize, RuntimeClass, CertificateApproval, CertificateSigning, CertificateSubjectRestriction, DefaultIngressClass, MutatingAdmissionWebhook, ValidatingAdmissionWebhook, ResourceQuota.

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.32.10	eks.31	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	19. Dezember 2025
1.32.9	eks.30	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	18. November 2025
1.32.9	eks.29	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	17. November 2025
1.32.9	eks.28	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	6. November 2025
1.32.9	eks.27	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	30. Oktober 2025
1.32.9	eks.26	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	28. Oktober 2025
1.32.9	eks.25	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	15. Oktober 2025
1.32.9	eks.24	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	15. Oktober 2025
1.32.9	eks.23	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	2. Oktober 2025

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.32.8	eks.22	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	29. September 2025
1.32.8	eks.21	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	29. September 2025
1.32.8	eks.20	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	16. September 2025
1.32.8	eks.19	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	5. September 2025
1.32.7	eks.18	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	25. August 2025
1.32.7	eks.17	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	17. August 2025
1.32.6	eks.16	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. Die Versionen 1.32 eks.14 und 1.32 eks.15 wurden intern verworfen und nie veröffentlicht.	30. Juli 2025

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.32.5	eks.13	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	26. Juni 2025
1.32.5	eks.12	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	11. Juni 2025
1.32.5	eks.11	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	30. Mai 2025
1.32.3	eks.10	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	16. Mai 2025
1.32.3	eks.9	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	29. April 2025
1.32.3	eks.8	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	18. April 2025
1.32.3	eks.7	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	18. April 2025
1.32.3	eks.6	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	2. April 2025
1.32.2	eks.5	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	17. März 2025

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.32.2	eks.4	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	4. März 2025
1.32.1	eks.3	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	24. Februar 2025
1.32.0	eks.2	Erstversion von der Kubernetes-Version 1.32 für EKS. Weitere Informationen finden Sie unter the section called “Kubernetes 1.32”.	Januar 2025

Kubernetes-Version **1.31**

Die folgenden Zugangssteuerungen sind für alle 1.31-Plattformversionen aktiviert: NodeRestriction, ExtendedResourceToleration, NamespaceLifecycle, LimitRanger, ServiceAccount, TaintNodesByCondition, PodSecurity, Priority, DefaultTolerationSeconds, DefaultStorageClass, StorageObjectInUseProtection, PersistentVolumeClaimResize, RuntimeClass, CertificateApproval, CertificateSigning, CertificateSubjectRestriction, DefaultIngressClass, MutatingAdmissionWebhook, ValidatingAdmissionWebhook, ResourceQuota, ObjectCount.

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.31.14	eks.47	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	19. Dezember 2025

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.31.13	eks.46	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	18. November 2025
1.31.13	eks.45	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	17. November 2025
1.31.13	eks.44	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	6. November 2025
1.31.13	eks.43	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	30. Oktober 2025
1.31.13	eks.42	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	28. Oktober 2025
1.31.13	eks.41	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	15. Oktober 2025
1.31.13	eks.40	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	15. Oktober 2025
1.31.13	eks.39	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	2. Oktober 2025
1.31.12	eks.38	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	29. September 2025

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.31.12	eks.37	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	29. September 2025
1.31.12	eks.36	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	16. September 2025
1.31.12	eks.35	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	5. September 2025
1.31.11	eks.34	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	25. August 2025
1.31.11	eks.33	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	17. August 2025
1.31.10	eks.32	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. Die Versionen 1.31 eks.30 und 1.31 eks.31 wurden intern verworfen und nie veröffentlicht.	30. Juli 2025
1.31.9	eks.29	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	26. Juni 2025

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.31.9	eks.28	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	11. Juni 2025
1.31.9	eks.27	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	30. Mai 2025
1.31.7	eks.26	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	16. Mai 2025
1.31.7	eks.25	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	29. April 2025
1.31.7	eks.24	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	18. April 2025
1.31.7	eks.23	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	18. April 2025
1.31.7	eks.22	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	2. April 2025
1.31.6	eks.21	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	17. März 2025
1.31.6	eks.20	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	4. März 2025

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.31.5	eks.19	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	24. Februar 2025
1.31.5	eks.18	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	24. Februar 2025
1.31.4	eks.17	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	17. Januar 2025
1.31.2	eks.12	Neue Plattformversion mit Unterstützung für Amazon EKS Hybrid Nodes und Verbesserungen der Beobachtbarkeit der Steuerebene. Lesen Sie the section called “Hybridknoten” und Amazon EKS verbessert die Beobachtbarkeit der Leistung.	15. November 2024
1.31.1	eks.6	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	21. Oktober 2024

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.31.0	eks.4	Erstversion von der Kubernetes-Version 1.31 für EKS. Weitere Informationen finden Sie unter the section called “Kubernetes 1.31”.	26. September 2024

Kubernetes-Version **1.30**

Die folgenden Zugangssteuerungen sind für alle aktiviert 1.30-Plattformversionen: NodeRestriction, ExtendedResourceToleration, NamespaceLifecycle, LimitRanger, ServiceAccount, TaintNodesByCondition, PodSecurity, Priority, DefaultTolerationSeconds, DefaultStorageClass, StorageObjectInUseProtection, PersistentVolumeClaimResize, RuntimeClass, CertificateApproval, CertificateSigning, CertificateSubjectRestriction, DefaultIngressClass, MutatingAdmissionWebhook, ValidatingAdmissionWebhook, ResourceQuota.

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.30.14	eks.55	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	19. Dezember 2025
1.30.14	eks.54	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	18. November 2025
1.30.14	eks.53	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	17. November 2025

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.30.14	eks.52	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	6. November 2025
1.30.14	eks.51	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	30. Oktober 2025
1.30.14	eks.50	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	28. Oktober 2025
1.30.14	eks.49	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	15. Oktober 2025
1.30.14	eks.48	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	15. Oktober 2025
1.30.14	eks.47	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	2. Oktober 2025
1.30.14	eks.46	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	29. September 2025
1.30.14	eks.45	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	29. September 2025
1.30.14	eks.44	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	16. September 2025

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.30.14	eks.43	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	5. September 2025
1.30.14	eks.42	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	25. August 2025
1.30.14	eks.41	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	17. August 2025
1.30.14	eks.40	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. Die Versionen 1.30 eks.38 und 1.30 eks.39 wurden intern verworfen und nie veröffentlicht.	30. Juli 2025
1.30.13	eks.37	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	26. Juni 2025
1.30.13	eks.36	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	11. Juni 2025
1.30.13	eks.35	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	30. Mai 2025

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.30.11	eks.34	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	16. Mai 2025
1.30.11	eks.33	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	29. April 2025
1.30.11	eks.32	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	18. April 2025
1.30.11	eks.31	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	18. April 2025
1.30.11	eks.30	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	2. April 2025
1.30.10	eks.29	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	17. März 2025
1.30.10	eks.28	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	4. März 2025
1.30.9	eks.27	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	24. Februar 2025
1.30.8	eks.25	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	17. Januar 2025

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.30.8	eks.24	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	3. Januar 2025
1.30.7	eks.23	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	13. Dezember 2024
1.30.6	eks.22	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	13. Dezember 2024
1.30.6	eks.21	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	13. Dezember 2024
1.30.6	eks.20	Neue Plattformversion mit Unterstützung für Amazon EKS Hybrid Nodes und Verbesserungen der Beobachtbarkeit der Steuerebene. Lesen Sie the section called “Hybridknoten” und Amazon EKS verbessert die Beobachtbarkeit der Leistung.	15. November 2024
1.30.5	eks.12	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	21. Oktober 2024

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.30.4	eks.8	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	3. September 2024
1.30.3	eks.7	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	28. August 2024
1.30.3	eks.6	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	9. August 2024
1.30.2	eks.5	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	2. Juli 2024
1.30.0	eks.2	Erstversion von der Kubernetes-Version 1.30 für EKS. Weitere Informationen finden Sie unter the section called “Kubernetes 1.30”.	23. Mai 2024

Kubernetes-Version **1.29**

Die folgenden Zugangssteuerungen sind für alle aktiviert 1.29-Plattformversionen: NodeRestriction, ExtendedResourceToleration, NamespaceLifecycle, LimitRanger, ServiceAccount, TaintNodesByCondition, PodSecurity, Priority, DefaultTolerationSeconds, DefaultStorageClass, StorageObjectInUseProtection, PersistentVolumeClaimResize, RuntimeClass, CertificateApproval, CertificateSigning, CertificateSubjectRestriction, DefaultIngressClass, MutatingAdmissionWebhook, ValidatingAdmissionWebhook, ResourceQuota.

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.29.15	eks.58	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	19. Dezember 2025
1.29.15	eks.57	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	18. November 2025
1.29.15	eks.56	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	17. November 2025
1.29.15	eks.55	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	6. November 2025
1.29.15	eks.54	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	30. Oktober 2025
1.29.15	eks.53	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	28. Oktober 2025
1.29.15	eks.52	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	15. Oktober 2025
1.29.15	eks.51	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	15. Oktober 2025
1.29.15	eks.50	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	2. Oktober 2025

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.29.15	eks.49	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	29. September 2025
1.29.15	eks.48	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	29. September 2025
1.29.15	eks.47	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	16. September 2025
1.29.15	eks.46	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	5. September 2025
1.29.15	eks.45	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	25. August 2025
1.29.15	eks.44	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	17. August 2025
1.29.15	eks.43	Neue Plattformversion mit Sicherheitskorrekturen und Verbesserungen. Die Versionen 1.29 eks.41 und 1.29 eks.42 wurden intern verworfen und nie veröffentlicht.	30. Juli 2025

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.29.15	eks.40	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	26. Juni 2025
1.29.15	eks.39	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	11. Juni 2025
1.29.15	eks.38	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	30. Mai 2025
1.29.15	eks.37	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	16. Mai 2025
1.29.15	eks.36	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	29. April 2025
1.29.15	eks.35	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	18. April 2025
1.29.15	eks.34	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	18. April 2025
1.29.15	eks.33	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	2. April 2025
1.29.14	eks.32	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	17. März 2025

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.29.14	eks.31	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	4. März 2025
1.29.13	eks.30	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	24. Februar 2025
1.29.13	eks.29	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	24. Februar 2025
1.29.12	eks.28	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	20. Januar 2025
1.29.12	eks.27	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	3. Januar 2025
1.29.11	eks.26	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	13. Dezember 2024
1.29.10	eks.25	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	13. Dezember 2024
1.29.10	eks.24	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	13. Dezember 2024

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.29.10	eks.23	Neue Plattformversion mit Unterstützung für Amazon EKS Hybrid Nodes und Verbesserungen der Beobachtbarkeit der Steuerebene. Lesen Sie the section called “Hybridknoten” und Amazon EKS verbessert die Beobachtbarkeit der Leistung.	15. November 2024
1.29.9	eks.17	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	21. Oktober 2024
1.29.8	eks.13	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	3. September 2024
1.29.7	eks.12	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	28. August 2024
1.29.7	eks.11	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	9. August 2024
1.29.6	eks.10	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	2. Juli 2024

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.29.4	eks.7	Neue Plattformversion mit CoreDNS-Autoscaling, Sicherheitskorrekturen und Verbesserungen. Weitere Informationen zum CoreDNS-Autoscaling finden Sie unter the section called “Skalierbarkeit für hohen Datenverkehr”.	16. Mai 2024
1.29.3	eks.6	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	18. April 2024
1.29.1	eks.5	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	29. März 2024
1.29.1	eks.4	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	20. März 2024
1.29.1	eks.3	Neue Plattformversion mit Sicherheitsfixes und -verbesserungen.	12. März 2024

Kubernetes-Version	EKS-Plattformversionen	Versionshinweise	Datum der Veröffentlichung
1.29.0	eks.1	Erstversion von der Kubernetes-Version 1.29 für EKS. Weitere Informationen finden Sie unter the section called “Kubernetes 1.29”.	23. Januar 2024

Aktuelle Plattformversion abrufen

1. Öffnen Sie die Amazon-EKS-Konsole.
2. Klicken Sie im Navigationsbereich auf Cluster.
3. Wählen Sie in der Liste der Cluster den Cluster-Namen aus, dessen Plattformversion Sie überprüfen möchten.
4. Wählen Sie die Übersicht-Registerkarte.
5. Die Plattformversion finden Sie im Abschnitt Details.
6. Bestimmen Sie den Namen des Clusters, dessen Plattformversion Sie überprüfen möchten.
7. Führen Sie den folgenden Befehl aus:

```
aws eks describe-cluster --name my-cluster --query cluster.platformVersion
```

Eine Beispielausgabe sieht wie folgt aus.

```
"eks.10"
```

Plattformversion ändern

Sie können die Plattformversion eines EKS-Clusters nicht ändern. Wenn neue Versionen der Amazon-EKS-Plattform für eine Kubernetes-Version verfügbar werden, aktualisiert EKS automatisch alle vorhandenen Cluster auf die neueste Version der Amazon-EKS-Plattform für die entsprechende Kubernetes-Version. Automatische Upgrades bestehender Amazon-EKS-Plattformversionen

werden schrittweise durchgeführt. Sie können die AWS Konsole oder CLI nicht verwenden, um die Plattformversion zu ändern.

Wenn Sie Ihre Kubernetes-Version aktualisieren, wird Ihr Cluster auf die aktuellste Plattformversion für die Kubernetes-Version umgestellt.

Beheben von Problemen mit Amazon-EKS-Clustern und -Knoten

In diesem Kapitel werden einige häufige Fehler bei der Verwendung von Amazon EKS sowie deren Lösung behandelt. Wenn Sie Probleme in bestimmten Amazon-EKS-Bereichen beheben müssen, lesen Sie die separaten Themen the section called “Fehlerbehebung”, the section called “Fehlerbehebung für EKS Connector” und Fehlerbehebung für ADOT mithilfe von EKS-Add-ons.

Weitere Informationen zur Fehlerbehebung finden Sie in den Wissenszentrum-Inhalten zu Amazon Elastic Kubernetes Service in AWS re:Post.

Unzureichende Kapazität

Wenn beim Versuch, einen Amazon-EKS-Cluster zu erstellen, der folgende Fehler auftritt, verfügt eine der angegebenen Availability Zones nicht über ausreichende Kapazität, um einen Cluster zu unterstützen.

```
Cannot create cluster 'example-cluster' because region-1d, the targeted Availability Zone, does not currently have sufficient capacity to support the cluster. Retry and choose from these Availability Zones: region-1a, region-1b, region-1c
```

Versuchen Sie erneut, den Cluster mit Subnetzen in Ihrer Cluster-VPC zu erstellen, die in den Availability Zones gehostet werden, die diesen Fehler verursacht haben.

Es gibt Availability Zones, in denen sich kein Cluster befinden darf. Vergleichen Sie die Availability Zones, in denen sich Ihre Subnetze befinden, mit der Liste der Availability Zones in den Anforderungen und Überlegungen zu Subnetzen.

Knoten können nicht mit dem Cluster verknüpft werden

Es gibt zwei häufige Gründe, aus denen Knoten nicht mit dem Cluster verknüpft werden können:

- Wenn es sich bei den Knoten um verwaltete Knoten handelt, fügt Amazon EKS Einträge zu `aws-auth ConfigMap` hinzu, wenn Sie die Knotengruppe erstellen. Wenn der Eintrag entfernt oder geändert wurde, müssen Sie ihn erneut hinzufügen. Weitere Informationen erhalten Sie durch

Eingabe von `eksctl create iamidentitymapping --help` im Terminal. Sie können Ihre aktuellen `aws-auth ConfigMap`-Einträge anzeigen, indem Sie im folgenden Befehl *my-cluster* durch den Namen Ihres Clusters ersetzen und dann den geänderten Befehl ausführen: `eksctl get iamidentitymapping --cluster` *my-cluster* . Der ARN der von Ihnen angegebenen Rolle darf als Pfad nur `/` enthalten. Wenn der Name Ihrer Rolle also beispielsweise `development/apps/my-role` lautet, müssen Sie ihn bei Angabe des ARN für die Rolle in `my-role` ändern. Achten Sie darauf, den ARN der IAM-Rolle des Knotens (nicht den ARN des Instances-Profils) anzugeben.

Wenn es sich um selbstverwaltete Knoten handelt und Sie keine Zugriffseinträge für den ARN der IAM-Rolle des Knotens erstellt haben, führen Sie die gleichen Befehle aus, die für verwaltete Knoten aufgeführt sind. Wenn Sie einen Zugriffseintrag für den ARN für die IAM-Rolle Ihres Knotens erstellt haben, ist er im Zugriffseintrag möglicherweise nicht richtig konfiguriert. Achten Sie darauf, dass in Ihrem `aws-auth ConfigMap`-Eintrag oder Zugriffseintrag der ARN der IAM-Rolle des Knotens (nicht der ARN des Instances-Profils) als Haupt-ARN angegeben ist. Weitere Informationen zu Zugriffseinträgen finden Sie unter the section called “Zugriffseinträge”.

- Der ClusterName in der AWS-CloudFormation-Vorlage stimmt nicht genau mit dem Namen des Clusters überein, dem Ihre Knoten beitreten sollen. Wenn Sie diesem Feld einen falschen Wert übergeben, führt dies zu einer falschen Konfiguration der `/var/lib/kubelet/kubeconfig`-Datei des Knotens und die Knoten werden dem Cluster nicht beitreten.
- Der Knoten ist nicht als Eigentum des Clusters gekennzeichnet. Auf Ihre Knoten muss das folgende Tag angewendet werden, wobei *my-cluster* durch den Namen Ihres Clusters ersetzt wird.

Schlüssel	Wert
`kubernetes.io/cluster/` *my-cluster*	`owned`

- Die Knoten können möglicherweise nicht über eine öffentliche IP-Adresse auf den Cluster zugreifen. Stellen Sie sicher, dass den Knoten, die in öffentlichen Subnetzen bereitgestellt werden, eine öffentliche IP-Adresse zugewiesen wird. Ist dies nicht der Fall, können Sie einem Knoten eine Elastic-IP-Adresse nach dem Start zuordnen. Weitere Informationen dazu finden Sie unter Zuordnen einer Elastic-IP-Adresse zu einer laufenden Instance oder einer Netzwerkschnittstelle. Wenn das öffentliche Subnetz nicht so eingestellt ist, dass es Instances, die es nutzen, automatisch öffentliche IP-Adressen zuweist, empfehlen wir, diese Einstellung zu aktivieren. Weitere Informationen finden Sie unter Ändern des öffentlichen IPv4-Adressierungsattributs für Ihr

Subnetz. Wenn der Arbeitsknoten in einem privaten Subnetz bereitgestellt wird, muss das Subnetz über eine Route zu einem NAT-Gateway verfügen, dem eine öffentliche IP-Adresse zugewiesen ist.

- Der AWS-STS-Endpunkt für die AWS-Region, für die Sie die Worker-Knoten bereitstellen, ist für Ihr Konto nicht aktiviert. Informationen zum Aktivieren der Region finden Sie unter Aktivieren und Deaktivieren von AWS STS in einer AWS-Region.
- Der Knoten hat keinen privaten DNS-Eintrag, was dazu führt, dass das `kubelet`-Protokoll einen `node "" not found`-Fehler enthält. Stellen Sie sicher, dass für die VPC, in der der Knoten erstellt wird, Werte für `domain-name` und `domain-name-servers` als `Options` in einem `DHCP options set` festgelegt sind. Die Standard-Werte sind `domain-name:<region>.compute.internal` und `domain-name-servers:AmazonProvidedDNS`. Weitere Informationen finden Sie unter DHCP-Optionssätze im Amazon-VPC-Benutzerhandbuch.
- Wenn die Knoten in der verwalteten Knotengruppe nicht innerhalb von 15 Minuten eine Verbindung zum Cluster herstellen, wird ein Integritätsproblem „NodeCreationFailure“ ausgegeben und der Konsolen-Status auf `Create failed` gesetzt. Bei Windows-AMIs mit langsamen Startzeiten kann dieses Problem durch den Schnellstart behoben werden.

Zur Ermittlung und Behebung häufiger Ursachen, die den Beitritt von Worker-Knoten zu einem Cluster verhindern, können Sie das Runbook `AWSSupport-TroubleshootEKSWorkerNode` verwenden. Weitere Informationen finden Sie unter `AWSSupport-TroubleshootEKSWorkerNode` in der AWS-Runbook-Referenz zu Systems Manager Automation.

Nicht autorisiert oder Zugriff verweigert (**`kubectl`**)

Wenn Sie beim Ausführen von `kubectl`-Befehlen einen der folgenden Fehler erhalten, ist `kubectl` für Amazon EKS nicht richtig konfiguriert oder die Anmeldeinformationen für den von Ihnen verwendeten IAM-Prinzipal (Rolle oder Benutzer) lassen sich keinem Kubernetes-Benutzernamen zuordnen, der über ausreichende Berechtigungen für Kubernetes-Objekte in Ihrem Amazon-EKS-Cluster verfügt.

- `could not get token: AccessDenied: Access denied`
- `error: You must be logged in to the server (Unauthorized)`
- `error: the server doesn’t have a resource type "svc"`

Das kann auf eine der folgenden Ursachen zurückzuführen sein:

- Der Cluster wurde mit Anmeldeinformationen für einen bestimmten IAM-Prinzipal erstellt und `kubectl` ist für die Verwendung von Anmeldeinformationen für einen anderen IAM-Prinzipal konfiguriert. Aktualisieren Sie in diesem Fall die Datei `kube config` mit den Anmeldeinformationen, mit denen der Cluster erstellt wurde, um das Problem zu beheben. Weitere Informationen finden Sie unter the section called “Zugriff auf den Cluster mit kubectl”.
- Wenn Ihr Cluster die Mindestplattformanforderungen im Abschnitt „Voraussetzungen“ unter IAM-Benutzern Zugriff auf Kubernetes mit EKS-Zugriffseinträgen gewähren erfüllt, ist bei Ihrem IAM-Prinzipal kein Zugriffseintrag vorhanden. Wenn es vorhanden ist, sind dafür nicht die erforderlichen Kubernetes-Gruppennamen definiert oder ihm ist nicht die richtige Zugriffsrichtlinie zugeordnet. Weitere Informationen finden Sie unter the section called “Zugriffseinträge”.
- Wenn Ihr Cluster die Mindestanforderungen für die Plattform in IAM-Benutzern Zugriff auf Kubernetes mit EKS-Zugriffseinträgen gewähren nicht erfüllen, ist kein Eintrag mit Ihrem IAM-Prinzipal in `aws-auth ConfigMap` vorhanden. Wenn er vorhanden ist, ist er nicht den Kubernetes-Gruppennamen zugeordnet, die an ein Kubernetes `Role` oder `ClusterRole` mit den erforderlichen Berechtigungen gebunden sind. Weitere Informationen zu rollenbasierten Autorisierungsobjekten (RBAC) von Kubernetes finden Sie unter Verwendung der RBAC-Autorisierung in der Kubernetes-Dokumentation. Sie können Ihre aktuellen `aws-auth ConfigMap`-Einträge anzeigen, indem Sie im folgenden Befehl *`my-cluster`* durch den Namen Ihres Clusters ersetzen und dann den geänderten Befehl ausführen: `eksctl get iamidentitymapping --cluster` *`my-cluster`* . Wenn in `ConfigMap` kein Eintrag mit dem ARN Ihres IAM-Prinzipals enthalten ist, geben Sie in Ihrem Terminal `eksctl create iamidentitymapping --help` ein, um zu erfahren, wie Sie einen solchen Eintrag erstellen.

Wenn Sie die AWS CLI installieren und konfigurieren, können Sie die von Ihnen verwendeten IAM-Anmeldeinformationen konfigurieren. Weitere Informationen finden Sie unter Konfiguration der AWS CLI im Benutzerhandbuch zur AWS-Befehlszeilenschnittstelle. Sie können auch `kubectl` für die Verwendung einer IAM-Rolle konfigurieren, wenn Sie für den Zugriff auf Kubernetes-Objekte in Ihrem Cluster eine IAM-Rolle annehmen. Weitere Informationen finden Sie unter the section called “Zugriff auf den Cluster mit kubectl”.

`hostname doesn’t match`

Die Python-Version Ihres Systems muss `2.7.9` oder höher sein. Andernfalls erhalten Sie `hostname doesn’t match`-Fehler mit AWS-CLI-Aufrufen an Amazon EKS. Weitere Informationen finden Sie unter Was sind Fehlermeldungen vom Typ „Host-Name stimmt nicht überein“? in den Häufig gestellten Fragen zu Python-Anfragen.

`getsockopt: no route to host`

Docker wird im `172.17.0.0/16`-CIDR-Bereich in Amazon-EKS-Clustern ausgeführt. Wir empfehlen, dass die VPC-Subnetze Ihres Clusters diesen Bereich nicht überschneiden. Andernfalls erhalten Sie den folgenden Fehler:

```
Error: : error upgrading connection: error dialing backend: dial tcp
 172.17.<nn>.<nn>:10250: getsockopt: no route to host
```

`Instances failed to join the Kubernetes cluster`

Wenn die Fehlermeldung `Instances failed to join the Kubernetes cluster` in der AWS-Managementkonsole angezeigt wird, stellen Sie sicher, dass entweder der private Endpunktzugriff des Clusters aktiviert ist oder dass Sie CIDR-Blöcke für den öffentlichen Endpunktzugriff korrekt konfiguriert haben. Weitere Informationen finden Sie unter the section called “Cluster-Endpunktzugriff”.

Fehlercodes bei verwalteten Knotengruppen

Wenn bei Ihrer verwaltete Knotengruppe ein Hardwarezustandsproblem auftritt, gibt Amazon EKS einen Fehlercode zurück, der Ihnen bei der Diagnose des Problems hilft. Diese Zustandsprüfungen erkennen keine Softwareprobleme, da sie auf Amazon-EC2-Zustandsprüfungen basieren. Die Fehlercodes sind in der folgenden Liste beschrieben.

AccessDenied

> Amazon EKS oder einer oder mehrere Ihrer verwalteten Knoten können sich bei Ihrem Kubernetes-Cluster-API-Server nicht authentifizieren oder autorisieren. Weitere Informationen zum Beheben einer häufigen Ursache finden Sie unter the section called “Behebung einer häufigen Ursache für `AccessDenied`-Fehler bei verwalteten Knotengruppen”. Private Windows-AMIs können ebenfalls diesen Fehlercode zusammen mit der Fehlermeldung `Not authorized for images` verursachen. Weitere Informationen finden Sie unter the section called “`Not authorized for images`”.

AmiIdNotFound

> Die AMI-ID, die Ihrer Startvorlage zugeordnet ist, konnte nicht gefunden werden. Stellen Sie sicher, dass das AMI vorhanden ist und für Ihr Konto freigegeben ist.

AutoScalingGroupNotFound

Die Auto-Scaling-Gruppe, die der verwalteten Knotengruppe zugeordnet ist, konnte nicht gefunden werden. Möglicherweise können Sie eine Auto-Scaling-Gruppe mit den gleichen Einstellungen zur Wiederherstellung erstellen.

ClusterUnreachable

Amazon EKS oder einer oder mehrere Ihrer verwalteten Knoten können nicht mit Ihrem Kubernetes-Cluster-API-Server kommunizieren. Dies kann passieren, wenn Netzwerkunterbrechungen auftreten oder wenn API-Server eine Zeitüberschreitung für die Verarbeitung von Anforderungen vornehmen.

Ec2SecurityGroupNotFound

Die Cluster-Sicherheitsgruppe für den Cluster konnte nicht gefunden werden. Sie müssen Ihren Cluster neu erstellen.

Ec2SecurityGroupDeletionFailure

Die RAS-Sicherheitsgruppe für Ihre verwaltete Knotengruppe konnte nicht gelöscht werden. Entfernen Sie alle Abhängigkeiten aus der Sicherheitsgruppe.

Ec2LaunchTemplateNotFound

Die Amazon-EC2-Startvorlage für Ihre verwaltete Knotengruppe konnte nicht gefunden werden. Sie müssen Ihre Knotengruppe neu erstellen, um sie wiederherzustellen.

Ec2LaunchTemplateVersionMismatch

Die Amazon-EC2-Startvorlagenversion für Ihre verwaltete Knotengruppe stimmt nicht mit der von Amazon EKS erstellten Version überein. Möglicherweise können Sie zu der Version zurückkehren, die Amazon EKS für die Wiederherstellung erstellt hat.

IamInstanceProfileNotFound

Das IAM-Instance-Profil für Ihre verwaltete Knotengruppe konnte nicht gefunden werden. Möglicherweise können Sie erneut ein Instance-Profil mit den gleichen Einstellungen zur Wiederherstellung erstellen.

IamNodeRoleNotFound

Die IAM-Rolle für Ihre verwaltete Knotengruppe konnte nicht gefunden werden. Möglicherweise können Sie erneut eine IAM-Rolle mit den gleichen Einstellungen zur Wiederherstellung erstellen.

AsgInstanceLaunchFailures

Beim Versuch, Instances zu starten, treten in Ihrer Auto-Scaling-Gruppe Fehlfunktionen auf.

NodeCreationFailure

Ihre gestarteten Instances können sich nicht bei Ihrem Amazon-EKS-Cluster registrieren. Häufige Ursachen für diesen Fehler sind unzureichende Knoten-IAM-Rollen-Berechtigungen oder fehlender ausgehender Internetzugriff für die Knoten. Ihre Knoten müssen eine der folgenden Anforderungen erfüllen:

- Kann über eine öffentliche IP-Adresse auf das Internet zugreifen. Die Sicherheitsgruppe, die mit dem Subnetz verknüpft ist, in dem sich der Knoten befindet, muss die Kommunikation zulassen. Weitere Informationen erhalten Sie unter the section called “Subnetz-Anforderungen und -Überlegungen” und the section called “Anforderungen an Sicherheitsgruppen”.
- Ihre Knoten und VPC müssen die Anforderungen unter Bereitstellen privater Cluster mit eingeschränktem Internetzugriff erfüllen.

InstanceLimitExceeded

Ihr AWS-Konto kann keine weiteren Instances des angegebenen Instance-Typs starten. Möglicherweise können Sie eine Erhöhung des Amazon-EC2-Instance-Limits zur Wiederherstellung anfordern.

InsufficientFreeAddresses

Mindestens eines der Subnetze, die Ihrer verwalteten Knotengruppe zugeordnet sind, verfügt nicht über genügend verfügbare IP-Adressen für neue Knoten.

InternalFailure

Diese Fehler werden normalerweise durch ein serverseitiges Amazon-EKS-Problem verursacht.

Behebung einer häufigen Ursache für **`AccessDenied`**-Fehler bei verwalteten Knotengruppen

Die häufigste Ursache für `AccessDenied`-Fehler beim Ausführen von Vorgängen auf verwalteten Knotengruppen ist das Fehlen von `eks:node-manager`, `ClusterRole` oder `ClusterRoleBinding`. Amazon EKS richtet diese Ressourcen in Ihrem Cluster als Teil des Onboarding mit verwalteten Knotengruppen ein, die für die Verwaltung der Knotengruppen erforderlich sind.

Die `ClusterRole` kann sich im Laufe der Zeit ändern, sollte aber dem folgenden Beispiel ähneln:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: eks:node-manager
rules:
- apiGroups:
  - ''
  resources:
  - pods
  verbs:
  - get
  - list
  - watch
  - delete
- apiGroups:
  - ''
  resources:
  - nodes
  verbs:
  - get
  - list
  - watch
  - patch
- apiGroups:
  - ''
  resources:
  - pods/eviction
  verbs:
  - create
```

Die `ClusterRoleBinding` kann sich im Laufe der Zeit ändern, sollte aber dem folgenden Beispiel ähneln:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks:node-manager
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: eks:node-manager
```

```
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: User
  name: eks:node-manager
```

Stellen Sie sicher, dass die Ressource `eks:node-manager` `ClusterRole` besteht.

```
kubectl describe clusterrole eks:node-manager
```

Wenn bestehend, vergleichen Sie die Ausgabe mit dem vorherigen `ClusterRole`-Beispiel.

Stellen Sie sicher, dass die Ressource `eks:node-manager` `ClusterRoleBinding` besteht.

```
kubectl describe clusterrolebinding eks:node-manager
```

Wenn bestehend, vergleichen Sie die Ausgabe mit dem vorherigen `ClusterRoleBinding`-Beispiel.

Wenn Sie einen fehlenden oder defekten `ClusterRole` oder `ClusterRoleBinding` als Ursache eines `AcessDenied`-Fehlers bei der Anforderung verwalteter Knotengruppen-Operationen finden, können Sie sie wiederherstellen. Speichern Sie den folgenden Inhalt in einer Datei mit dem Namen *`eks-node-manager-role.yaml`*.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: eks:node-manager
rules:
- apiGroups:
  - ''
  resources:
  - pods
  verbs:
  - get
  - list
  - watch
  - delete
- apiGroups:
  - ''
  resources:
  - nodes
  verbs:
  - get
```

```
  - list
  - watch
  - patch
- apiGroups:
  - ''
  resources:
  - pods/eviction
  verbs:
  - create
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks:node-manager
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: eks:node-manager
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: User
  name: eks:node-manager
```

Wenden Sie die Datei an.

```
kubectl apply -f eks-node-manager-role.yaml
```

Wiederholen Sie den Vorgang der Knotengruppe, um festzustellen, ob das Problem dadurch behoben wurde.

Not authorized for images

Eine mögliche Ursache für eine `Not authorized for images`-Fehlermeldung ist die Verwendung eines privaten Amazon-EKS-Windows-AMI zum Starten von Windows-verwalteten Knotengruppen. Nach der Veröffentlichung neuer Windows-AMIs macht AWS AMIs, die älter als 4 Monate sind, privat, wodurch sie nicht mehr zugänglich sind. Wenn Ihre verwaltete Knotengruppe ein privates Windows-AMI verwendet, sollten Sie eine Aktualisierung Ihrer verwalteten Windows-Knotengruppe in Betracht ziehen. Wir können zwar nicht garantieren, dass wir Zugriff auf privat gemachte AMIs gewähren können, Sie können jedoch Zugriff anfordern, indem Sie beim AWS Support ein Ticket einreichen. Weitere Informationen finden Sie unter Patches im Amazon-EC2-Benutzerhandbuch.

Knoten befindet sich im **`NotReady`**-Zustand

Wenn Ihr Knoten in einen `NotReady`-Status wechselt, deutet dies wahrscheinlich darauf hin, dass der Knoten nicht ordnungsgemäß funktioniert und für die Planung neuer Pods nicht verfügbar ist. Dies kann verschiedene Ursachen haben, beispielsweise wenn dem Knoten ausreichende Ressourcen für CPU, Arbeitsspeicher oder verfügbaren Festplattenspeicher fehlen.

Bei für Amazon EKS optimierten Windows-AMIs sind in der `kubelet`-Konfiguration standardmäßig keine Reservierungen für Rechenressourcen festgelegt. Um Ressourcenprobleme zu vermeiden, können Sie Rechenressourcen für Systemprozesse reservieren, indem Sie dem `kubelet` Konfigurationswerte für kube-reserved und/oder system-reserved bereitstellen. Dies tun Sie mithilfe des `-KubeletExtraArgs`-Befehlszeilen-Parameters im Bootstrap-Skript. Weitere Informationen finden Sie unter Reservieren von Rechenressourcen für System Daemons in der Kubernetes-Dokumentation und the section called “Bootstrap-Skript-Konfigurationsparameter” in diesem Benutzerhandbuch.

EKS-Protokollsammler

Zur Fehlerbehebung bei Problemen mit Amazon-EKS-Knoten steht auf den Knoten unter `/etc/eks/log-collector-script/eks-log-collector.sh` ein vorgefertigtes Skript zur Verfügung. Sie können das Skript verwenden, um Diagnoseprotokolle für Support-Fälle und allgemeine Fehlerbehebung zu sammeln.

Mit dem folgenden Befehl können Sie das Skript auf dem Knoten ausführen:

```
sudo bash /etc/eks/log-collector-script/eks-log-collector.sh
```

> **Note**
>
> Wenn das Skript an diesem Speicherort nicht vorhanden ist. Sie können das Skript mit dem folgenden Befehl manuell herunterladen und ausführen:
>
> ```
> curl -O https://amazon-eks.s3.amazonaws.com/support/log-collector-script/linux/eks-log-collector.sh
> sudo bash eks-log-collector.sh
> ```

Das Skript sammelt die folgenden Diagnoseinformationen:

```
$ sudo bash /etc/eks/log-collector-script/eks-log-collector.sh

      This is version 0.7.8. New versions can be found at https://github.com/awslabs/
amazon-eks-ami/blob/main/log-collector-script/

Trying to collect common operating system logs...
Trying to collect kernel logs...
Trying to collect mount points and volume information...
...
...

 Done... your bundled logs are located in /var/log/eks_i-EXAMPLE_2025-03-25_0000-
UTC_0.7.8.tar.gz
```

Die Diagnoseinformationen werden gesammelt und gespeichert unter:

```
/var/log/eks_i-EXAMPLE_2025-03-25_0000-UTC_0.7.8.tar.gz
```

Weitere Informationen zum Abrufen des Protokollpakets für Bottlerocket-Knoten finden Sie im Bottlerocket-Protokoll.

Container-Laufzeitnetzwerk nicht bereit

Möglicherweise erhalten Sie einen `Container runtime network not ready`-Fehler und einen Autorisierungsfehler, die den folgenden ähneln:

```
4191 kubelet.go:2130] Container runtime network not ready: NetworkReady=false
 reason:NetworkPluginNotReady message:docker: network plugin is not ready: cni config
 uninitialized
4191 reflector.go:205] k8s.io/kubernetes/pkg/kubelet/kubelet.go:452: Failed to list
 *v1.Service: Unauthorized
4191 kubelet_node_status.go:106] Unable to register node
 "ip-10-40-175-122.ec2.internal" with API server: Unauthorized
4191 reflector.go:205] k8s.io/kubernetes/pkg/kubelet/kubelet.go:452: Failed to list
 *v1.Service: Unauthorized
```

Dieses Problem kann folgende Ursachen haben:

1. Auf Ihrem Cluster ist keine `aws-auth ConfigMap` vorhanden oder sie enthält keine Einträge für die IAM-Rolle, mit der Sie Ihre Knoten konfiguriert haben.

Sehen Sie sich zur Behebung des Problems die vorhandenen Einträge in `ConfigMap` an, indem Sie *`my-cluster`* im folgenden Befehl durch den Namen Ihres Clusters ersetzen und dann den geänderten Befehl ausführen: `eksctl get iamidentitymapping --cluster` *`my-cluster`* . Sollten Sie im Zusammenhang mit dem Befehl eine Fehlermeldung erhalten, verfügt Ihr Cluster möglicherweise nicht über eine `aws-auth ConfigMap`. Der folgende Befehl fügt `ConfigMap` einen Eintrag hinzu. Ist `ConfigMap` nicht vorhanden, wird sie durch den Befehl erstellt. Ersetzen Sie *`111122223333`* durch die AWS-Konto-ID für die IAM-Rolle und *`myAmazonEKSNodeRole`* durch den Namen der Rolle Ihres Knotens.

```
eksctl create iamidentitymapping --cluster my-cluster \
    --arn arn:aws:iam::111122223333:role/myAmazonEKSNodeRole --group
 system:bootstrappers,system:nodes \
    --username system:node:{{EC2PrivateDNSName}}
```

Der ARN der von Ihnen angegebenen Rolle darf als Pfad nur `/` enthalten. Wenn der Name Ihrer Rolle also beispielsweise `development/apps/my-role` lautet, müssen Sie ihn beim Angeben des ARN der Rolle in `my-role` ändern. Achten Sie darauf, den ARN der IAM-Rolle des Knotens (nicht den ARN des Instances-Profils) anzugeben.

2. Ihre selbstverwalteten Knoten befinden sich in einem Cluster mit einer Plattformversion, die mindestens der in den Voraussetzungen im Abschnitt IAM-Benutzern Zugriff auf Kubernetes mit EKS-Zugriffseinträgen gewähren aufgeführten Version entspricht. Ein Eintrag für die IAM-Rolle des Knotens ist jedoch nicht im `aws-auth ConfigMap` (siehe vorheriger Punkt) aufgeführt, oder es existiert kein Zugriffseintrag für die Rolle. Sehen Sie sich zur Behebung des Problems Ihre vorhandenen Zugriffseinträge an, indem Sie *`my-cluster`* im folgenden Befehl durch den Namen Ihres Clusters ersetzen und dann den geänderten Befehl ausführen: `aws eks list-access-entries --cluster-name` *`my-cluster`* . Der folgende Befehl fügt einen Zugriffseintrag für die IAM-Rolle des Knotens hinzu. Ersetzen Sie *`111122223333`* durch die AWS-Konto-ID für die IAM-Rolle und *`myAmazonEKSNodeRole`* durch den Namen der Rolle Ihres Knotens. Ersetzen Sie im Falle eines Windows-Knotens *`EC2_Linux`* durch `EC2_Windows`. Achten Sie darauf, den ARN der IAM-Rolle des Knotens (nicht den ARN des Instances-Profils) anzugeben.

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/myAmazonEKSNodeRole --type EC2_LINUX
```

TLS-Handshake-Zeitüberschreitung

Wenn ein Knoten nicht in der Lage ist, eine Verbindung zum öffentlichen API-Server-Endpunkt herzustellen, erhalten Sie möglicherweise einen Fehler ähnlich dem folgenden.

```
server.go:233] failed to run Kubelet: could not init cloud provider "aws": error
 finding instance i-1111f2222f333e44c: "error listing AWS instances: \"RequestError:
 send request failed\\ncaused by: Post  net/http: TLS handshake timeout\""
```

Der `kubelet`-Prozess wird weiter fortgesetzt und testet den API-Server-Endpunkt. Der Fehler kann auch vorübergehend während jeder Prozedur auftreten, die eine laufende Aktualisierung des Clusters auf der Steuerungsebene durchführt, z. B. eine Konfigurationsänderung oder eine Versionsaktualisierung.

Um das Problem zu beheben, überprüfen Sie die Routing-Tabelle und die Sicherheitsgruppen, um sicherzustellen, dass der Datenverkehr von den Knoten den öffentlichen Endpunkt erreichen kann.

InvalidClientTokenId

Wenn Sie IAM-Rollen für Servicekonten für einen Pod oder DaemonSet verwenden, der in einem Cluster in einer AWS-Region China bereitgestellt ist, und die `AWS_DEFAULT_REGION`-Umgebungsvariable in der Spezifikation nicht festgelegt haben, kann für den Pod oder DaemonSet der folgende Fehler auftreten:

```
An error occurred (InvalidClientTokenId) when calling the GetCallerIdentity operation:
 The security token included in the request is invalid
```

Zur Behebung des Problems müssen Sie die `AWS_DEFAULT_REGION`-Umgebungsvariable Ihrer Pod- oder DaemonSet-Spezifikation wie in der folgenden Beispiel-Pod-Spezifikation hinzufügen.

```
apiVersion: v1
kind: Pod
metadata:
  name: envar-demo
  labels:
    purpose: demonstrate-envars
spec:
  containers:
```

```
  - name: envar-demo-container
    image: gcr.io/google-samples/node-hello:1.0
    env:
    - name: AWS_DEFAULT_REGION
      value: "region-code"
```

Knotengruppen müssen vor dem Upgrade der Steuerebene mit der Kubernetes-Version übereinstimmen

Bevor Sie eine Steuerebene auf eine neue Kubernetes-Version aktualisieren, muss die Nebenversion der verwalteten und Fargate-Knoten in Ihrem Cluster mit der Version der aktuellen Version Ihrer Steuerebene übereinstimmen. Die Amazon EKS-`update-cluster-version`-API akzeptiert keine Anforderungen, bis für alle von Amazon EKS verwalteten Knoten ein Upgrade auf die aktuelle Cluster-Version durchgeführt wurde. Amazon EKS stellt APIs zum Upgraden verwalteter Knoten bereit. Informationen zum Upgrade der Kubernetes-Version einer verwalteten Knotengruppe finden Sie unter the section called “Aktualisierung”. Löschen Sie zum Upgraden der Version eines Fargate-Knotens den Pod, der durch den Knoten dargestellt wird, und stellen Sie den Pod erneut bereit, nachdem Sie das Upgrade für Ihre Steuerebene durchgeführt haben. Weitere Informationen finden Sie unter the section called “Kubernetes-Version aktualisieren”.

Beim Starten vieler Knoten gibt es **`Too Many Requests`**-Fehler

Wenn Sie viele Knoten gleichzeitig starten, wird möglicherweise die Fehlermeldung `Too Many Requests` in den Ausführungsprotokollen für die Amazon-EC2-Benutzerdaten angezeigt. Dies kann auftreten, weil die Steuerebene mit `describeCluster`-Aufrufen überlastet wird. Die Überladung führt zu einer Drosselung, Knoten, die das Bootstrap-Skript nicht ausführen können, und Knoten, die dem Cluster insgesamt nicht beitreten können.

Stellen Sie sicher, dass die Argumente `--apiserver-endpoint`, `--b64-cluster-ca` und `--dns-cluster-ip` an das Bootstrap-Skript des Knotens übergeben werden. Wenn Sie diese Argumente einschließen, muss das Bootstrap-Skript keinen `describeCluster`-Aufruf ausführen, wodurch verhindert wird, dass die Steuerebene überlastet wird. Weitere Informationen finden Sie unter the section called “Geben Sie Benutzerdaten an, um Argumente an die `bootstrap.sh` Datei zu übergeben, die in einem Amazon EKS-optimierten Linux/Bottlerocket AMI enthalten ist”.

Unautorisierte Fehlerantwort (HTTP 401) bei Kubernetes-API-Serveranfragen

Diese Fehler werden angezeigt, wenn das Servicekonto-Token eines Pods in einem Cluster abgelaufen ist.

Der Kubernetes-API-Server Ihres Amazon-EKS-Clusters lehnt Anfragen mit Tokens ab, die älter als 90 Tage sind. In früheren Kubernetes-Versionen hatten Token kein Ablaufdatum. Clients, die diese Tokens verwenden, müssen die Tokens nun innerhalb einer Stunde aktualisieren. Um zu verhindern, dass der Kubernetes-API-Server Ihre Anfrage aufgrund eines ungültigen Tokens ablehnt, muss die von Ihrem Workload verwendete Kubernetes-Client-SDK-Version dieselbe oder höher als die folgenden Versionen sein:

- Go-Version `0.15.7` und höher
- Python-Version `12.0.0` und höher
- Java-Version `9.0.0` und höher
- JavaScript-Version `0.10.3` und höher
- Ruby `master`-Branch
- Haskell-Version `0.3.0.0`
- C#-Version `7.0.5` und höher

Sie können alle vorhandenen Pods in Ihrem Cluster ermitteln, die veraltete Token verwenden. Weitere Informationen finden Sie unter the section called “Servicekonto-Tokens”.

Die Amazon-EKS-Plattformversion liegt mehr als zwei Versionen hinter der aktuellen Plattformversion

Dies kann passieren, wenn Amazon EKS die Plattformversion Ihres Clusters nicht automatisch aktualisieren kann. Obwohl es dafür viele Ursachen gibt, folgen einige der häufigsten Ursachen. Wenn Ihr Cluster von einem dieser Probleme betroffen ist, funktioniert er möglicherweise immer noch, aber seine Plattformversion wird nicht von Amazon EKS aktualisiert.

Problem

Die Cluster-IAM-Rolle wurde gelöscht – Diese Rolle wurde beim Erstellen des Clusters angegeben. Sie können mit dem folgenden Befehl überprüfen, welche Rolle angegeben wurde. Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters.

```
aws eks describe-cluster --name my-cluster --query cluster.roleArn --output text | cut
 -d / -f 2
```

Eine Beispielausgabe sieht wie folgt aus.

```
eksClusterRole
```

Lösung

Erstellen einer neuen Cluster-IAM-Rolle mit demselben Namen.

Problem

Ein bei der Clustererstellung festgelegtes Subnetz wurde gelöscht – Die mit dem Cluster zu verwendenden Subnetze wurden bei der Clustererstellung angegeben. Sie können mit dem folgenden Befehl überprüfen, welche Subnetze angegeben wurden. Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters.

```
aws eks describe-cluster --name my-cluster --query cluster.resourcesVpcConfig.subnetIds
```

Eine Beispielausgabe sieht wie folgt aus.

```
[
"subnet-EXAMPLE1",
"subnet-EXAMPLE2"
]
```

Lösung

Bestätigen Sie, ob die Subnetz-IDs in Ihrem Konto vorhanden sind.

```
vpc_id=$(aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.vpcId --output text)
aws ec2 describe-subnets --filters "Name=vpc-id,Values=$vpc_id" --query
 "Subnets[*].SubnetId"
```

Eine Beispielausgabe sieht wie folgt aus.

```
[
"subnet-EXAMPLE3",
"subnet-EXAMPLE4"
]
```

Wenn die in der Ausgabe zurückgegebenen Subnetz-IDs nicht mit den Subnetz-IDs übereinstimmen, die bei der Erstellung des Clusters angegeben wurden, müssen Sie die vom Cluster verwendeten Subnetze ändern, wenn Sie möchten, dass Amazon EKS den Cluster aktualisiert. Dies liegt daran, dass Amazon EKS, wenn Sie bei der Erstellung Ihres Clusters mehr als zwei Subnetze angegeben haben, nach dem Zufallsprinzip Subnetze auswählt, die Sie für die Erstellung neuer Elastic-Network-Schnittstellen angegeben haben. Diese Netzwerkschnittstellen ermöglichen es der Steuerebene, mit Ihren Knoten zu kommunizieren. Amazon EKS aktualisiert den Cluster nicht, wenn das von ihm ausgewählte Subnetz nicht vorhanden ist. Sie haben keine Kontrolle darüber, in welchem der Subnetze, die Sie bei der Clustererstellung angegeben haben, Amazon EKS wählt, um eine neue Netzwerkschnittstelle zu erstellen.

Wenn Sie eine Kubernetes-Versionsaktualisierung für Ihren Cluster initiieren, kann die Aktualisierung aus dem gleichen Grund fehlschlagen.

Problem

Eine bei der Clustererstellung angegebene Sicherheitsgruppe wurde gelöscht – Wenn Sie bei der Clustererstellung Sicherheitsgruppen angegeben haben, können Sie deren IDs mit dem folgenden Befehl anzeigen. Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters.

```
aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.securityGroupIds
```

Eine Beispielausgabe sieht wie folgt aus.

```
[
    "sg-EXAMPLE1"
]
```

Wenn `[]` zurückgegeben wird, wurden beim Erstellen des Clusters keine Sicherheitsgruppen angegeben und eine fehlende Sicherheitsgruppe ist nicht das Problem. Wenn Sicherheitsgruppen zurückgegeben werden, vergewissern Sie sich, dass die Sicherheitsgruppen in Ihrem Konto vorhanden sind.

Lösung

Bestätigen Sie, ob diese Sicherheitsgruppen in Ihrem Konto vorhanden sind.

```
vpc_id=$(aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.vpcId --output text)
aws ec2 describe-security-groups --filters "Name=vpc-id,Values=$vpc_id" --query
 "SecurityGroups[*].GroupId"
```

Eine Beispielausgabe sieht wie folgt aus.

```
[
"sg-EXAMPLE2"
]
```

Wenn die in der Ausgabe zurückgegebenen Sicherheitsgruppen-IDs nicht mit den Sicherheitsgruppen-IDs übereinstimmen, die bei der Erstellung des Clusters angegeben wurden, müssen Sie die vom Cluster verwendeten Sicherheitsgruppen ändern, wenn Sie möchten, dass Amazon EKS den Cluster aktualisiert. Amazon EKS aktualisiert einen Cluster nicht, wenn die bei der Cluster-Erstellung angegebenen Sicherheitsgruppen-IDs nicht vorhanden sind.

Wenn Sie eine Kubernetes-Versionsaktualisierung für Ihren Cluster initiieren, kann die Aktualisierung aus dem gleichen Grund fehlschlagen.

- Sie haben nicht mindestens sechs verfügbare IP-Adressen (wir empfehlen jedoch 16) in jedem der Subnetze, die Sie bei der Erstellung Ihres Clusters angegeben haben. Wenn Sie nicht über genügend verfügbare IP-Adressen im Subnetz verfügen, müssen Sie entweder IP-Adressen im Subnetz freigeben oder die vom Cluster verwendeten Subnetze ändern, um Subnetze mit genügend verfügbaren IP-Adressen zu verwenden.
- Sie haben die Verschlüsselung von Geheimnissen aktiviert, als Sie Ihren Cluster erstellt haben, und der von Ihnen angegebene AWS-KMS-Schlüssel wurde gelöscht. Wenn Sie möchten, dass Amazon EKS den Cluster aktualisiert, müssen Sie einen neuen Cluster erstellen

Häufig gestellte Fragen zur Cluster-Integrität und Fehlercodes mit Lösungspfaden

Amazon EKS erkennt Probleme mit Ihren EKS-Clustern und der Cluster-Infrastruktur und speichert diese im Objekt Integrität Ihrer EKS-Cluster-Ressource. Mithilfe von Cluster-Integritätsinformationen

können Sie Cluster-Probleme schneller erkennen, behandeln und beheben. Dies ermöglicht die Erstellung von sichereren und aktuelleren Anwendungsumgebungen. Darüber hinaus kann es vorkommen, dass Sie aufgrund von Problemen mit der erforderlichen Infrastruktur oder Cluster-Konfiguration kein Upgrade auf neuere Versionen von Kubernetes durchführen können oder dass Amazon EKS keine Sicherheitsaktualisierungen auf einem beeinträchtigten Cluster installieren kann. Es kann drei Stunden dauern, bis Amazon EKS Probleme erkennt oder feststellt, dass ein Problem behoben wurde.

Amazon EKS und die Benutzer sind gemeinsam für die Integrität eines Amazon EKS-Clusters verantwortlich. Sie selbst sind für die erforderliche Infrastruktur von IAM-Rollen und Amazon VPC-Subnetzen sowie für andere notwendige Infrastrukturkomponenten verantwortlich, die im Voraus bereitgestellt werden müssen. Amazon EKS erkennt Änderungen an der Konfiguration dieser Infrastruktur und des Clusters.

Um den Status Ihres Clusters in der Amazon-EKS-Konsole anzuzeigen, suchen Sie in der Registerkarte Cluster-Integritätsprobleme des Beobachtbarkeits-Dashboards, das Sie über die Amazon-EKS-Cluster-Detailseite aufrufen können, nach einer Tabelle mit dem Namen Integritätsprobleme. Diese Daten sind auch durch Aufrufen der `DescribeCluster`-Aktion in der EKS-API verfügbar, beispielsweise über die AWS-Befehlszeilenschnittstelle.

Warum sollte ich dieses Feature nutzen?

> Sie erhalten einen besseren Einblick in die Integrität Ihres Amazon EKS-Clusters und können Probleme schnell diagnostizieren und beheben, ohne Zeit für das Debuggen oder das Erstellen von AWS-Supportanfragen aufwenden zu müssen. Ein Beispiel: Sie haben versehentlich ein Subnetz für den Amazon-EKS-Cluster gelöscht und Amazon EKS kann keine kontoübergreifenden Netzwerkschnittstellen erstellen und keine AWS-`kubectl`-Befehle wie „ exec“ oder „`kubectl` logs“ ausführen. Bei diesen tritt folgender Fehler auf: `Error from server: error dialing backend: remote error: tls: internal error.` Nun wird ein Amazon EKS-Integritätsproblem mit folgendem Hinweis angezeigt: `subnet-da60e280 was deleted: could not create network interface`.

In welcher Beziehung steht dieses Feature zu anderen AWS-Services bzw. wie arbeitet sie mit ihnen zusammen?

> IAM-Rollen und Amazon VPC-Subnetze sind zwei Beispiele für erforderliche Infrastrukturkomponenten, für die die Cluster-Integritätsfunktion Probleme erkennt. Diese Funktion gibt detaillierte Informationen zurück, wenn diese Ressourcen nicht ordnungsgemäß konfiguriert sind.

Fallen für Cluster mit Integritätsproblemen Gebühren an?

Ja. Jeder Amazon EKS-Cluster wird zu den Amazon EKS-Standardpreisen abgerechnet. Die Funktion Cluster-Integrität steht ohne Aufpreis zur Verfügung.

Funktioniert dieses Feature mit Amazon-EKS-Clustern in AWS Outposts?

Ja. Cluster-Probleme werden für EKS-Cluster in der AWS Cloud erkannt. Dies schließt auch erweiterte Cluster in AWS Outposts sowie lokale Cluster in AWS ein. Die Cluster-Integrität erkennt keine Probleme mit Amazon EKS Anywhere oder Amazon EKS Distro (EKS-D).

Kann ich benachrichtigt werden, wenn neue Probleme erkannt werden?

Ja. AWS sendet eine E-Mail und eine Benachrichtigung über das persönliche Integritäts-Dashboard, wenn neue Probleme mit der Cluster-Integrität erkannt werden.

Werden in der Konsole Warnungen im Zusammenhang mit Integritätsproblemen ausgegeben?

Ja. Cluster mit Problemen verfügen über ein Banner im oberen Bereich der Konsole.

Die ersten beiden Spalten werden für API-Antwortwerte benötigt. Das dritte Feld des Health ClusterIssue-Objekts sind resourceIds, deren Rückgabe vom Problemtyp abhängt.

Code	Fehlermeldung	ResourceIds	Cluster wiederherstellbar?
SUBNET_NOT_FOUND	Es konnten derzeit keine Subnetze gefunden werden, die Ihrem Cluster zugeordnet sind. Rufen Sie die Amazon-EKS-API „update-cluster-config“ auf, um Subnetze zu aktualisieren.	Subnetz-IDs	Ja
SECURITY_GROUP_NOT_FOUND	Wir konnten eine oder mehrere Sicherheitsgruppen, die	Sicherheitsgruppen-IDs	Ja

Code	Fehlermeldung	ResourceIds	Cluster wiederherstellbar?
	derzeit Ihrem Cluster zugeordnet sind, nicht finden. Call Amazon EKS update-cluster-config API to update security groups		
IP_NOT_AVAILABLE	One or more of the subnets associated with your cluster does not have enough available IP addresses for Amazon EKS to perform cluster management operations. Free up addresses in the subnet(s), or associate different subnets to your cluster using the Amazon EKS update-cluster-config API.	Subnetz-IDs	Ja
VPC_NOT_FOUND	Wir konnten die Ihrem Cluster zugeordnete VPC nicht finden. You must delete and recreate your cluster.	VPC-ID	Nein

Code	Fehlermeldung	ResourceIds	Cluster wiederherstellbar?
ASSUME_ROLE_ACCESS_DENIED	Your cluster is not using the Amazon EKS service-linked-role. Wir konnten die Ihrem Cluster zugeordnete Rolle nicht übernehmen, um die erforderlichen Amazon-EKS-Verwaltungsvorgänge durchzuführen. Check the role exists and has the required trust policy.	Die IAM-Rolle des Clusters	Ja
PERMISSION_ACCESS_DENIED	Your cluster is not using the Amazon EKS service-linked-role. The role associated with your cluster does not grant sufficient permissions for Amazon EKS to perform required management operations. Check the policies attached to the cluster role and if any separate deny policies are applied.	Die IAM-Rolle des Clusters	Ja

Code	Fehlermeldung	ResourceIds	Cluster wiederherstellbar?
ASSUME_ROLE_ACCESS_DENIED_USING_SLR	Wir konnten die serviceverknüpfte Rolle des Amazon-EKS-Cluster-Managements nicht übernehmen. Check the role exists and has the required trust policy.	Die mit dem Amazon EKS-Service verknüpfte Rolle	Ja
PERMISSION_ACCESS_DENIED_USING_SLR	The Amazon EKS cluster management service-linked-role does not grant sufficient permissions for Amazon EKS to perform required management operations. Check the policies attached to the cluster role and if any separate deny policies are applied.	Die mit dem Amazon EKS-Service verknüpfte Rolle	Ja
OPT_IN_REQUIRED	Ihr Konto verfügt derzeit über kein Abonnement für den Amazon-EC2-Service. Update your account subscriptions in your account settings page.	N/A	Ja

Code	Fehlermeldung	ResourceIds	Cluster wiederherstellbar?
STS_REGIONAL_ENDPOINT_DISABLED	Der regionale STS-Endpunkt ist deaktiviert. Enable the endpoint for Amazon EKS to perform required cluster management operations.	N/A	Ja
KMS_KEY_DISABLED	Der Ihrem Cluster zugeordnete AWS-KMS-Schlüssel ist deaktiviert. Re-enable the key to recover your cluster.	Der ARN des KMS-Schlüssels	Ja
KMS_KEY_NOT_FOUND	Wir konnten den AWS-KMS-Schlüssel für Ihren Cluster nicht finden. You must delete and recreate the cluster.	Der ARN des KMS-Schlüssels	Nein
KMS_GRANT_REVOKED	Berechtigungen für den Ihrem Cluster zugeordneten AWS-KMS-Schlüssel werden widerrufen. You must delete and recreate the cluster.	Der ARN des KMS-Schlüssels	Nein

Erweiterte Amazon-EKS-Funktionen mit Open-Source-Projekten

Diese Open-Source-Projekte erweitern die Funktionalität von Kubernetes-Clustern, die auf oder außerhalb von laufen AWS, einschließlich Clustern, die von Amazon EKS verwaltet werden.

Support für in EKS bereitgestellte Software

Bei der Durchsicht der Amazon-EKS-Dokumentation werden Sie in unseren Verfahren und Beispielen auf Verweise auf verschiedene Open-Source-Tools und -Software stoßen. Zu diesen Tools gehören Kubernetes Metrics Server und Cert Manager.

Bitte beachten Sie, dass jegliche Drittanbieter- oder Open-Source-Software, die Sie bereitstellen, nicht in den Geltungsbereich Ihrer AWS Support-Verträge fällt. Ein Vorteil der Verwendung von Kubernetes ist die aktive Open-Source-Community. Wir empfehlen die direkte Zusammenarbeit mit den entsprechenden Open-Source-Communitys und Projektverantwortlichen, um geeignete Supportkanäle für solche Komponenten einzurichten. Weitere Informationen finden Sie in den abgeschlossenen und in der Entwicklung befindlichen Projekten der Cloud Native Computing Foundation (CNCF).

Das Kubernetes-Ökosystem umfasst zahlreiche Projekte und Komponenten, die sich hinsichtlich Community-Support, Reaktionszeiten und vorgesehenen Anwendungsfällen unterscheiden. Wenn Sie diese Technologien zusammen mit EKS implementieren, stellen Sie sicher, dass Sie die Support-Matrix für jede Komponente verstehen.

AWS verwaltet die Open-Source-Komponenten, die wir in die EKS-Steuerebene integrieren. Dazu gehört unsere umfassende Sicherheits-Pipeline, die Verifizierung von Entwicklungen, Schwachstellen-Scans, Validierungstests und Patch-Management für alle von uns verteilten Container-Images und Binär-Dateien umfasst. AWS Ist beispielsweise für den Kubernetes API-Server verantwortlich. Der Kubernetes-API-Server ist durch das Amazon EKS Service Level Agreement abgedeckt. Sie können Ihr Supportpaket für Amazon Web Services nutzen, um Probleme mit dem Kubernetes-API-Server zu beheben oder allgemeine Anleitungen zu erhalten.

Sie müssen den Support, der für verschiedene Amazon EKS-Add-Ons angeboten wird, sorgfältig prüfen. AWS Add-Ons sind die einzige Art von Amazon EKS-Add-on, die vollständig von unterstützt werden AWS. AWS Marketplace-Add-Ons werden hauptsächlich von AWS Partnern unterstützt.

Community-Add-ons erhalten grundlegenden Lebenszyklus-Support von AWS. Weitere Informationen finden Sie unter Add-On-Support.

Jedes EKS-Add-on, unabhängig vom Typ, erhält grundlegenden Lebenszyklus-Support von EKS, einschließlich Marketplace-Add-Ons. Der allgemeine Lebenszyklus-Support umfasst die Installation und Deinstallation des Add-Ons. Weitere Informationen zu den verfügbaren Amazon-EKS-Add-Ons und den damit verbundenen Support-Stufen finden Sie unter Umfang des Supports für Amazon-EKS-Add-Ons. Add-Ons, die vollständig von unterstützt werden AWS, finden Sie unter Amazon Web Services Services-Add-Ons.

- Weitere Informationen zu unseren Sicherheitsmaßnahmen und Support-Grenzen finden Sie unter Sicherheit in Amazon EKS.
- Weitere Informationen zu Community- und AWS Marketplace-Add-ons, die über Amazon EKS Add-ons erhältlich sind, finden Sie unter EKS Add-ons Support.

Verwaltungs-Tools

Verwandte Verwaltungs-Tools für Amazon EKS und Kubernetes-Cluster.

eksctl

`eksctl` ist ein einfaches CLI-Tool zum Erstellen von Clustern in Amazon EKS.

- Projekt-URL
- Projekt-Dokumentation
- AWS Open-Source-Blog: eksctl: Amazon EKS-Cluster mit einem Befehl

AWS Controller für Kubernetes

Mit AWS Controllern für Kubernetes können Sie AWS Ressourcen direkt von Ihrem Kubernetes-Cluster aus erstellen und verwalten.

Verfügbar in EKS-Funktionen.

- Projekt-URL
- AWS Open-Source-Blog: AWS Service Operator für Kubernetes jetzt verfügbar

kro (Kube Resource Orchestrator)

Mit kro können Sie benutzerdefinierte Kubernetes erstellen, die mehrere Ressourcen zu Abstraktionen auf höherer Ebene APIs zusammensetzen. Plattformteams können mithilfe von Leitplanken wiederverwendbare Muster definieren, während Anwendungsteams einfache, übergeordnete Muster für die Bereitstellung und Verwaltung von Ressourcen verwenden. APIs

Verfügbar in EKS-Funktionen.

- Projekt-URL
- Projekt-Dokumentation

Argo CD

Argo CD ist ein deklaratives Tool zur GitOps kontinuierlichen Bereitstellung für Kubernetes. Es überwacht kontinuierlich Ihre Git-Repositorys und synchronisiert automatisch Änderungen an Ihren Clustern.

Verfügbar in EKS-Funktionen.

- Projekt-URL
- Projekt-Dokumentation

Flux CD

Flux ist ein Tool, mit dem Sie Ihre Cluster-Konfiguration mit Git verwalten können. Es verwendet einen Operator im Cluster, um Bereitstellungen innerhalb von Kubernetes auszulösen. Weitere Informationen zu Operatoren finden Sie unter OperatorHub.io on GitHub.

- Projekt-URL
- Projekt-Dokumentation

CDK für Kubernetes

Mit dem CDK für Kubernetes (cdk8s) können Sie Kubernetes-Apps und Komponenten mit vertrauten Programmiersprachen definieren. Cdk8s-Apps synthetisieren zu Standard-Kubernetes-Manifests, die auf jeden Kubernetes-Cluster angewendet werden können.

- Projekt-URL
- Projekt-Dokumentation
- AWS Container-Blog: Wir stellen vor: cdk8s+: Intent-Driven für Kubernetes-Objekte APIs

Netzwerk

Verwandte Netzwerkprojekte für Amazon EKS und Kubernetes-Cluster.

Amazon-VPC-CNI-Plug-In für Kubernetes

Amazon EKS unterstützt native VPC-Netzwerke über das Amazon-VPC-CNI-Plug-In für Kubernetes. Das Plugin weist jedem Pod eine IP-Adresse von Ihrer VPC zu.

- Projekt-URL
- Projekt-Dokumentation

AWS Load Balancer Controller für Kubernetes

Der Load AWS Balancer Controller hilft bei der Verwaltung von AWS Elastic Load Balancers für einen Kubernetes-Cluster. Er erfüllt die Anforderungen der Kubernetes Ingress-Ressourcen durch die Bereitstellung von Application Load Balancers. AWS Es erfüllt die Anforderungen der Kubernetes-Serviceressourcen durch die Bereitstellung von Network Load Balancers. AWS

- Projekt-URL
- Projekt-Dokumentation

ExternalDNS

ExternalDNS synchronisiert exponierte Kubernetes-Dienste und -Ingresses mit DNS-Anbietern wie Amazon Route 53 und Service Discovery. AWS

- Projekt-URL
- Projekt-Dokumentation

Machine Learning

Verwandte Machine-Learning-Projekte für Amazon EKS und Kubernetes-Cluster.

Kubeflow

Ein Machine-Learning-Toolkit für Kubernetes.

- Projekt-URL
- Projekt-Dokumentation
- AWS Open-Source-Blog: Kubeflow auf Amazon EKS

Auto Scaling

Verwandte Auto-Scaling-Projekte für Amazon EKS und Kubernetes-Cluster.

Cluster Autoscaler

Cluster Autoscaler ist ein Tool, das die Größe des Kubernetes-Clusters automatisch an die CPU- und Speicher-Auslastung anpasst.

- Projekt-URL
- Projekt-Dokumentation
- Amazon-EKS-Workshop: Cluster Autoscaler

Karpenter

Karpenter ist ein Kubernetes-Knoten-Autoscaler, der auf Flexibilität, Leistung und Einfachheit ausgelegt ist.

- Projekt-URL
- Projekt-Dokumentation
- Amazon-EKS-Workshop: Karpenter

Escalator

Escalator ist ein Batch- oder Auftrag-optimiertes Tool zur automatischen horizontalen Skalierung für Kubernetes.

- Projekt-URL
- Projekt-Dokumentation

Überwachung

Verwandte Überwachungs-Projekte für Amazon EKS und Kubernetes-Cluster.

Prometheus

Prometheus ist ein Open-Source-Toolkit zur Überwachung und Benachrichtigung für Systeme.

- Projekt-URL
- Projekt-Dokumentation
- Amazon EKS-Workshop: https://eksworkshop.com/intermediate/240_monitoring/

Fortlaufende Integration/Fortlaufende Bereitstellung

Verwandte wichtige CI/CD Projekte für Amazon EKS- und Kubernetes-Cluster.

Jenkins X

CI/CD-Lösung für moderne Cloud-Anwendungen auf Amazon-EKS- und Kubernetes-Clustern.

- Projekt-URL
- Projekt-Dokumentation

Weitere Informationen zu neuen Features und der Roadmap von Amazon EKS

Sie können mehr über die neuen Amazon-EKS-Feature erfahren, indem Sie zum Feed „Neuheiten“ auf der Seite Neuheiten mit AWS gehen. Sie können auch die Roadmap auf GitHub ansehen, um sich über bevorstehende Funktionen und Prioritäten zu informieren, damit Sie planen können, wie Sie Amazon EKS in Zukunft nutzen möchten. Sie können uns direktes Feedback zu den Roadmap-Prioritäten geben.

Dokumentverlauf

In der folgenden Tabelle werden einige der wichtigsten Aktualisierungen und neuen Features des Amazon-EKS-Benutzerhandbuchs beschrieben. Um Benachrichtigungen zu erhalten, wenn diese Tabelle einen neuen Eintrag erhält, können Sie die folgende URL mit einem RSS-Reader abonnieren:

```
https://docs.aws.amazon.com/eks/latest/userguide/doc-history.rss
```

Änderung	Beschreibung	Datum
AWS verwaltete Richtlinienaktualisierungen	Es wurde die `ec2:LockSnapshot` Berechtigung hinzugefügt`AmazonEBSCSIDriverPolicy` , dem EBS-CSI-Treiber das direkte Sperren von EBS-Snapshots zu ermöglichen.	15. Januar 2026
Funktionen von Amazon EKS	Amazon EKS bietet jetzt vollständig verwaltete Cluster-Funktionen mit EKS-Funktionen, mit Unterstützung für deklarative kontinuierliche Bereitstellung mit Unterstützung für Argo CD, AWS Ressourcenmanagement mit Unterstützung für AWS Controller für Kubernetes und Kubernetes-Ressourcenzusammensetzung und Orchestrierung mit Unterstützung für Kube Resource Orchestrator.	30. November 2025
Neue AWS verwaltete Richtlinie	Amazon EKS hat eine neue verwaltete Richtlinie veröffent	21. November 2025

	lichtAmazonEKSMCPReadOnlyAccess , um schreibgeschützte Tools auf dem Amazon EKS MCP-Server für Beobachtbarkeit und Fehlerbehebung zu aktivieren. Weitere Informationen finden Sie unter Amazon EKS-Updates für AWS verwaltete Richtlinien.	
Beobachtbarkeit des Netzwerks	Amazon EKS bietet jetzt eine verbesserte Beobachtbarkeit von Containernetzwerken mit Leistungsüberwachung und Transparenz des Workload-Datenverkehrs.	19. November 2025
AWS verwaltete Richtlinienaktualisierungen	Es wurde die ec2:CopyVolumes Berechtigung AmazonEBSCSIDriverPolicy hinzugefügt, dem EBS-CSI-Treiber das direkte Kopieren von EBS-Volumes zu ermöglichen.	17. November 2025

AWS verwaltete Richtlinienaktualisierungen	Hinzugefügt `ec2:DescribeRouteTables` und `ec2:DescribeNetworkAcls` Berechtigungen für`AmazonEKSServiceRolePolicy` . Auf diese Weise kann Amazon EKS im Rahmen von Cluster-Erkenntnissen Konfigurationsprobleme mit VPC-Routing-Tabellen und Netzwerken ACLs für Hybridknoten erkennen.	22. Oktober 2025
Aktualisierung der dienstverknüpften Rolle	Die `ssmmessages:OpenDataChannel` Erlaubnis wurde hinzugefügt zu `AmazonEKSConnectorServiceRolePolicy`	15. Oktober 2025
Kubernetes-Version `1.34`	Support für Kubernetes-Version `1.34` für neue Cluster und Versions-Upgrades hinzugefügt.	2. Oktober 2025
Konfigurierbare automatische Knotenreparatur	Amazon EKS bietet nun eine differenzierte Kontrolle über das Verhalten der automatischen Knotenreparatur.	22. September 2025
Aktualisieren von Cluster-Erkenntnissen	Sie können Cluster-Erkenntnisse nun manuell aktualisieren.	27. August 2025

AWS verwaltete Richtlinienaktualisierungen	Berechtigung zu `AmazonEKSServiceRolePolicy` hinzugefügt. Diese Rolle kann die neue Zugriffsrichtlinie `AmazonEKSEventPolicy` anfügen. Eingeschränkte Berechtigungen für `ec2:DeleteLaunchTemplate` und `ec2:TerminateInstances` .	26. August 2025
Vermeidung des dienstübergreifenden Confused-Deputy-Problems	Thema mit einer Beispiel-Vertrauensrichtlinie hinzugefügt, die Sie für die serviceübergreifende Verhinderung von Angriffen durch verwirrte Stellvertreter anwenden können. Amazon EKS akzeptiert die `aws:SourceArn` - und `aws:SourceAccount` -Bedingungen in der Vertrauensrichtlinie einer EKS-Cluster-Rolle.	19. August 2025
Aktualisierung der Amazon-EKS-Plattformversion	Dies ist eine neue Plattformversion mit Sicherheitsfixes und -verbesserungen. Dies schließt neue Patch-Versionen von Kubernetes `1.33.2`, `1.32.6`, `1.31.10` und `1.30.14` mit ein.	30. Juli 2025

Feature von VPC CNI Multi NIC für Multi-Homed-Pods	Amazon EKS fügt dem VPC CNI Multi-Homed-Pods hinzu. Jetzt können Sie einen Workload und die vom VPC-CNI zugewiesenen IP-Adressen von jeder NIC auf der EC2 Instance zu jedem Pod konfigurieren. Die Anwendung kann gleichzeitige Verbindungen herstellen, um die Bandbreite jeder Netzwerkkarte zu nutzen. Jede Netzwerkschnittstelle ist in demselben Subnetz und denselben Sicherheitsgruppen wie der Knoten konfiguriert. Bisher mussten Sie Multus CNI verwenden, um mehrere andere zu verwenden, um mehrfach vernetzte Pods zu erstellen. CNIs Die Dokumentation und die entsprechenden Schritte wurden verschoben nach. https://docs.aws.amazon.com/eks/latest/userguide/pod-multus.html	15. Juli 2025
Inhaltsaktualisierung zur Fehlerbehebung bei VPC CNI	Die Seite zur Fehlerbehebung für die Netzwerkrichtlinie für Kubernetes im VPC CNI wurde erweitert. Die Berechtigungen CRDs und RBAC sowie 12 bekannte Probleme wurden hinzugefügt.	30. Juni 2025

Aktualisierung der Amazon-EKS-Plattformversion	Dies ist eine neue Plattformversion mit Sicherheitsfixes und -verbesserungen. In dieser Plattformversion gibt es keine neuen Patch-Versionen von Kubernetes.	26. Juni 2025
AWS verwaltete Richtlinienaktualisierungen	`ssmmessages:OpenDataChannel` -Berechtigung zu `AmazonEKSLocalOutpostServiceRolePolicy` hinzugefügt.	26. Juni 2025
AWS verwaltete Richtlinienaktualisierungen	Es wurden Berechtigungen hinzugefügt, `AmazonEKSServiceRolePolicy` `AmazonEKSComputePolicy` um Amazon EKS Auto Mode das Starten von Instances mithilfe der EC2 On-Demand-Kapazitätsreservierungen in Ihrem Konto zu ermöglichen. Außerdem wurden die Berechtigungen `AmazonEKSComputePolicy` für Amazon EKS hinzugefügt, um die mit dem EC2 Spot-Dienst verknüpfte Rolle in Ihrem Namen zu erstellen.	20. Juni 2025
Aktualisierung der verwalteten Richtlinien	Richtlinie `AmazonEKSDashboardConsoleReadOnly` hinzugefügt.	19. Juni 2025

Aktualisierung von Amazon EKS Auto Mode auf Version `NodeClass`	Die `NodeClass` -Vorlage für Knoten im Automatikmodus wurde um eine Konfiguration für separate Pod-Subnetze erweitert. Dadurch werden die optionalen Schlüssel `podSubnetSelectorTerms` und `podSecurityGroupSelectorTerms` hinzugefügt, um die Subnetze und Sicherheitsgruppen für die Pods festzulegen.	13. Juni 2025
Sekundäre und kontoüber greifende Rollen mit EKS-Pod-Identitäten ansteuern	Amazon EKS fügt Ziel-IAM-Rollen zu EKS-Pod-Identitäten hinzu, um eine automatisierte Rollenkette zu ermöglichen. Damit können Sie automatisch eine Rolle in einem anderen Konto übernehmen und EKS Pod Identity rotiert die temporären Anmeldeinformationen. Jede Pod-Identity-Zuordnung muss zunächst über eine IAM-Rolle im selben Konto verfügen, die sie übernehmen kann. Anschließend wird diese Rolle verwendet, um die Zielrolle zu übernehmen.	11. Juni 2025

Aktualisierung der Amazon-EKS-Plattformversion	Dies ist eine neue Plattformversion mit Sicherheitsfixes und -verbesserungen. In dieser Plattformversion gibt es keine neuen Patch-Versionen von Kubernetes.	11. Juni 2025
Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in der AWS Region Asien-Pazifik (Taipeh) (`ap-east-2`) verfügbar.	6. Juni 2025
`IPv6`-Zugriffskontrolle für Dual-Stack-Endpunkte für neue `IPv6`-Cluster	Amazon EKS fügt `IPv6`-CIDR-Blöcke hinzu, um den Zugriff auf den öffentlichen Cluster-Endpunkt für neue `IPv6`-Cluster zu steuern. Bisher konnten Sie nur `IPv4`-CIDR-Blöcke hinzufügen, um Datenverkehr zum öffentlichen Cluster-Endpunkt zuzulassen, selbst für Dual-Stack-Cluster-Endpunkte.	5. Juni 2025
Aktualisierung der Amazon-EKS-Plattformversion	Dies ist eine neue Plattformversion mit Sicherheitsfixes und -verbesserungen. Dies schließt neue Patch-Versionen von Kubernetes `1.32.5`, `1.31.9`, und `1.30.13` mit ein.	30. Mai 2025

Neue Cluster-Erkenntnisse für EKS-Hybridknoten	Amazon EKS fügt neue Cluster-Erkenntnisse hinzu, welche die Konfiguration Ihrer Hybridknoten überprüfen. Diese Erkenntnisprüfungen warnen Sie vor Problemen mit On-Premises-Knoten und Pods sowie der Netzwerkk onfiguration des Clusters aus der Ferne.	29. Mai 2025
Kubernetes-Version `1.33`	Support für Kubernetes-Version `1.33` für neue Cluster und Versions-Upgrades hinzugefügt.	29. Mai 2025
Zusätzliche Unterstützung für den Amazon FSx CSI-Treiber	Sie können jetzt die AWS Managementkonsole, AWS CLI und API verwenden, um den Amazon FSx CSI-Treiber zu verwalten.	23. Mai 2025
Prometheus-Scraper in der Konsole bearbeiten	Sie können jetzt Amazon Managed Service für Prometheus-Scraper in der Amazon-EKS-Konsole bearbeiten.	22. Mai 2025
Aktualisierung der verwalteten Richtlinien	Richtlinie `AmazonEKS DashboardServiceRo lePolicy` hinzugefügt.	21. Mai 2025

Aktualisierung der Amazon-EKS-Plattformversion	Dies ist eine neue Plattformversion mit Sicherheitsfixes und -verbesserungen. In diesen Plattformversionen sind keine neuen Patch-Versionen von Kubernetes enthalten.	16. Mai 2025
Neue Seiten für Amazon FSx for Lustre Performance	Es wurden neue Themen mit Details zur Optimierung der Leistung von Amazon FSx for Lustre hinzugefügt.	2. Mai 2025
Aktualisierung von Amazon EKS Auto Mode auf Version `NodeClass`	Die `NodeClass` -Vorlage für Knoten im Automatikmodus wurde um eine Konfiguration für die Weiterleitung von Netzwerk-Proxys erweitert. Dadurch wird der optionale Schlüssel `advancedNetworking` zum Festlegen Ihres HTTPS-Proxys hinzugefügt.	30. April 2025
Bottlerocket für Hybridknoten	Bottlerocket ist jetzt für EKS-Hybridknoten verfügbar.	29. April 2025
Aktualisierung der Amazon-EKS-Plattformversion	Dies ist eine neue Plattformversion mit Sicherheitsfixes und -verbesserungen. In diesen Plattformversionen sind keine neuen Patch-Versionen von Kubernetes enthalten.	29. April 2025

Neue Konzeptseiten für Hybrid-Netzwerke	Seiten für Konzepte zu EKS-Hybridknoten hinzugefügt. Diese behandeln detailliert die On-Premises- und Cloud-Vernetzung anhand von Diagrammen.	18. April 2025
AWS verwaltete Richtlini enaktualisierungen	Berechtigungen zu `AmazonEKSClusterPolicy` hinzugefügt, um Amazon EKS die Nutzung der vom VPC CNI erstellten elastischen Netzwerkschnittstellen zu ermöglichen. Dies ist erforderlich, damit EKS elastische Netzwerkschnittstellen bereinigen kann, die zurückbleiben, wenn das VPC CNI unerwartet beendet wird.	16. April 2025
AWS verwaltete Richtlini enaktualisierungen	Es wurden Berechtigungen hinzugefügt`AmazonEKSServiceRolePolicy` , um AI/ML EKS-Kunden das Hinzufügen von Ausgangsregeln zur standardmäßigen EKS-Cluster-Sicherheitsgruppe zu ermöglichen.	14. April 2025

Knotenintegrität für EKS-Hybridknoten	Sie können `eks-node-monitoring-agent` in Hybridknoten ab Version `1.2.0-eksbuild.1` verwenden. Führen Sie `eks-node-monitoring-agent` als Amazon EKS-Add-On aus, um Zustandsprobleme zu erkennen und anzuzeigen.	31. März 2025
EKS-Hybridknoten für vorhandene Cluster	Sie können nun die Hybridknoten-Konfiguration vorhandener Cluster hinzufügen, ändern oder entfernen. Bisher war es nur möglich, die Hybridknoten-Konfiguration neuen Clustern hinzuzufügen, wenn diese erstellt wurden. Mit Amazon EKS Hybrid Nodes können Sie Ihre lokale Infrastruktur und Ihre Edge-Infrastruktur als Knoten in Amazon EKS-Clustern verwenden. AWS verwaltet die AWS gehostete Kubernetes-Steuerebene des Amazon EKS-Clusters, und Sie verwalten die Hybridknoten, die in Ihren lokalen oder Edge-Umgebungen ausgeführt werden.	31. März 2025

Rollback: Vermeidung unbeabsichtigter Upgrades mithilfe von Cluster-Erkenntnissen	Amazon EKS hat vorübergehend das Feature zurückgesetzt, das bei bestimmten Problemen mit Cluster-Erkenntnissen die Verwendung eines `--force`-Flags für das Upgrade Ihres Clusters erforderlich gemacht hätte. Weitere Informationen finden Sie unter Vorübergehendes Rollback der Durchsetzung von Upgrade-Erkenntnissen bei der Aktualisierung der Cluster-Version auf. GitHub	28. März 2025
Bottlerocket FIPS AMIs	Bottlerocket FIPS AMIs sind jetzt in standardmäßigen verwalteten Knotengruppen verfügbar.	27. März 2025
AWS verwaltete Richtlinienaktualisierungen	Es wurden Berechtigungen hinzugefügt, `AmazonEKSServiceRolePolicy` um Amazon EKS das Beenden von EC2 Instances zu ermöglichen, die im automatischen Modus erstellt wurden. Es wurden Berechtigungen hinzugefügt, `AmazonEKSServiceRolePolicy` um Amazon EKS das Beenden von EC2 Instances zu ermöglichen, die im automatischen Modus erstellt wurden.	28. Februar 2025

Aktualisierungsstrategien für verwaltete Knotengruppen	Sie können nun mithilfe von Aktualisierungsstrategien den Versionsaktualisierungsprozess für verwaltete Knotengruppen konfigurieren. Damit wird die minimale Aktualisierungsstrategie eingeführt, bei der Knoten beendet werden, bevor neue erstellt werden. Dies ist in Umgebungen mit begrenzter Kapazität von Vorteil. Die Standard-Aktualisierungsstrategie behält das bisherige Verhalten bei.	27. Januar 2025
Kubernetes-Version `1.32`	Support für Kubernetes-Version `1.32` für neue Cluster und Versions-Upgrades hinzugefügt.	23. Januar 2025
Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in den AWS -Regionen Asien-Pazifik (Thailand) (`ap-southeast-7`) und Mexiko (Zentral) (`mx-central-1`) verfügbar . EKS Auto Mode und die VPC-Endpunkte für die EKS-API sind in keiner der beiden Regionen verfügbar.	14. Januar 2025

AWS verwaltete Richtlinienaktualisierungen	Mehrere Berechtigungen zu `AmazonEBSCSIDriverPolicy` hinzugefügt, damit der Amazon EBS CSI-Treiber alle Snapshots wiederherstellen, Fast Snapshot Restore (FSR) auf EBS-Volumes aktivieren und Tags auf Volumes ändern kann.	13. Januar 2025
AWS verwaltete Richtlinienaktualisierungen	Berechtigungen zu `AmazonEKSLoadBalancingPolicy` hinzugefügt.	26. Dezember 2024
Aktualisierte Cluster-Erkenntnisse	Amazon EKS Upgrade Insights warnt nun vor weiteren Problemen hinsichtlich des Cluster-Zustands und der Versionskompatibilität. Es kann Probleme zwischen verschiedenen Kubernetes- und Amazon-EKS-Komponenten wie `kubelet`, `kube-proxy` und Amazon-EKS-Add-Ons erkennen.	20. Dezember 2024
Knoten-Überwachungsagent und automatische Reparatur	Sie können das neue `eks-node-monitoring-agent` als Amazon-EKS-Add-On verwenden, um Zustandsprobleme zu erkennen und anzuzeigen. Sie können auch die automatische Knotenreparatur aktivieren, um Knoten automatisch zu ersetzen, wenn Probleme erkannt werden.	16. Dezember 2024

Amazon EKS Hybrid Nodes	Sie können nun Knoten, die mit Amazon-EKS-Clustern verbunden sind, On-Premises ausführen. Mit Amazon EKS Hybrid Nodes können Sie Ihre lokale Infrastruktur und Ihre Edge-Infrastruktur als Knoten in Amazon EKS-Clustern verwenden. AWS verwaltet die AWS gehostete Kubernetes-Steuerebene des Amazon EKS-Clusters, und Sie verwalten die Hybridknoten, die in Ihren lokalen oder Edge-Umgebungen ausgeführt werden.	01. Dezember 2024
Amazon EKS Auto Mode	Amazon EKS Auto Mode automatisiert vollständig die Verwaltung der Kubernetes-Cluster-Infrastruktur für Rechenleistung, Speicher und Netzwerk in AWS. Es vereinfacht das Kubernetes-Management durch die automatische Bereitstellung der Infrastruktur, die Auswahl optimaler Recheninstanzen, die dynamische Skalierung von Ressourcen, die kontinuierliche Optimierung der Kosten, das Patchen von Betriebssystemen und die Integration mit Sicherheitsdiensten. AWS	01. Dezember 2024

Aktualisierung der Amazon-EKS-Plattformversion	Dies ist eine neue Plattformversion mit Sicherheitsfixes und -verbesserungen. Dies schließt neue Patch-Versionen von Kubernetes `1.31.2`, `1.30.6`, `1.29.10` und `1.28.15` mit ein.	22. November 2024
AWS verwaltete Richtlinienaktualisierungen	`AWSServiceRoleForAmazonEKSNodegroup` für Kompatibilität mit chinesischen Regionen aktualisiert.	22. November 2024
Kubernetes Version 1.30 ist jetzt für lokale Cluster auf Outposts verfügbar AWS	Sie können jetzt mit Kubernetes Version 1.30 einen lokalen Amazon EKS-Cluster auf einem AWS Outposts erstellen.	21. November 2024
AWS verwaltete Richtlinienaktualisierungen	EKS hat die AWS verwaltete Richtlinie aktualisiertAmazonEKSLocalOutpostClusterPolicy . Es wurde eine `ec2:DescribeAvailabilityZones` Berechtigung hinzugefügt, damit der AWS Cloud Controller Manager auf der Cluster-Steuerebene die Availability Zone identifizieren kann, in der sich jeder Knoten befindet.	21. November 2024

Bottlerocket, die FIPS AMIs 140-3 verwenden	Bottlerocket AMIs sind verfügbar, die für die Verwendung von FIPS 140-3-validierten kryptografischen Modulen vorkonfiguriert sind. Dazu gehören das Amazon Linux 2023 Kernel Cryptographic API Cryptographic Module und das AWS-LC Cryptographic Module.	20. November 2024
AWS verwaltete Richtlinienaktualisierungen	Die `AWSServiceRoleForAmazonEKSNodegroup` - Richtlinie wurde so aktualisiert, dass Instances, die von Amazon EKS verwalteten Knotengruppen erstellt wurden, `ec2:RebootInstances` zulassen können. Die `ec2:CreateTags` Berechtigungen für EC2 Amazon-Ressourcen wurden eingeschränkt.	20. November 2024
Dashboard zur Beobachtbarkeit	Das Dashboard zur Beobachtbarkeit unterstützt Sie dabei, Probleme schnell zu erkennen, zu beheben und zu lösen. Im `AWS/EKS` Namespace sind auch neue CloudWatch Verkaufsmetriken verfügbar.	18. November 2024

AWS verwaltete Richtlinienaktualisierungen	EKS hat die AWS verwaltete Richtlinie aktualisiertAmazonEKSServiceRolePolicy . Berechtigungen für EKS-Zugriffsrichtlinien, Load-Balancer-Verwaltung und automatische Bereinigung von Cluster-Ressourcen hinzugefügt.	16. November 2024
Neue Rollenerstellung in der Konsole für Add-Ons, die EKS Pod Identities unterstützen	Bei der Verwendung der Konsole zum Erstellen oder Aktualisieren von Add-Ons, die EKS-Pod-Identities unterstützen, gibt es neue Schritte. Diese ermöglichen die automatische Generierung von IAM-Rollen mit dem entsprechenden Namen, der entsprechenden Rollenrichtlinie und der entsprechenden Vertrauensrichtlinie für das Add-On.	15. November 2024
Verwaltete Knotengruppen in AWS Local Zones	Verwaltete Knotengruppen können jetzt in AWS Local Zones erstellt werden.	15. November 2024
Es sind neue Metriken verfügbar	Unter der API-Gruppe metrics.eks.amazonaws.com sind neue Metriken verfügbar.	11. November 2024

AWS verwaltete Richtlinienaktualisierungen	EKS hat die AWS verwaltete Richtlinie aktualisiertAmazonEKSComputePolicy . Aktualisierte Ressourcenberechtigungen für die Aktion iam:AddRoleToInstanceProfile .	7. November 2024
AWS verwaltete Richtlinienaktualisierungen	EKS hat eine neue AWS verwaltete Richtlinie hinzugefügt: AmazonEKSComputePolicy	1. November 2024
AWS verwaltete Richtlinienaktualisierungen	Berechtigungen zu AmazonEKSClusterPolicy hinzugefügt. Berechtigung ec2:DescribeInstanceTopology hinzugefügt, damit Amazon EKS Topologieinformationen als Beschriftungen an den Knoten anfügen kann.	1. November 2024
AWS verwaltete Richtlinienaktualisierungen	EKS hat eine neue AWS verwaltete Richtlinie hinzugefügt: AmazonEKSBlockStoragePolicy	30. Oktober 2024
AWS verwaltete Richtlinienaktualisierungen	EKS hat eine neue AWS verwaltete Richtlinie hinzugefügt: AmazonEKSLoadBalancingPolicy	30. Oktober 2024

AWS verwaltete Richtlinienaktualisierungen	Es wurden `cloudwatch:PutMetricData` Berechtigungen hinzugefügt, `AmazonEKSServiceRolePolicy` die es Amazon EKS ermöglichen, Metriken auf Amazon zu veröffentlichen CloudWatch.	29. Oktober 2024
AWS verwaltete Richtlinienaktualisierungen	EKS hat eine neue AWS verwaltete Richtlinie hinzugefügt: `AmazonEKSNetworkingPolicy`	28. Oktober 2024
Dual-Stack-Endpunkte für neue `IPv6`-Cluster	Stellen Sie eine Verbindung zu neuen `IPv6`-Clustern mit einem Dual-Stack-Endpunkt `eks-cluster.` *`region`*`.api.aws` her. Dieser Endpunkt wird zurückgegeben, wenn Sie diese Cluster beschreiben. `kubectl` und andere Kubernetes-API-Clients in `IPv4`-, `IPv6`- oder Dual-Stack-Umgebungen können diese Endpunkte für öffentliche oder private Cluster auflösen und eine Verbindung zu ihnen herstellen.	21. Oktober 2024

AWS verwaltete Richtlinienaktualisierungen	`autoscaling:ResumeProcesses` , `autoscaling:SuspendProcesses` und zugehörige Berechtigungen zu `AWSServiceRoleForAmazonEKSNodegroup` in China-Regionen hinzugefügt, um eine Integration mit Amazon Application Recovery Controller für EKS zu ermöglichen. Keine Änderungen für andere Regionen.	21. Oktober 2024
AL2023 beschleunigt AMIs	Sie können jetzt beschleunigte `NVIDIA` und AWS Neuron-Instances für verwenden, die auf AMIs 023 basieren. AL2	11. Oktober 2024
Neues Quellformat	Wir sind auf ein neues Quellformat mit einigen Layoutänderungen umgestiegen. Es gibt vorübergehend kleinere Formatierungsprobleme, die wir beheben.	10. Oktober 2024

AWS verwaltete Richtlinienaktualisierungen	Berechtigungen für `AmazonEKSServicePolicy` und `AmazonEKSServiceRolePolicy` hinzugefügt. `ec2:GetSecurityGroupsForVpc` und zugehörige Tag-Berechtigungen hinzugefügt, damit EKS Sicherheitsgruppeninformationen lesen und zugehörige Tags aktualisieren kann.	10. Oktober 2024
AWS verwaltete Richtlinienaktualisierungen — Neue Richtlinie	EKS hat eine neue AWS verwaltete Richtlinie hinzugefügt.	3. Oktober 2024
Kubernetes-Version `1.31`	Support für Kubernetes-Version `1.31` für neue Cluster und Versions-Upgrades hinzugefügt.	24. September 2024
AWS verwaltete Richtlinienaktualisierungen — Aktualisierung einer bestehenden Richtlinie	Amazon EKS hat eine bestehende AWS verwaltete Richtlinie aktualisiert.	21. August 2024
Kubernetes Version 1.29 ist jetzt für lokale Cluster auf Outposts verfügbar AWS	Sie können jetzt mit Kubernetes Version 1.29 einen lokalen Amazon EKS-Cluster auf einem AWS Outposts erstellen.	20. August 2024

EKS-Pod-Identität in (USA) AWS GovCloud	Amazon EKS Pod Identities verknüpfen eine IAM-Rolle mit einem Kubernetes-Servicekonto. Mit diesem Feature müssen Sie der Knoten-IAM-Rolle keine erweiterten Berechtigungen mehr bereitstellen. Auf diese Weise können Pods auf diesem Knoten anrufen AWS APIs. Im Gegensatz zu IAM-Rollen für Servicekonten befinden sich EKS Pod Identities vollständig in EKS. Sie benötigen keinen OIDC-Identitätsanbieter.	14. August 2024
Szenariobasierte Inhaltsaktualisierungen	Wir haben die Themen umbenannt und aktualisiert, um sie im gesamten Leitfaden stärker an konkreten Szenarien auszurichten.	9. August 2024
Dual-Stack-VPC-Schnittstellenendpunkte für Amazon EKS	Sie können jetzt Dual-Stack-VPC-Schnittstellenendpunkte für Amazon EKS mit `IPv4`- und `IPv6`-IP-Adressen und DNS-Namen erstellen.	7. August 2024

Neue Dual-Stack-Endpunkte für Amazon EKS APIs mit Adressen IPv6	Die EKS-API für die Erstellung und Verwaltung von Clustern und der OIDC-Issuer URLs für Cluster verfügen über neue Dual-Stack-Endpunkte. Der neue DNS-Name für die Amazon-EKS-API lautet `eks.region.api.aws` und wird in IPv4-Adressen und IPv6-Adressen aufgelöst. Neue Cluster haben eine neue Dual-Stack-OIDC-Aussteller-URL (`oidc-eks.region.api.aws`).	1. August 2024
Kapazitätsblöcke für verwaltete Knotengruppen	Sie können nun Kapazitätsblöcke für verwaltete Knotengruppen verwenden.	1. Juli 2024
Die Erfassung von Metriken der Auto-Scaling-Gruppe ist standardmäßig aktiviert	Für von Amazon EKS verwaltete Knotengruppen sind jetzt standardmäßig Amazon EC2 Auto Scaling Scaling-Gruppenmetriken ohne zusätzliche Kosten aktiviert. Bisher waren mehrere Schritte erforderlich, um dieses Feature zu aktivieren.	28. Juni 2024
AWS verwaltete Richtlinienaktualisierungen — Aktualisierung einer bestehenden Richtlinie	Amazon EKS hat eine bestehende AWS verwaltete Richtlinie aktualisiert.	27. Juni 2024
Kubernetes-Version `1.26`	Die Kubernetes-Version `1.26` wird jetzt erweitert unterstützt.	12. Juni 2024

Verbesserungen an den AMI-Informationsreferenzen	Wir haben Verbesserungen an den AMI-Informationsreferenzen vorgenommen, insbesondere für Bottlerocket.	12. Juni 2024
Kubernetes-Version `1.30`	Support für Kubernetes-Version `1.30` für neue Cluster und Versions-Upgrades hinzugefügt.	23. Mai 2024
CoreDNS Autoscaling	Der CoreDNS-Autoscaler passt die Anzahl der Replikate der CoreDNS-Bereitstellung in einem EKS-Cluster dynamisch an die Anzahl der Knoten und CPU-Kerne an. Dieses Feature ist für CoreDNS `v1.9` und die neueste Plattform version der EKS-Version `1.25` und höher verfügbar.	14. Mai 2024
Aktualisierung der Amazon-EKS-Plattformversion	Dies ist eine neue Plattform version mit Sicherheitsfixes und -verbesserungen. Dies schließt neue Patch-Versionen von Kubernetes `1.29.4`, `1.28.9`, und `1.27.13` mit ein.	14. Mai 2024
CloudWatch Container Insights-Unterstützung für Windows	Das Amazon CloudWatch Observability Operator-Add-on ermöglicht jetzt auch Container Insights auf Windows-Worker-Knoten im Cluster.	10. April 2024

Kubernetes-Konzepte	Neues Thema zu Kubernetes-Konzepten hinzugefügt.	5. April 2024
Zugriff und IAM-Inhalte neu strukturieren	Verschieben Sie vorhandene Seiten zu Zugriffs- und IAM-Themen, wie z. B. Konfigurationszuordnung für die Authentifizierung, Zugriffseinträge, Pod-ID und IRSA, in einen neuen Abschnitt. Überprüfen Sie den Inhalt der Übersicht.	2. April 2024
Bottlerocket-OS-Support für Amazon-S3-CSI-Treiber	Der CSI-Treiber für Mountpoint für Amazon S3 ist nun mit Bottlerocket kompatibel.	13. März 2024
AWS verwaltete Richtlinienaktualisierungen — Aktualisierung einer bestehenden Richtlinie	Amazon EKS hat eine bestehende AWS verwaltete Richtlinie aktualisiert.	4. März 2024
Amazon Linux 2023	Amazon Linux 2023 (AL2023) ist ein neues Linux-basiertes Betriebssystem, das eine sichere, stabile und leistungsstarke Umgebung für Ihre Cloud-Anwendungen bietet.	29. Februar 2024

EKS Pod Identity und IRSA unterstützen Sidecars in Kubernetes 1.29	In Kubernetes 1.29 sind Sidecar-Container in Amazon-EKS-Clustern verfügbar. Sidecar-Container werden mit IAM-Rollen für Servicekonten oder EKS Pod Identity unterstützt. Weitere Informationen zu Sidecars finden Sie unter Sidecar-Container in der Kubernetes-Dokumentation.	26. Februar 2024
Kubernetes-Version 1.29	Support für Kubernetes-Version 1.29 für neue Cluster und Versions-Upgrades hinzugefügt.	23. Januar 2024
Vorversion: Erweiterter Support von Amazon EKS für Kubernetes-Versionen	Dank der erweiterten Kubernetes-Versionsunterstützung können Sie länger als 14 Monate bei einer bestimmten Kubernetes-Version bleiben.	16. Januar 2024

Zustandserkennung von Amazon EKS-Clustern in der AWS Cloud	Amazon EKS erkennt Probleme mit Ihren Amazon-EKS-Clustern und der Infrastruktur der Cluster-Voraussetzungen in Cluster-Integrität . Sie können die Probleme mit Ihren EKS-Clustern im AWS-Managementkonsole und im Cluster in `health` der EKS-API anzeigen. Diese Probleme werden zusätzlich zu den Problemen erkannt, die von der Konsole erkannt und angezeigt werden. Bisher war die Clusterintegrität nur für lokale Cluster auf AWS Outposts verfügbar.	28. Dezember 2023
Cluster-Erkenntnisse	Sie können jetzt Empfehlungen für Ihren Cluster erhalten, die auf wiederkehrenden Prüfungen basieren.	20. Dezember 2023
Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in der AWS Region Kanada West (Calgary) (`ca-west-1`) verfügbar.	20. Dezember 2023

Sie können jetzt mithilfe von Zugriffseinträgen IAM-Rollen und -Benutzern Zugriff auf Ihren Cluster gewähren.	Vor der Einführung von Zugriffseinträgen haben Sie IAM-Rollen und -Benutzern Zugriff auf Ihren Cluster gewährt, indem Sie Einträge zu `aws-auth ConfigMap` hinzugefügt haben. Jetzt verfügt jeder Cluster über einen Zugriffsmodus und Sie können nach Ihrem eigenen Zeitplan auf die Verwendung von Zugriffseinträgen umsteigen. Nachdem Sie den Modus gewechselt haben, können Sie Benutzer hinzufügen, indem Sie Zugriffseinträge in der AWS CLI AWS CloudFormation, und dem hinzufügen AWS SDKs.	18. Dezember 2023
Aktualisierung der Amazon-EKS-Plattformversion	Dies ist eine neue Plattformversion mit Sicherheitsfixes und -verbesserungen. Dies schließt neue Patch-Versionen von Kubernetes `1.28.4`, `1.27.8`, `1.26.11` und `1.25.16` mit ein.	12. Dezember 2023
CSI-Treiber für Mountpoint für Amazon S3	Sie können jetzt den CSI-Treiber für Mountpoint für Amazon S3 in Amazon-EKS-Clustern installieren.	27. November 2023

Prometheus-Metriken beim Erstellen eines Clusters aktivieren	In der AWS-Managementkonsole können Sie jetzt Prometheus-Metriken aktivieren, wenn Sie einen Cluster erstellen. Sie können die Prometheus-Scraper-Details auch auf der Registerkarte Beobachtbarkeit anzeigen.	26. November 2023
Amazon-EKS-Pod-Identitäten	Amazon EKS Pod Identities verknüpfen eine IAM-Rolle mit einem Kubernetes-Servicekonto. Mit diesem Feature müssen Sie der Knoten-IAM-Rolle keine erweiterten Berechtigungen mehr bereitstellen. Auf diese Weise können Pods auf diesem Knoten aufrufen. AWS APIs Im Gegensatz zu IAM-Rollen für Servicekonten befinden sich EKS Pod Identities vollständig in EKS. Sie benötigen keinen OIDC-Identitätsanbieter.	26. November 2023
AWS verwaltete Richtlinienaktualisierungen — Aktualisierung einer bestehenden Richtlinie	Amazon EKS hat eine bestehende AWS verwaltete Richtlinie aktualisiert.	26. November 2023
CSI-Snapshot-Controller	Sie können den CSI-Snapshot-Controller jetzt für die Verwendung mit kompatiblen CSI-Treibern wie dem Amazon-EBS-CSI-Treiber installieren.	17. November 2023

Neuverfassung des Themas „ADOT Operator“	Der Bereich Amazon EKS-Add-On-Unterstützung für ADOT Operator war zusammen mit der AWS Distribution zur OpenTelemetry Dokumentation überflüssig. Wir haben die verbleibenden wichtigen Informationen auf diese Ressource migriert, um veraltete und inkonsistente Informationen zu reduzieren.	14. November 2023
CoreDNS-EKS-Add-On-Support für Prometheus-Metriken	Die Versionen `v1.10.1-eksbuild.5` , `v1.9.3-eksbuild.9` und `v1.8.7-eksbuild.8` des EKS-Add-ons für CoreDNS legen den Port, über den CoreDNS Metriken veröffentlicht hat, im `kube-dns`-Service offen. Dies macht es einfacher, die CoreDNS-Metriken in Ihre Überwachungssysteme aufzunehmen.	10. November 2023
Amazon EKS CloudWatch Observability Operator-Add-on	Die Amazon EKS CloudWatch Observability Operator-Seite wurde hinzugefügt.	6. November 2023
Kapazitätsblöcke für selbstverwaltete P5-Instances in USA Ost (Ohio)	In USA Ost (Ohio) können Sie Kapazitätsblöcke jetzt für selbstverwaltete P5-Instances verwenden.	31. Oktober 2023

Cluster unterstützen das Ändern von Subnetzen und Sicherheitsgruppen	Sie können den Cluster aktualisieren, um zu ändern, welche Subnetze und Sicherheitsgruppen der Cluster verwendet. Sie können von der AWS-Manag ementkonsole, der neuesten Version der AWS CLI AWS CloudFormation, und `eksctl` Version `v0.164.0-rc.0` oder höher aus aktualisieren. Möglicherweise müssen Sie dies tun, um Subnetzen mehr verfügbare IP-Adressen zur Verfügung zu stellen, damit eine Clusterversion erfolgreich aktualisiert werden kann.	24. Oktober 2023
Die Clusterrolle und die Rolle der verwalteten Knotengru ppe unterstützen vom Kunden verwaltete AWS Identity and Access Management Zugriffsv erwaltungsrichtlinien	Sie können anstelle der von Amazon Policy AWS verwaltet en Richtlinie eine benutzerd efinierte EKSCluster IAM-Richtlinie für die Cluster-Rolle verwenden. Außerdem können Sie anstelle der von Amazon verwalteten Richtlinie eine benutzerdefinierte IAM-Richt linie für die Knotenrolle in einer EKSWorker NodePolicy AWS verwalteten Knotengruppe verwenden. Auf diese Weise können Sie eine Richtlinie mit der geringsten Berechtig ung erstellen, um strenge Compliance-Anforderungen zu erfüllen.	23. Oktober 2023

Link zur Installation von eksctl korrigiert	Korrigiert den Installationslink für eksctl, nachdem die Seite verschoben wurde.	06. Oktober 2023
Vorversion: Erweiterter Amazon-EKS-Support für Kubernetes-Versionen	Dank der erweiterten Kubernetes-Versionsunterstützung können Sie länger als 14 Monate bei einer bestimmten Kubernetes-Version bleiben.	04. Oktober 2023
Verweise auf die AWS App Mesh Mesh-Integration entfernen	Amazon EKS-Integrationen mit AWS App Mesh stehen weiterhin nur bestehenden Kunden von App Mesh zur Verfügung.	29. September 2023
Kubernetes-Version `1.28`	Support für Kubernetes-Version `1.28` für neue Cluster und Versions-Upgrades hinzugefügt.	26. September 2023
Vorhandene Cluster unterstützen die Durchsetzung von Kubernetes-Netzwerkrichtlinien im Amazon-VPC-CNI-Plugin für Kubernetes	Sie können die Kubernetes-Netzwerkrichtlinie in vorhandenen Clustern mit dem Amazon-VPC-CNI-Plugin für Kubernetes verwenden, anstatt eine Drittanbieterlösung zu benötigen. Sie können die Kubernetes-Netzwerkrichtlinie in vorhandenen Clustern mit dem Amazon-VPC-CNI-Plugin für Kubernetes verwenden, anstatt eine Drittanbieterlösung zu benötigen.	15. September 2023

Amazon-EKS-Add-On für CoreDNS unterstützt die Änderung von PDB	Sie können das `PodDisruptionBudget` des EKS-Add-Ons für CoreDNS in Versionen `v1.9.3-eksbuild.7` und höher sowie `v1.10.1-eksbuild.4` und höher ändern.	15. September 2023
Amazon-EKS-Unterstützung für gemeinsam genutzte Subnetze	Neue Anforderungen und Überlegungen für gemeinsam genutzte Subnetze für die Erstellung von Amazon-EKS-Clustern in gemeinsam genutzten Subnetzen.	07. September 2023
Aktualisierungen an „Was ist Amazon EKS?“	Neue Themen in Häufige Anwendungsfälle und Architektur hinzugefügt. Andere Themen wurden aktualisiert.	6. September 2023
Durchsetzung der Kubernetes-Netzwerkrichtlinien im Amazon-VPC-CNI-Plugin für Kubernetes	Sie können die Kubernetes-Netzwerkrichtlinie mit dem Amazon-VPC-CNI-Plugin für Kubernetes verwenden, anstatt eine Drittanbieterlösung zu benötigen. Sie können die Kubernetes-Netzwerkrichtlinie mit dem Amazon-VPC-CNI-Plugin für Kubernetes verwenden, anstatt eine Drittanbieterlösung zu benötigen.	29. August 2023
Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in der AWS Region Israel (Tel Aviv) (`il-central-1`) verfügbar.	1. August 2023

Konfigurierbarer flüchtiger Speicher für Fargate	Sie können die Gesamtmenge des flüchtigen Speichers für jeden Pod erhöhen, der in Amazon EKS Fargate ausgeführt wird.	31. Juli 2023
Add-on-Support für Amazon-EFS-CSI-Treiber	Sie können jetzt die AWS CLI und die AWS-Managementkonsole API verwenden, um den Amazon EFS CSI-Treiber zu verwalten.	26. Juli 2023
AWS verwaltete Richtlinienaktualisierungen — Neue Richtlinie	Amazon EKS hat eine neue AWS verwaltete Richtlinie hinzugefügt.	26. Juli 2023
Kubernetes-Versionsupdates für 1.27, 1.26, 1.25 und 1.24 sind jetzt für lokale Cluster auf Outposts verfügbar AWS	Kubernetes-Versionsupdates auf 1.27.3, 1.26.6, 1.25.11 und 1.24.15 sind jetzt für lokale Cluster auf Outposts verfügbar AWS	20. Juli 2023
Unterstützung von IP-Präfixen für Windows-Knoten	Wenn Sie Ihren Knoten IP-Präfixe zuweisen, können Sie auf Ihren Knoten eine deutlich höhere Anzahl von Pods hosten, als wenn Sie Ihren Knoten einzelne sekundäre IP-Adressen zuweisen.	6. Juli 2023
Amazon FSx für OpenZFS CSI-Treiber	Sie können jetzt den Amazon FSx for OpenZFS CSI-Treiber auf Amazon EKS-Clustern installieren.	30. Juni 2023

Pods auf Linux-Knoten in IPv4-Clustern können jetzt mit IPv6-Endpunkten kommunizieren.	Nachdem Sie Ihrem Knoten eine IPv6 Adresse zugewiesen haben, ist die Adresse Ihres Pods eine IPv4 Netzwerkadresse, die in die IPv6 Adresse des Knotens übersetzt wird, auf dem er läuft.	19. Juni 2023
Von Windows verwaltete Knotengruppen in Regionen AWS GovCloud (USA)	In den Regionen AWS GovCloud (USA) können verwaltete Amazon EKS-Knotengruppen jetzt Windows-Container ausführen.	30. Mai 2023
Kubernetes-Version 1.27	Support für Kubernetes-Version 1.27 für neue Cluster und Versions-Upgrades hinzugefügt.	24. Mai 2023
Kubernetes-Version 1.26	Support für Kubernetes-Version 1.26 für neue Cluster und Versions-Upgrades hinzugefügt.	11. April 2023
Domainlose gMSA	Sie können nun domainlose gMSA mit Windows Pods verwenden.	27. März 2023
Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in der AWS Region Asien-Pazifik (Melbourne) (ap-southeast-4) verfügbar.	10. März 2023
CSI-Treiber für Amazon File Cache	Sie können jetzt den CSI-Treiber für Amazon File Cache in Amazon-EKS-Clustern installieren.	3. März 2023

Kubernetes Version 1.25 ist jetzt für lokale Cluster auf Outposts verfügbar AWS	Sie können jetzt einen lokalen Amazon-EKS-Cluster in einem Outpost mit Kubernetes-Versionen `1.22` – `1.25` erstellen.	1. März 2023
Kubernetes-Version `1.25`	Support für Kubernetes-Version `1.25` für neue Cluster und Versions-Upgrades hinzugefügt.	22. Februar 2023
AWS verwaltete Richtlinienaktualisierungen — Aktualisierung einer bestehenden Richtlinie	Amazon EKS hat eine bestehende AWS verwaltete Richtlinie aktualisiert.	07. Februar 2023
Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in den AWS Regionen Asien-Pazifik (Hyderabad`ap-south-2`) (), Europa (Zürich) (`eu-central-2`) und Europa (Spanien) (`eu-south-2`) verfügbar.	06. Februar 2023
Kubernetes-Versionen `1.21` — `1.24` sind jetzt für lokale Cluster auf AWS Outposts verfügbar.	Sie können jetzt einen lokalen Amazon-EKS-Cluster in einem Outpost mit Kubernetes-Versionen `1.21` – `1.24` erstellen. Bisher war nur die Version `1.21` verfügbar.	17. Januar 2023
Amazon EKS unterstützt jetzt AWS PrivateLink	Sie können eine verwenden AWS PrivateLink , um eine private Verbindung zwischen Ihrer VPC und Amazon EKS herzustellen.	16. Dezember 2022

Windows-Unterstützung für verwaltete Knotengruppen	Sie können jetzt Windows für von Amazon EKS verwaltete Knotengruppen verwenden.	15. Dezember 2022
Amazon EKS-Add-Ons von unabhängigen Softwareanbietern sind jetzt im AWS Marketplace erhältlich	Sie können jetzt Amazon EKS-Add-Ons von unabhängigen Softwareanbietern über den AWS Marketplace durchsuchen und abonnieren.	28. November 2022
AWS verwaltete Richtlinienaktualisierungen — Aktualisierung einer bestehenden Richtlinie	Amazon EKS hat eine bestehende AWS verwaltete Richtlinie aktualisiert.	17. November 2022
Kubernetes-Version `1.24`	Support für Kubernetes-Version `1.24` für neue Cluster und Versions-Upgrades hinzugefügt.	15. November 2022
Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in der AWS Region Naher Osten (VAE) (`me-central-1`) verfügbar.	03. November 2022
AWS verwaltete Richtlinienaktualisierungen — Aktualisierung einer bestehenden Richtlinie	Amazon EKS hat eine bestehende AWS verwaltete Richtlinie aktualisiert.	24. Oktober 2022
AWS verwaltete Richtlinienaktualisierungen — Aktualisierung einer bestehenden Richtlinie	Amazon EKS hat eine bestehende AWS verwaltete Richtlinie aktualisiert.	20. Oktober 2022
Lokale Cluster auf AWS Outposts sind jetzt verfügbar	Sie können jetzt einen lokalen Amazon-EKS-Cluster auf einem Outpost erstellen.	19. September 2022

vCPU-basierte Kontingente in Fargate	Fargate wird von Pod-basierten Kontingenten auf vCPU-basierte Kontingente umgestellt.	8. September 2022
AWS verwaltete Richtlinienaktualisierungen — Aktualisierung einer bestehenden Richtlinie	Amazon EKS hat eine bestehende AWS verwaltete Richtlinie aktualisiert.	31. August 2022
Kostenüberwachung	Amazon EKS unterstützt jetzt Kubecost, wodurch Sie die Kosten aufgeschlüsselt nach Kubernetes-Ressourcen, einschließlich Pods, Knoten, Namespaces und Labels, überwachen können.	24. August 2022
AWS verwaltete Richtlinienaktualisierungen — Neue Richtlinie	Amazon EKS hat eine neue AWS verwaltete Richtlinie hinzugefügt.	24. August 2022
AWS verwaltete Richtlinienaktualisierungen — Neue Richtlinie	Amazon EKS hat eine neue AWS verwaltete Richtlinie hinzugefügt.	23. August 2022
Markieren von Ressourcen für die Fakturierung	Unterstützung für ein von `aws:eks:cluster-name` generiertes Kostenzuweisungs-Tag wurde für alle Cluster hinzugefügt.	16. August 2022
Fargate-Profil-Platzhalter	Unterstützung für Fargate-Profil-Platzhalter in den Selektorkriterien für Namespaces, Labelschlüssel und Labelwerte wurde hinzugefügt.	16. August 2022

Kubernetes-Version `1.23`	Support für Kubernetes-Version `1.23` für neue Cluster und Versions-Upgrades hinzugefügt.	11. August 2022
Sehen Sie sich die Kubernetes-Ressourcen an im AWS-Managementkonsole	Sie können jetzt Informationen über Kubernetes-Ressourcen, die in Ihrem Cluster bereitgestellt sind, in der AWS-Managementkonsole anzeigen.	3. Mai 2022
Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in der AWS Region Asien-Pazifik (Jakarta) (`ap-southeast-3`) verfügbar.	2. Mai 2022
Seite „Beobachtbarkeit“ und ADOT-Add-on-Unterstützung	Die Observability-Seite und die AWS Distribution für OpenTelemetry (ADOT) wurden hinzugefügt.	21. April 2022
Kubernetes-Version `1.22`	Support für Kubernetes-Version `1.22` für neue Cluster und Versions-Upgrades hinzugefügt.	4. April 2022
AWS verwaltete Richtlinienaktualisierungen — Neue Richtlinie	Amazon EKS hat eine neue AWS verwaltete Richtlinie hinzugefügt.	4. April 2022

Details zum Fargate-Pod-Patching hinzugefügt	Beim Upgrade von Fargate-Pods versucht Amazon EKS zunächst, Pods basierend auf Ihren Budgets für Pod-Unterbrechungen zu vertreiben. Sie können Ereignisregeln erstellen, um auf fehlgeschlagene Bereinigungen zu reagieren, bevor die Pods gelöscht werden.	1. April 2022
Vollständiger Release: Add-on-Support für Amazon-EBS-CSI-Treiber	Sie können jetzt die AWS CLI und die AWS-Management konsole API verwenden, um den Amazon EBS CSI-Treiber zu verwalten.	31. März 2022
AWS Aktualisierung der Inhalte von Outposts	Anweisungen zur Bereitstellung eines Amazon EKS-Clusters auf AWS Outposts.	22. März 2022
AWS verwaltete Richtlinienaktualisierungen — Aktualisierung einer bestehenden Richtlinie	Amazon EKS hat eine bestehende AWS verwaltete Richtlinie aktualisiert.	21. März 2022
Windows `containerd` -Unterstützung	Sie können jetzt die `containerd` -Laufzeit für Windows-Knoten auswählen.	14. März 2022
Überlegungen zum Amazon EKS Connector zur Sicherheitsdokumentation hinzugefügt	Beschreibt das Modell der gemeinsamen Verantwortung in Bezug auf verbundene Cluster.	25. Februar 2022

Pods und Services IPv6-Adressen zuweisen	Sie können jetzt einen Cluster mit 1.21 oder höher erstellen, der Ihren Pods und Services IPv6-Adressen zuweist.	6. Januar 2022
AWS verwaltete Richtlinienaktualisierungen — Aktualisierung einer bestehenden Richtlinie	Amazon EKS hat eine bestehende AWS verwaltete Richtlinie aktualisiert.	13. Dezember 2021
Vorversion: Add-on-Support für Amazon-EBS-CSI-Treiber	Sie können jetzt eine Vorschau anzeigen AWS-Managementkonsole, indem Sie die AWS CLI und die API verwenden, um den Amazon EBS CSI-Treiber zu verwalten.	9. Dezember 2021
Support von Karpenter Autoscaler	Sie können jetzt das Karpenter-Open-Source-Projekt verwenden, um Ihre Knoten automatisch zu skalieren.	29. November 2021
Support für Fluent-Bit-Kubernetes-Filter bei der Fargate-Protokollierung	Sie können jetzt den Fluent-Bit-Kubernetes-Filter mit Fargate-Protokollierung verwenden.	10. November 2021
Windows-Support in der Steuerebene verfügbar	Windows-Support ist jetzt in Ihrer Steuerebene verfügbar. Sie müssen ihn nicht mehr in Ihrer Datenebene aktivieren.	9. November 2021

Bottlerocket wurde als AMI-Typ für verwaltete Knotengruppen hinzugefügt	Zuvor war Bottlerocket nur als Option für selbstverwaltete Knoten verfügbar. Jetzt kann es als verwaltete Knotengruppe konfiguriert werden, wodurch der Aufwand reduziert wird, der zur Erfüllung der Knoten-Compliance-Anforderungen erforderlich ist.	28. Oktober 2021
DL1 Treiber-Unterstützung	Benutzerdefiniertes Amazon Linux unterstützt AMIs jetzt Deep-Learning-Workloads für Amazon Linux 2. Diese Aktivierung ermöglicht eine generische On-Premises- oder Cloud-Baseline-Konfiguration.	25. Oktober 2021
VT1 Video-Unterstützung	Benutzerdefiniertes Amazon Linux unterstützt AMIs VT1 jetzt einige Distributionen. Diese Aktivierung bewirbt Xilinx U30-Geräte in Ihrem Amazon-EKS-Cluster.	13. September 2021
Amazon EKS Connector ist jetzt verfügbar	Sie können Amazon EKS Connector verwenden, um jeden konformen Kubernetes-Cluster zu registrieren und zu verbinden AWS und ihn in der Amazon EKS-Konsole zu visualisieren.	8. September 2021

Amazon EKS Anywhere ist jetzt verfügbar	Amazon EKS Anywhere ist eine neue Bereitstellungsoption für Amazon EKS, mit der Sie Kubernetes-Cluster einfach On-Premises erstellen und betreiben können.	8. September 2021
Amazon FSx für NetApp ONTAP CSI-Treiber	Es wurde ein Thema hinzugefügt, das den Amazon FSx for NetApp ONTAP CSI-Treiber zusammenfasst und Links zu anderen Referenzen enthält.	2. September 2021
Verwaltete Knotengruppen berechnen jetzt automatisch die von Amazon EKS empfohlenen maximalen Pods für Knoten	Verwaltete Knotengruppen berechnen jetzt automatisch die maximalen Amazon-EKS-Pods für Knoten, die Sie ohne eine Startvorlage bereitstellen oder mit einer Startvorlage, in der Sie keine AMI-ID angegeben haben.	30. August 2021
Amazon-EKS-Verwaltung der Add-on-Einstellungen entfernen, ohne die Amazon-EKS-Add-on-Software zu entfernen	Sie können jetzt ein Amazon-EKS-Add-on entfernen, ohne die Add-on-Software aus Ihrem Cluster zu entfernen.	20. August 2021
Erstellen mehrfach vernetzter Pods mit Multus	Sie können jetzt mit Multus mehrere Netzwerkschnittstellen zu einem Pod hinzufügen.	2. August 2021

Fügen Sie Ihren EC2 Linux-Amazon-Knoten weitere IP-Adressen hinzu	Sie können Ihren EC2 Linux-Amazon-Knoten jetzt deutlich mehr IP-Adressen hinzufüge n. Dies bedeutet, dass Sie auf jedem Knoten eine höhere Dichte an Pods ausführen können. Sie können Ihren EC2 Linux-Amazon-Knoten jetzt deutlich mehr IP-Adressen hinzufügen. Dies bedeutet, dass Sie auf jedem Knoten eine höhere Dichte an Pods ausführen können.	27. Juli 2021
`containerd` -Laufzeit-Bootstrap	Das für Amazon EKS optimiert e, beschleunigte Amazon Linux Amazon Machine Image (AMI) enthält jetzt ein Bootstrap-Flag, mit dem Sie die `containe rd` Laufzeit in Amazon EKS Optimized und AMIs Bottleroc ket aktivieren können. Dieses Flag ist in allen unterstützten Kubernetes-Versionen des AMI verfügbar.	19. Juli 2021
Kubernetes-Version `1.21`	Support für Kubernetes-Version `1.21` hinzugefügt.	19. Juli 2021
Thema zu verwalteten Richtlini en hinzugefügt	Eine Liste aller von Amazon EKS IAM verwalteten Richtlini en und Änderungen, die seit dem 17. Juni 2021 daran vorgenommen wurden.	17. Juni 2021

Verwendung von Sicherheitsgruppen für Pods mit Fargate	Sie können jetzt Sicherheitsgruppen für Pods mit Fargate verwenden, zusätzlich zu deren Verwendung mit EC2 Amazon-Knoten.	1. Juni 2021
CoreDNS- und `kube-proxy` -Amazon EKS-Add-ons hinzugefügt	Amazon EKS kann Ihnen jetzt bei der Verwaltung der CoreDNS- und `kube-proxy` -Amazon-EKS-Add-ons für Ihren Cluster helfen.	19. Mai 2021
Kubernetes-Version `1.20`	Support für Kubernetes-Version `1.20` für neue Cluster und Versions-Upgrades hinzugefügt.	18. Mai 2021
AWS Load Balancer Controller veröffentlicht `2.2.0`	Sie können jetzt den Load AWS Balancer Controller verwenden, um Elastic Load Balancer mithilfe von Instance- oder IP-Zielen zu erstellen.	14. Mai 2021
Knoten-Taints für verwaltete Knoten-Gruppen	Amazon EKS unterstützt jetzt das Hinzufügen von Notizmarkierungen zu verwalteten Knotengruppen.	11. Mai 2021
Geheimnisverschlüsselung für vorhandene Cluster	Amazon EKS unterstützt jetzt das Hinzufügen von Geheimnis-Verschlüsselung zu vorhandenen Clustern.	26. Februar 2021
Kubernetes-Version `1.19`	Support für Kubernetes-Version `1.19` für neue Cluster und Versions-Upgrades hinzugefügt.	16. Februar 2021

Amazon EKS unterstützt jetzt OpenID Connect (OIDC)-Identitätsanbieter als Methode zur Authentifizierung von Benutzern bei Clustern mit Version `1.16` oder höher.	OIDC-Identitätsanbieter können zusammen mit oder als Alternative zu AWS Identity and Access Management (IAM) verwendet werden.	12. Februar 2021
Sehen Sie sich Knoten- und Workload-Ressourcen an im AWS-Managementkonsole	Sie können jetzt Details zu Ihren verwalteten, selbstverwalteten und Fargate-Knoten und Ihren bereitgestellten Kubernetes-Workloads in der AWS-Managementkonsole.	1. Dezember 2020
Spot-Instance-Typen in einer verwalteten Knotengruppe bereitstellen	Sie können nun mehrere Spot- oder On-Demand-Instance -Typen für eine verwaltete Knotengruppe bereitstellen.	1. Dezember 2020
Amazon EKS kann jetzt spezifische Add-ons für Ihren Cluster verwalten	Sie können Add-Ons selbst verwalten oder Amazon EKS erlauben, den Start und die Version eines Add-Ons über die Amazon-EKS-API zu steuern.	1. Dezember 2020
Teilen Sie eine ALB über mehrere Ingresses	Sie können jetzt einen AWS Application Load Balancer (ALB) für mehrere Kubernetes-Ingresses gemeinsam nutzen. In der Vergangenheit mussten Sie für jeden Ingress eine separate ALB bereitstellen.	23. Oktober 2020

Unterstützung von NLB-IP-Zielen	Sie können jetzt einen Network Load Balancer (NLB) mit IP-Zielen bereitstellen. Das bedeutet, dass Sie einen NLB verwenden können, um den Netzwerkverkehr auf Fargate Pods und direkt auf Pods, die auf EC2 Amazon-Knoten laufen, zu verteilen.	23. Oktober 2020
Kubernetes-Version `1.18`	Support für Kubernetes-Version `1.18` für neue Cluster und Versions-Upgrades hinzugefügt.	13. Oktober 2020
Geben Sie einen benutzerdefinierten CIDR-Block für die Zuweisung der Kubernetes-Service-IP-Adresse an.	Sie können jetzt einen benutzerdefinierten CIDR-Block angeben, von dem Kubernetes-Service-IP-Adressen zuweist.	29. September 2020
Einzelnen Pods Sicherheitsgruppen zuweisen	Sie können jetzt einigen der einzelnen Pods, die auf vielen EC2 Amazon-Instance-Typen ausgeführt werden, verschiedene Sicherheitsgruppen zuordnen.	9. September 2020
Stellen Sie Bottlerocket auf Ihren Knoten bereit	Sie können jetzt Knoten bereitstellen, auf denen Bottlerocket ausgeführt wird.	31. August 2020
Die Möglichkeit, Arm-Knoten zu starten, ist allgemein verfügbar	Sie können jetzt Arm-Knoten in verwalteten und selbstverwalteten Knotengruppen starten.	17. August 2020

Startvorlagen für verwaltete Knotengruppen und benutzerdefiniertes AMI	Sie können jetzt eine verwaltete Knotengruppe bereitstellen, die eine EC2 Amazon-Startvorlage verwendet. Die Startvorlage kann bei Bedarf ein benutzerdefiniertes AMI angeben.	17. August 2020
EFS-Unterstützung für AWS Fargate	Sie können jetzt Amazon EFS mit AWS Fargate verwenden.	17. August 2020
Aktualisierung der Amazon-EKS-Plattformversion	Dies ist eine neue Plattformversion mit Sicherheitsfixes und -verbesserungen. Dies beinhaltet UDP-Support für Services vom Typ `LoadBalancer` bei Verwendung von Network Load Balancers mit Kubernetes-Version `1.15` oder höher. Weitere Informationen finden Sie unter dem Problem UDP für AWS Network Load Balancer zulassen unter GitHub.	12. August 2020
Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in den AWS -Regionen Afrika (Kapstadt) (`af-south-1`) und Europa (Mailand) (`eu-south-1`) verfügbar.	6. August 2020
Fargate-Nutzungskennzahlen	AWS Fargate stellt CloudWatch Nutzungsmetriken bereit, die Aufschluss darüber geben, wie Ihr Konto die Fargate-On-Demand-Ressourcen nutzt.	3. August 2020

Kubernetes-Version `1.17`	Support für Kubernetes-Version `1.17` für neue Cluster und Versions-Upgrades hinzugefügt.	10. Juli 2020
Erstellen und Verwalten von App-Mesh-Ressourcen aus Kubernetes mit dem App-Mesh-Controller für Kubernetes	Sie können App-Mesh-Ressourcen in Kubernetes erstellen und verwalten. Der Controller injiziert außerdem automatisch den Envoy-Proxy und die Init-Container in Pods, die Sie bereitstellen.	18. Juni 2020
Amazon EKS unterstützt jetzt Amazon EC2 Inf1-Knoten	Sie können Ihrem Cluster Amazon EC2 Inf1-Knoten hinzufügen.	4. Juni 2020
Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in den Regionen AWS GovCloud (US-Ost) (`us-gov-east-1`) und AWS GovCloud (US-West) (`us-gov-west-1`) AWS verfügbar.	13. Mai 2020
Kubernetes `1.12` wird in Amazon EKS nicht mehr unterstützt	Kubernetes-Version `1.12` wird in Amazon EKS nicht mehr unterstützt. Aktualisieren Sie alle `1.12`-Cluster auf Version `1.13` oder höher, um Service-Unterbrechungen zu vermeiden.	12. Mai 2020
Kubernetes-Version `1.16`	Support für Kubernetes-Version `1.16` für neue Cluster und Versions-Upgrades hinzugefügt.	30. April 2020

Die serviceverknüpfte AWSServiceRoleForAmazonEKS-Rolle wurde hinzugefügt	Die dienstgebundene AWSServiceRoleForAmazonEKS-Rolle wurde hinzugefügt.	16. April 2020
Kubernetes-Version `1.15`	Support für Kubernetes-Version `1.15` für neue Cluster und Versions-Upgrades hinzugefügt.	10. März 2020
Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in den AWS -Regionen Peking (`cn-north-1`) und Ningxia (`cn-northwest-1`) verfügbar.	26. Februar 2020
FSx für den Lustre CSI-Treiber	Es wurde ein Thema für die Installation des FSx for Lustre CSI-Treibers auf Kubernetes `1.14` Amazon EKS-Clustern hinzugefügt.	23. Dezember 2019
Beschränken des Netzwerkzugriffs auf den öffentlichen Zugriffsendpunkt eines Clusters	Mit diesem Update können Sie Amazon EKS verwenden, um die CIDR-Bereiche einzuschränken, die mit dem öffentlichen Zugriffsendpunkt des Kubernetes-API-Servers kommunizieren können.	20. Dezember 2019
Auflösen der privaten Zugriffsendpunkt-Adresse für einen Cluster von außerhalb einer VPC	Mit diesem Update können Sie Amazon EKS verwenden , um den privaten Zugriffs-Endpunkt des Kubernetes-API-Servers von außerhalb einer VPC aufzulösen.	13. Dezember 2019

(Beta) Amazon EC2 A1 EC2 Amazon-Instanzknoten	Starten Sie Amazon EC2 EC2 A1-Amazon-Instance-Knoten, die sich bei Ihrem Amazon EKS-Cluster registrieren.	4. Dezember 2019
Einen Cluster auf AWS Outposts erstellen	Amazon EKS unterstützt jetzt die Erstellung von Clustern auf AWS Outposts.	3. Dezember 2019
AWS Fargate auf Amazon EKS	Amazon-EKS-Kubernetes-Cluster unterstützen jetzt das Ausführen von Pods in Fargate.	3. Dezember 2019
Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in der AWS Region Kanada (Zentral) (`ca-central-1`) verfügbar.	21. November 2019
Verwaltete Knotengruppen	Von Amazon EKS verwaltete Knotengruppen automatisieren die Bereitstellung und das Lebenszyklusmanagement von Knoten (EC2 Amazon-Instances) für Amazon EKS Kubernetes-Cluster.	18. November 2019
Aktualisierung der Amazon-EKS-Plattformversion	Neue Plattformversionen zur Berücksichtigung von CVE-2019-11253.	6. November 2019
Kubernetes `1.11` wird in Amazon EKS nicht mehr unterstützt	Kubernetes-Version `1.11` wird in Amazon EKS nicht mehr unterstützt. Bitte aktualisieren Sie alle `1.11`-Cluster auf Version `1.12` oder höher, um Service-Unterbrechungen zu vermeiden.	4. November 2019

Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in der AWS Region Südamerika (São Paulo) (`sa-east-1`) verfügbar.	16. Oktober 2019
Windows-Unterstützung	Amazon-EKS-Cluster, auf denen Kubernetes Version `1.14` ausgeführt wird, unterstützen nun Windows-Workloads.	7. Oktober 2019
Auto Scaling	Es wurde ein Kapitel hinzugefügt, in dem einige der verschiedenen Arten von Kubernetes-Auto Scaling behandelt werden, die auf Amazon-EKS-Clustern unterstützt werden.	30. September 2019
Kubernetes-Dashboard-Update	Thema für die Installation des Kubernetes-Dashboards in Amazon-EKS-Clustern zur Verwendung der Beta `2.0`-Version aktualisiert.	28. September 2019
Amazon EFS-CSI-Treiber	Thema für die Installation des Amazon-EFS-CSI-Treibers in Kubernetes `1.14`-Amazon-EKS-Clustern hinzugefügt.	19. September 2019
Amazon EC2 Systems Manager Manager-Parameter für Amazon EKS-optimierte AMI-ID	Es wurde ein Thema zum Abrufen der für Amazon EKS optimierten AMI-ID mithilfe eines Amazon EC2 Systems Manager Manager-Parameters hinzugefügt. Mit diesem Parameter müssen Sie nicht mehr nach AMI suchen IDs.	18. September 2019

Amazon EKS-Ressourcenmarkierung	Sie können die Markierung Ihrer Amazon-EKS-Cluster verwalten.	16. September 2019
Amazon-EBS-CSI-Treiber	Thema für die Installation des Amazon-EBS-CSI-Treibers in Kubernetes `1.14`-Amazon-EKS-Clustern hinzugefügt.	9. September 2019
Neues Amazon-EKS-optimiertes AMIs für `CVE-2019-9512` und `CVE-2019-9514` gepatcht	Amazon EKS hat das für Amazon-EKS-optimierte AMI aktualisiert, um CVE-2019-9512 und CVE-2019-9514 zu adressieren.	6. September 2019
Ankündigung der Beendigung des Supports für Kubernetes `1.11` in Amazon EKS	Amazon EKS hat den Support für Kubernetes-Version `1.11` am 4. November 2019 eingestellt.	4. September 2019
Kubernetes-Version `1.14`	Support für Kubernetes-Version `1.14` für neue Cluster und Versions-Upgrades hinzugefügt.	03. September 2019
IAM-Rollen für Servicekonten	Mit IAM-Rollen für Servicekonten auf Amazon-EKS-Clustern können Sie eine IAM-Rolle mit einem Kubernetes-Servicekonto verknüpfen. Mit diesem Feature müssen Sie der Knoten-IAM-Rolle keine erweiterten Berechtigungen mehr bereitstellen. Auf diese Weise können Pods auf diesem Knoten aufrufen AWS APIs.	03. September 2019

Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in der AWS Region Naher Osten (Bahrain) (`me-south-1`) verfügbar.	29. August 2019
Aktualisierung der Amazon-EKS-Plattformversion	Neue Plattformversionen für Adresse CVE-2019-9512 und CVE-2019-9514.	28. August 2019
Aktualisierung der Amazon-EKS-Plattformversion	Neue Plattformversionen für Adresse CVE-2019-11247 und CVE-2019-11249.	5. August 2019
Erweiterung der Amazon-EKS-Region	Amazon EKS ist jetzt in der AWS Region Asien-Pazifik (Hongkong) (`ap-east-1`) verfügbar.	31. Juli 2019
Kubernetes `1.10` wird in Amazon EKS nicht mehr unterstützt	Kubernetes-Version `1.10` wird in Amazon EKS nicht mehr unterstützt. Aktualisieren Sie alle `1.10`-Cluster auf Version `1.11` oder höher, um Service-Unterbrechungen zu vermeiden.	30. Juli 2019
Thema über ALB Ingress Controller hinzugefügt	Der AWS ALB Ingress Controller für Kubernetes ist ein Controller, der bewirkt, dass bei der Erstellung von Ingress-Ressourcen ein ALB erstellt wird.	11. Juli 2019
Neues Amazon-EKS-optimiertes AMIs	`kubectl`Unnötige AMIs Binärdatei wird entfernt von.	3. Juli 2019

Kubernetes-Version `1.13`	Support für Kubernetes-Version `1.13` für neue Cluster und Versions-Upgrades hinzugefügt.	18. Juni 2019
Neues Amazon-EKS-optimiertes AMIs für `AWS-2019-005` gepatcht	Amazon EKS hat das für Amazon-EKS-optimierte AMI aktualisiert, um die unter link:security/security-bulletins/AWS-2019-005/[AWS-2019-005,type="marketing"] beschriebenen Schwachstellen zu beheben.	17. Juni 2019
Ankündigung der Einstellung des Supports für Kubernetes `1.10` in Amazon EKS	Amazon EKS hat am 22. Juli 2019 den Support von Kubernetes-Version `1.10` eingestellt.	21. Mai 2019
Aktualisierung der Amazon-EKS-Plattformversion	Neue Plattformversion für Kubernetes `1.11` und `1.10`-Cluster zur Unterstützung von benutzerdefinierten DNS-Namen im `kubelet`-Zertifikat und zur Verbesserung der `etcd`-Leistung.	21. Mai 2019

Erste Schritte mit `eksctl`	In diesem Leitfaden für die ersten Schritte wird beschrieben, wie Sie alle erforderlichen Ressourcen installieren können, um mit Amazon EKS mit `eksctl` zu beginnen. Dies ist ein ein einfaches Befehlszeilen-Dienstprogramm zum Erstellen und Verwalten von Kubernetes-Clustern auf Amazon EKS.	10. Mai 2019
AWS`get-token` CLI-Befehl	Der `aws eks get-token` Befehl wurde der AWS CLI hinzugefügt. Sie müssen den AWS IAM Authenticator für Kubernetes nicht mehr installieren, um Client-Sicherheitstoken für die Cluster-API-Serverkommunikation zu erstellen. Aktualisieren Sie Ihre AWS CLI-Installation auf die neueste Version, um diese neue Funktionalität nutzen zu können. Weitere Informationen finden Sie unter Installation der AWS Befehlszeilenschnittstelle im Benutzerhandbuch für die AWS Befehlszeilenschnittstelle.	10. Mai 2019

Aktualisierung der Amazon-EKS-Plattformversion	Neue Plattformversion für `1.12`-Kubernetes-Cluster zur Unterstützung von benutzerdefinierten DNS-Namen im `kubelet`-Zertifikat und zur Verbesserung der `etcd`-Leistung. Dadurch wird ein Fehler behoben, durch den bedingt `kubelet`-Daemons von Knoten alle paar Sekunden ein neues Zertifikat angefordert haben.	8. Mai 2019
Prometheus-Tutorial	Thema zur Bereitstellung von Prometheus auf Ihrem Amazon-EKS-Cluster hinzugefügt.	5. April 2019
Amazon-EKS-Steuerebenen-Protokollierung	Mit diesem Update können Sie Audit- und Diagnoseprotokolle direkt aus dem Amazon-EKS-Steuerungsbereich abrufen. Sie können diese CloudWatch Protokolle in Ihrem Konto als Referenz für die Sicherung und Ausführung von Clustern verwenden.	4. April 2019
Kubernetes-Version `1.12`	Support für Kubernetes-Version `1.12` für neue Cluster und Versions-Upgrades hinzugefügt.	28. März 2019
Einführungsleitfaden für App Mesh hinzugefügt	Dokumentation für den Einstieg mit App Mesh und Kubernetes hinzugefügt.	27. März 2019

Privater Amazon-EKS-API-Server-Endpunktzugriff	Dokumentation zur Deaktivierung des öffentlichen Zugriffs für den Kubernetes-API-Server-Endpunkt Ihres Amazon-EKS-Clusters hinzugefügt.	19. März 2019
Thema zur Installation von Kubernetes Metrics Server hinzugefügt	Der Kubernetes-Metrik-Server ist ein Aggregator für Ressourcenverbrauchsdaten in Ihrem Cluster.	18. März 2019
Liste der zugehörigen Open-Source-Projekte hinzugefügt.	Diese Open-Source-Projekte erweitern die Funktionalität von Kubernetes-Clustern AWS, auf denen sie ausgeführt werden, einschließlich Clustern, die von Amazon EKS verwaltet werden.	15. März 2019
Thema zur lokalen Installation von Helm hinzugefügt.	Der `helm`-Paketmanager für Kubernetes unterstützt Sie bei der Installation und Verwaltung von Anwendungen in Ihrem Kubernetes-Cluster. In diesem Thema wird gezeigt, wie die `helm`- und `tiller`-Binärdateien lokal installiert und ausgeführt werden. Auf diese Weise können Sie Diagramme mit der Helm-CLI auf Ihrem lokalen System installieren und verwalten.	11. März 2019

Aktualisierung der Amazon-EKS-Plattformversion	Neue Plattformversion, die `1.11`-Kubernetes-Cluster für Amazon EKS auf Patch-Ebene `1.11.8` aktualisiert, um CVE-2019-1002100 zu beheben.	8. März 2019
Erhöhtes Cluster-Limit	Amazon EKS hat die Anzahl der Cluster, die Sie in einer AWS Region erstellen können, von 3 auf 50 erhöht.	13. Februar 2019
Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in den AWS Regionen Europa (London) (`eu-west-2`), Europa (Paris) (`eu-west-3`) und Asien-Pazifik (Mumbai) (`ap-south-1`) verfügbar.	13. Februar 2019
Neues Amazon-EKS-optimiertes AMIs für `ALAS-2019-1156` gepatcht	Amazon EKS hat das für Amazon-EKS-optimierte AMI aktualisiert, um die unter ALAS-2019-1156 beschriebene Schwachstelle zu beheben.	11. Februar 2019
Neues Amazon-EKS-optimiertes AMIs für `ALAS2-2019-1141` gepatcht	Amazon EKS hat das für Amazon EKS optimierte AMI aktualisiert, um die Probleme zu beheben, auf CVEs die in ALAS2-2019-1141 verwiesen wird.	9. Januar 2019
Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in der AWS Region Asien-Pazifik (Seoul) (`ap-northeast-2`) verfügbar.	9. Januar 2019

Erweiterung der Amazon-EKS-Region	Amazon EKS ist jetzt in den folgenden zusätzlichen AWS Regionen verfügbar: Europa (Frankfurt) (`eu-central-1`), Asien-Pazifik (Tokio) (`ap-northeast-1`), Asien-Pazifik (Singapur) (`ap-southeast-1`) und Asien-Pazifik (Sydney) (`ap-southeast-2`).	19. Dezember 2018
Amazon EKS-Cluster-Aktualisierungen	Dokumentation für Amazon-EKS-Cluster Kubernetes-Versionsupdates und Knotenaustausch hinzugefügt.	12. Dezember 2018
Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in der AWS Region Europa (Stockholm) (`eu-north-1`) verfügbar.	11. Dezember 2018
Aktualisierung der Amazon-EKS-Plattformversion	Neue Plattformversion, die Kubernetes auf Patch-Ebene `1.10.11` aktualisiert, um das Problem link:security/security-bulletins/AWS-2018-020/ [CVE-2018-1002105,type="marketing"] zu beheben.	4. Dezember 2018
Support für Version`1.0.0` für den ALB-Controller für eingehenden Datenverkehr hinzugefügt	Der ALB Ingress Controller veröffentlicht eine Version `1.0.0` mit formaler Unterstützung von. AWS	20. November 2018

Unterstützung der CNI-Netzwerkkonfiguration hinzugefügt	Das Amazon-VPC-CNI-Plug-In für Kubernetes-Version `1.2.1` unterstützt jetzt eine benutzerdefinierte Netzwerkkonfiguration für Netzwerkschnittstellen des sekundären Pods.	16. Oktober 2018
Unterstützung für `MutatingAdmissionWebhook` und `ValidatingAdmissionWebhook` hinzugefügt	Die Amazon-EKS-Plattformversion `1.10-eks.2` unterstützt jetzt die Zulassungscontroller `MutatingAdmissionWebhook` und `ValidatingAdmissionWebhook` .	10. Oktober 2018
Partner-AMI-Informationen hinzugefügt.	Canonical hat sich mit Amazon EKS zusammengetan, um Knoten zu erstellen AMIs , die Sie in Ihren Clustern verwenden können.	3. Oktober 2018
Anweisungen für den AWS`update-kubeconfig` CLI-Befehl hinzugefügt	Amazon EKS hat das `update-kubeconfig` zur AWS CLI hinzugefügt, um das Erstellen einer `kubeconfig` Datei für den Zugriff auf Ihren Cluster zu vereinfachen.	21. September 2018
Neues Amazon EKS optimiert AMIs	Amazon EKS hat Amazon EKS optimiert AMIs (mit und ohne GPU-Unterstützung) aktualisiert, um verschiedene Sicherheitskorrekturen und AMI-Optimierungen bereitzustellen.	13. September 2018

Erweiterung der Amazon AWS EKS-Region	Amazon EKS ist jetzt in der Region Europa (Irland) (`eu-west-1`) verfügbar.	5. September 2018
Aktualisierung der Amazon-EKS-Plattformversion	Neue Plattformversion mit Unterstützung für die Kubernetes-Aggregationsebene und den Horizontal Pod Autoscaler (HPA).	31. August 2018
Neue Amazon EKS-optimierte Version AMIs und GPU-Unterstützung	Amazon EKS hat das für Amazon EKS optimierte AMI aktualisiert, sodass es eine neue AWS CloudFormation Knotenvorlage und ein neues Bootstrap-Skript verwendet . Außerdem ist ein neues Amazon-EKS-optimiertes AMI mit GPU-Unterstützung verfügbar.	22. August 2018
Neues Amazon-EKS-optimiertes AMIs für `ALAS2-2018-1058` gepatcht	Amazon EKS hat das für Amazon EKS optimierte AMI aktualisiert, um die Probleme zu beheben, auf CVEs die in ALAS2-2018-1058 verwiesen wird.	14. August 2018
Amazon-EKS-optimierte AMI-Build-Skripts	Amazon EKS hat die Build-Skripts, die zum Erstellen des Amazon-EKS-optimierten AMI verwendet werden, als Open-Source-Software zur Verfügung gestellt. Diese Build-Skripte sind jetzt auf verfügbar. GitHub	10. Juli 2018

Erstveröffentlichung von Amazon EKS	Erste Dokumentation für den Servicestart	5. Juni 2018

Beitrag zum EKS-Benutzerhandbuch

AWS hat eine verbesserte Beitragserfahrung für das EKS-Benutzerhandbuch eingeführt.

Sie können nun die Quelle des EKS-Benutzerhandbuchs direkt auf GitHub bearbeiten.

Die Dokumente verwenden jetzt AsciiDoc, eine leistungsstarke Authoring-Sprache ähnlich Markdown. AsciiDoc kombiniert einfache Syntax mit Features der Unternehmensdokumentation wie erweiterter Formatierung, Querverweisen und Sicherheitskontrollen.

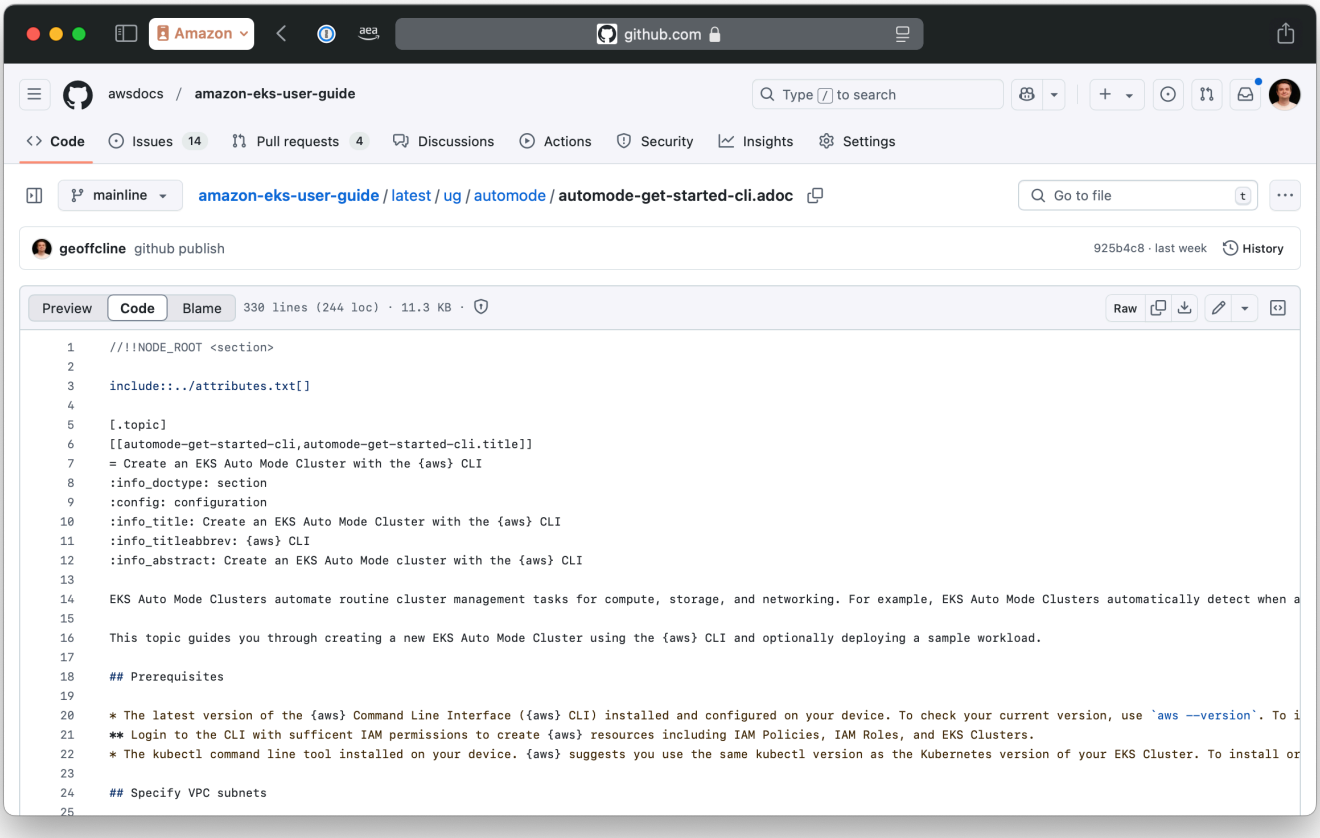
Sie können die EKS-Dokumente jetzt direkt auf GitHub bearbeiten. Unser optimierter Prozess umfasst:

- Schnellere Bearbeitung von Abrufanfragen
- Weniger manuelle Schritte
- Automatisierte Inhaltsqualitätsprüfungen

Wir freuen uns auf Ihre Beiträge.

Bearbeiten einer einzelnen Seite über einen Web-Browser

Sie können eine einzelne Seite im EKS-Benutzerhandbuch direkt über Ihren Web-Browser bearbeiten.



Informationen zum Bearbeiten mehrerer Seiten über Ihren Web-Browser finden Sie unter the section called “Bearbeiten Sie Dateien mit GitHub”.

Voraussetzungen

- Die Seite „Dokumente zum Ändern“ wurde im Web-Browser geöffnet.
- Bei GitHub angemeldet.

Verfahren

1. Navigieren Sie zu der Seite, die Sie im Amazon-EKS-Benutzerhandbuch bearbeiten möchten.
2. Wählen Sie im rechten Fensterbereich den Link Diese Seite in GitHub bearbeiten.
3. Öffnen Sie in GitHub den Editor, indem Sie entweder:
   - Die Taste e auf Ihrer Tastatur drücken.
   - Das Stiftsymbol klicken und im Dropdown-Menü die Option Vor Ort bearbeiten auswählen.

- Sie keine Bearbeitungsoption sehen, müssen Sie sich bei GitHub anmelden. Ihr GitHub-Konto benötigt keine besonderen Berechtigungen, um Änderungen vorzuschlagen. Interne Amazon-Mitwirkende sollten jedoch ihr GitHub-Profil verknüpfen.

4. Nehmen Sie die erforderlichen Änderungen am Inhalt im GitHub-Editor vor.
   - Der Editor bietet Syntaxhervorhebung und Vorschaufunktionen.
   - Sie können AsciiDoc-Markup verwenden, um Ihre Änderungen zu formatieren.
   - Mit `ctrl-f` können Sie eine Such-/Ersetzungsfunktion öffnen.
5. (Optional) Überprüfen Sie Ihre Änderungen in der Vorschau.
   - Verwenden Sie die Registerkarte `preview`, um eine Vorschau Ihrer Änderungen mit umfangreicher Formatierung anzuzeigen.
   - Nutzen Sie die Option `show diff`, um geänderte Abschnitte hervorzuheben. Entfernte Abschnitte sind am linken Rand mit einer roten Markierung gekennzeichnet. Neue Abschnitte sind am linken Rand mit einer grünen Markierung gekennzeichnet.
6. Wenn Sie mit der Bearbeitung fertig sind, klicken Sie oben im Editor auf die Schaltfläche Änderungen übernehmen …
7. Im Übernahme-Dialogfeld:
   - Überprüfen Sie, ob Ihre E-Mail-Adresse korrekt ist.
   - Fügen Sie eine kurze, aber aussagekräftige Übernahme-Nachricht hinzu, in der Sie Ihre Änderungen erläutern.
   - Bei Bedarf können Sie optional eine ausführlichere Beschreibung hinzufügen.
   - Wählen Sie diese Option, um eine neue Verzweigung und einen Abrufanfrage zu erstellen.

Sie haben eine Abrufanfrage mit den vorgeschlagenen Änderungen erstellt.

Übersicht über Abrufanfragen

Wenn Sie eine PR erstellen:

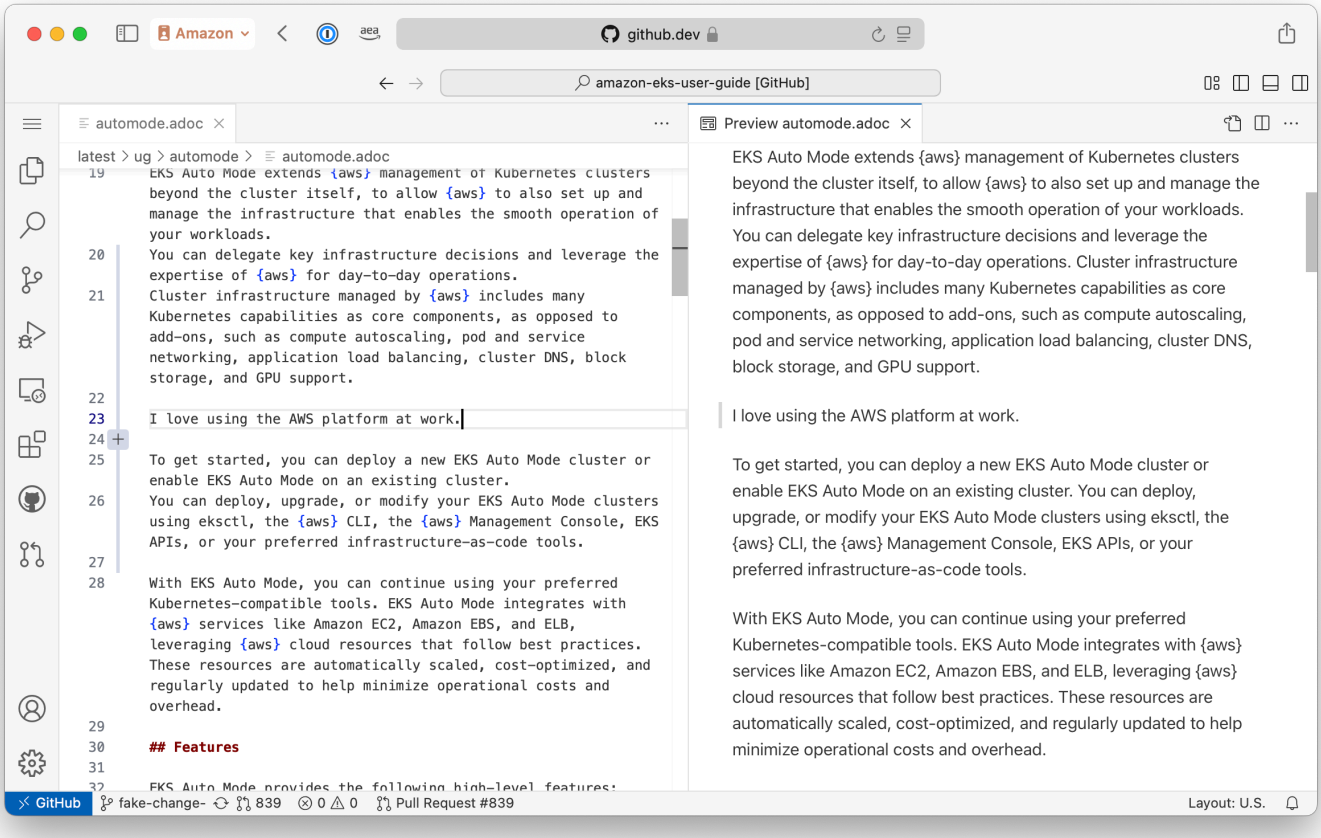
- Ihre Änderungen werden den Repository-Verwaltern zur Überprüfung vorgelegt.
- Die Prüfer können Ihre Änderungen kommentieren und Änderungen anfordern.
- Es können automatisierte Tests durchgeführt werden, um Ihre Änderungen zu validieren.
- Nach der Genehmigung können Ihre Änderungen in das Haupt-Repository übernommen werden.

Abrufanfragen tragen zur Qualitätssicherung bei und bieten die Möglichkeit, Änderungen vor ihrer Integration zu besprechen.

Erfahren Sie in der GitHub-Dokumentation, wie Abrufanfragen geprüft und genehmigt werden.

Bearbeiten Sie mehrere Dateien in einem Webbrowser mit dem GitHub Web Editor

Wenn Sie Änderungen an mehreren Seiten vorschlagen oder eine neue Dokumentseite erstellen möchten, verwenden Sie den GitHub .dev-Webeditor. Dieser Web-Editor basiert auf dem beliebten Texteditor Visual Studio Code.



Voraussetzungen

- Eingeloggt bei GitHub
- Vertrautheit mit Visual Studio Code Editor
- Vertrautheit mit Git

Verfahren

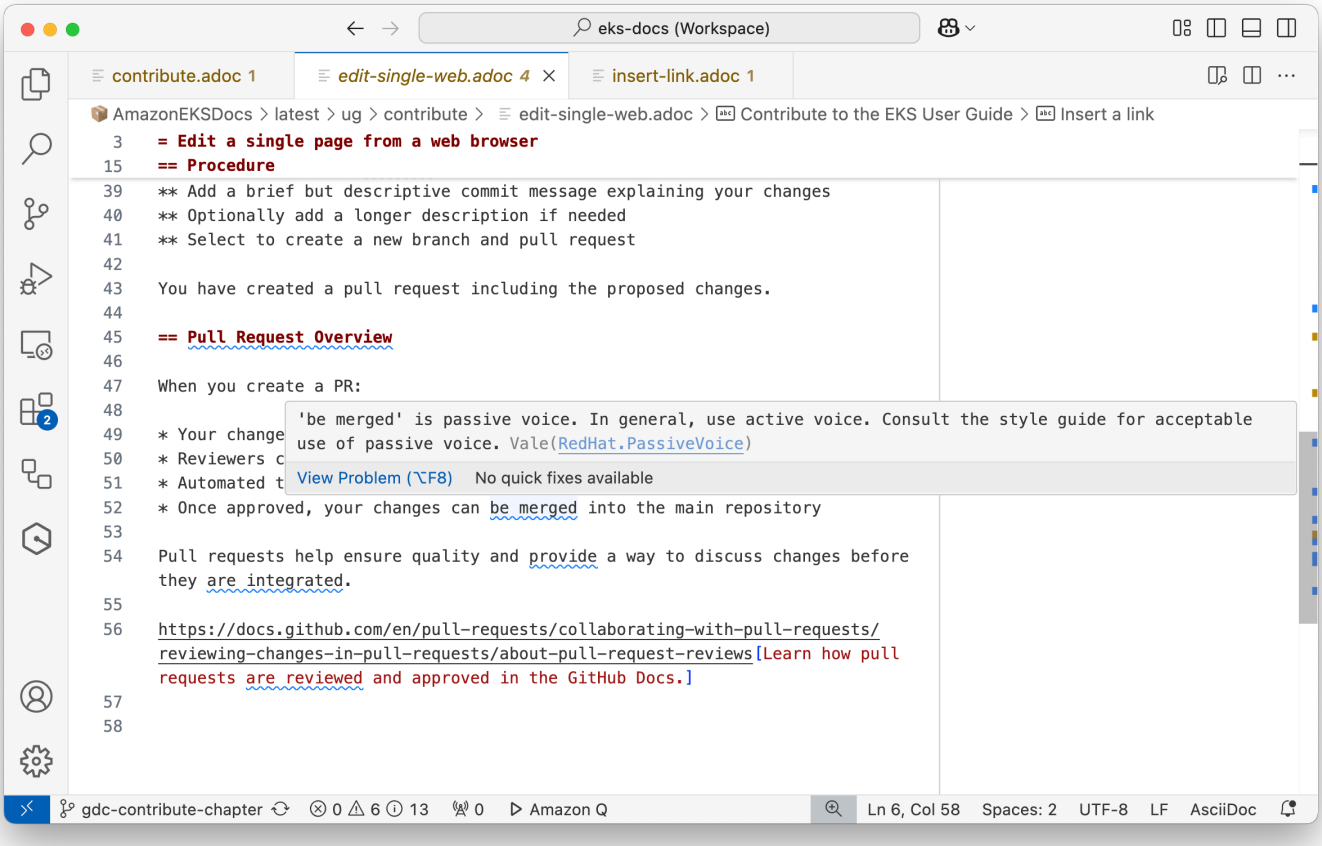
> **Note**
>
> Das Team von EKS Docs hat eine Workspace-Datei erstellt, die Konfigurationsvorschläge für den Editor enthält, z. B. Textumbruch und AsciiDoc Syntaxhervorhebung. Wir empfehlen Ihnen, diese Arbeitsbereichsdatei zu laden.

1. Öffnen Sie den Workspace auf GitHub .dev.
   - Sie können die URL `https://github.dev/awsdocs/amazon-eks-user-guide/blob/mainline/eks-docs.code-workspace?workspace=true` als Lesezeichen speichern
2. (Nur bei der Erstinstallation) Möglicherweise werden Sie aufgefordert, einen Fork des Repos in Ihrem eigenen GitHub Konto zu erstellen. Akzeptieren Sie diesen Prompt. Weitere Informationen finden Sie in den Dokumenten unter Über Forks. GitHub
3. (Nur bei der Erstinstallation) Akzeptieren Sie die Aufforderung unten rechts, um die AsciiDoc Erweiterung zu installieren.
4. Navigieren Sie zum Dokumentationsinhalt unter `latest/ug`.
   - Dokumentendateien werden nach ihrem obersten Abschnitt organisiert. Beispielsweise befinden sich die Quelldateien der Seiten im Kapitel „Sicherheit“ im Verzeichnis „security/“.
5. Um eine Vorschau einer Dokumentseite anzuzeigen, verwenden Sie die Schaltfläche Vorschau neben anzeigen oben rechts. Das Symbol enthält eine kleine Lupe.
6. Verwenden Sie die Registerkarte Quellcodeverwaltung auf der linken Seite, um Ihre Änderungen zu übernehmen. Weitere Informationen finden Sie in der Visual-Studio-Code-Dokumentation:
   - Änderungen übernehmen
   - Erstellen einer Pull-Anforderung

Nachdem Sie eine Abrufanfrage erstellt haben, wird diese vom Dokumentationsteam überprüft.

Vale lokal installieren, um während der Eingabe Stilempfehlungen anzuzeigen

Während der Eingabe werden Stilempfehlungen angezeigt. Dies unterstützt dabei, unpassende Formulierungen und Tippfehler zu erkennen.



Übersicht:

- Die Vale CLI lädt Stilrichtlinien und führt sie mit Quelldateien aus.
- Das EKS-Docs-Repository enthält eine Vale-Konfigurationsdatei, die Stilrichtlinien und lokale Regeln lädt.
- Die Vale-Erweiterung für Visual Studio (VS) Code zeigt Vale-Feedback innerhalb des Editors an.

Vale installieren

Befolgen Sie die Anweisungen in der Vale-CLI-Dokumentation, um Vale mit einem Paket-Manager zu installieren.

Erweiterung „VS Code Vale“ installieren

1. Öffnen Sie VS Code.

2. Klicken Sie in der Aktivitätsleiste auf das Symbol „Erweiterungen“ (oder drücken Sie Strg +Umschalt+X).
3. Suchen Sie nach „Vale“.
4. Klicken Sie bei der Erweiterung „Vale VSCode“ von Chris Chinchilla auf „Installieren“.
5. Laden Sie VS Code neu, wenn Sie dazu aufgefordert werden.

Vale synchronisieren

Vale verwendet die `.vale.ini`-Konfigurationsdatei in Ihrem Projektstammverzeichnis, um zu bestimmen, welche Stilregeln angewendet werden sollen.

1. Öffnen Sie VS Code.
2. Klicken Sie auf Ansicht > Terminal (oder drücken Sie Strg+’).
3. Navigieren Sie bei Bedarf zum Stammverzeichnis Ihres Projekts.
4. Führen Sie den Befehl aus:

```
vale sync
```

5. Warten Sie, bis Vale den Download und die Synchronisierung der Stilregeln abgeschlossen hat.

Stilempfehlungen in VS Code anzeigen

1. Öffnen Sie eine Markdown- oder AsciiDoc-Datei in VS Code.
2. Die Vale-Erweiterung prüft Ihren Text automatisch anhand der Stilregeln.
3. Stilistische Probleme werden im Editor unterstrichen.
4. Bewegen Sie den Mauszeiger über den unterstrichenen Text, um den entsprechenden Stilvorschlag anzuzeigen.
5. Beheben Sie Probleme, indem Sie den Vorschlägen folgen oder die Stilregeln konsultieren.

Erstellen einer neuen Seite

Erfahren Sie, wie Sie eine neue Dokumentationsseite erstellen. Dieses Thema enthält Anweisungen zum Erstellen der Metadaten für die Startseite und zum Hinzufügen der Seite zum Inhaltsverzeichnis des Leitfadens.

Seite erstellen

1. Navigieren Sie zum Kapitelverzeichnis. Wenn Sie beispielsweise im Bereich „Sicherheit“ eine neue Seite erstellen möchten, navigieren Sie zum `latest/ug/security`-Verzeichnis.
2. Ermitteln Sie die Seiten-ID. Die Seiten-ID besteht üblicherweise aus Kleinbuchstaben und ist mit `-` segmentiert. Die ID dieser Seite lautet `create-page`.
3. Erstellen Sie eine neue Datei mit der Seiten-ID und der `adoc`-Erweiterung. Beispiel, `create-page.adoc`.
4. Fügen Sie die Seitenmetadaten mithilfe dieser Vorlage ein:

```
include::../attributes.txt[]

[.topic]
[#unique-page-id]
= Page title
:info_titleabbrev: Short title

[abstract]
--
Brief summary of the page's content.
--

Introduction paragraph goes here.
```

Seite zur Navigation hinzufügen

1. Navigieren Sie zur übergeordneten Seite. Die übergeordnete Seite der Abschnitte der obersten Ebene lautet `book.adoc`.
2. Fügen Sie unten auf der übergeordneten Seite die untergeordnete Seite ein.

   `include::${filename}[leveloffset=+1]`

   Beispiel:

   `include::create-page.adoc[leveloffset=+1]`

Einfügen eines Links

AsciiDoc unterstützt mehrere Arten von Links. Es ist wichtig, die richtige Art von Link zu verwenden, damit der Link in verschiedenen Umgebungen ordnungsgemäß funktioniert.

Link zu einer Seite oder einem Abschnitt im EKS-Benutzerhandbuch

Verwenden Sie Querverweise (XRef), um zwischen Seiten oder Abschnitten innerhalb derselben Dokumentationssite zu verlinken, beispielsweise dem EKS-Benutzerhandbuch. Sie werden automatisch aktualisiert, wenn der Zielabschnitt verschoben oder umbenannt wird.

Seitentitel als Linktext verwenden

Wenn Sie in diesem Benutzerhandbuch auf eine andere ID verweisen, verwenden Sie in den meisten Fällen die folgende Vorgehensweise, damit der Linktext bei Bedarf automatisch auf den neuesten Titel aktualisiert wird.

```
For more information, see <<page-or-section-id>>.
```

Benutzerdefinierten Linktext festlegen

Für Fälle, in denen Sie einen benutzerdefinierten Linktext verwenden müssen, verwenden Sie das folgende Format.

```
Here's an example of a <<page-or-section-id,link with custom text>>.
```

Link zu einem anderen Leitfaden in den AWS Dokumenten

1. Finden Sie den Link zur AWS Dokumentationsseite.
2. Entfernen Sie das `https://docs.aws.amazon.com/`-Präfix und behalten Sie nur den Pfad bei. Der Pfad sollte mit einem Buchstaben beginnen.
3. Erstellen Sie einen Link wie unten gezeigt:

```
link:AmazonS3/latest/userguide/create-bucket-overview.html[Create a bucket,
 type="documentation"]
```

Link zu einer externen Webseite

Dieses Format erstellt einen Standard-Link zu einer Seite, die nicht von Amazon gehostet wird. Verwenden Sie dies beispielsweise für GitHub Links.

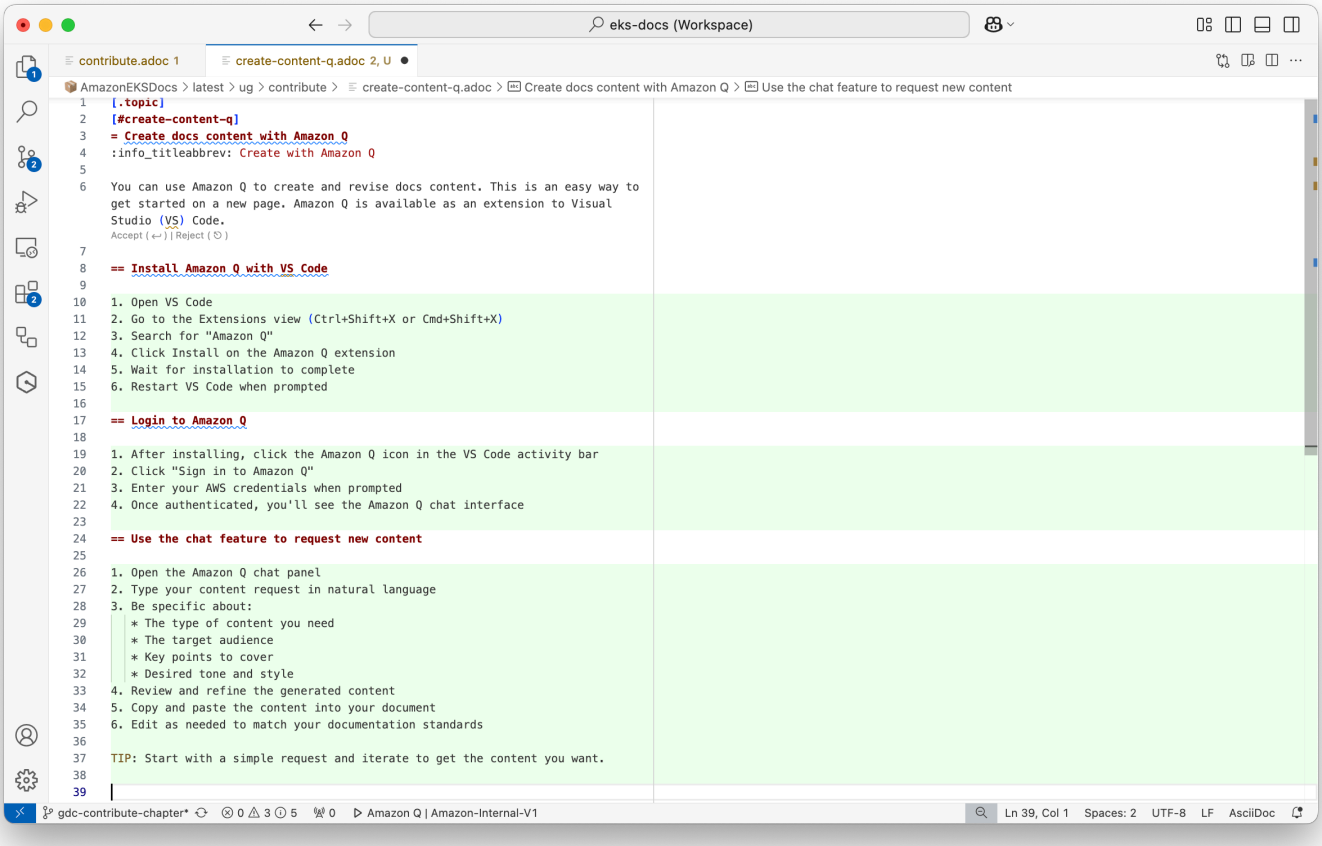
```
https://example.com[Link text]
```

> **Note**
>
> Wir verfügen über eine Zulassungsliste für externe Domains. Die Zulassungsliste befindet sich unter `vale/styles/EksDocs/ExternalDomains.yml`

Erstellung von Dokumenteninhalten mit Amazon Q

Sie können Amazon Q zum Erstellen und Überarbeiten von Dokumentinhalten verwenden. Dies ist eine einfache Möglichkeit, mit einer neuen Seite zu beginnen. Amazon Q ist als Erweiterung für Visual Studio (VS) Code verfügbar.

Im folgenden Image hat Amazon Q die grün markierten Zeilen generiert.



Amazon Q mit VS Code installieren

1. Öffnen Sie VS Code
2. Wechseln Sie zur Ansicht „Erweiterungen“ (Strg+Umschalt+X oder Cmd+Umschalt+X)
3. Suchen Sie nach „Amazon Q“
4. Klicken Sie auf „Installieren“ in der Amazon-Q-Erweiterung
5. Warten Sie, bis die Installation abgeschlossen ist
6. Starten Sie VS Code neu, wenn Sie dazu aufgefordert werden

Bei Amazon Q anmelden

1. Wählen Sie nach der Installation das Amazon-Q-Symbol in der VS-Code-Aktivitätsleiste.
2. Wählen Sie Bei Amazon Q anmelden aus.
3. Geben Sie Ihre AWS-Anmeldeinformationen ein, wenn Sie dazu aufgefordert werden.

4. Nach der Authentifizierung wird Ihnen die Amazon Q-Chat-Oberfläche angezeigt.

Verwendung von Amazon Q zur Erstellung von Inhalten

1. Öffnen Sie die Datei, die Sie bearbeiten möchten, in VS Code.
2. Wählen Sie den Text aus, den Sie überarbeiten möchten, oder den Speicherort für neue Inhalte.
3. Drücken Sie Strg+I oder Cmd+I.
4. Geben Sie im Prompt Folgendes an:
   - Die Art von Inhalten, die Sie benötigen.
   - Die Zielgruppe.
   - Wichtige Punkte, die Sie abdecken möchten.
   - Gewünschter Ton und Stil.
5. Überprüfen Sie den generierten Inhalt in der Inline-Vorschau.
6. Drücken Sie die Eingabetaste, um die Änderungen zu akzeptieren, oder die Esc-Taste, um sie abzulehnen.
7. Nehmen Sie bei Bedarf weitere Änderungen vor.

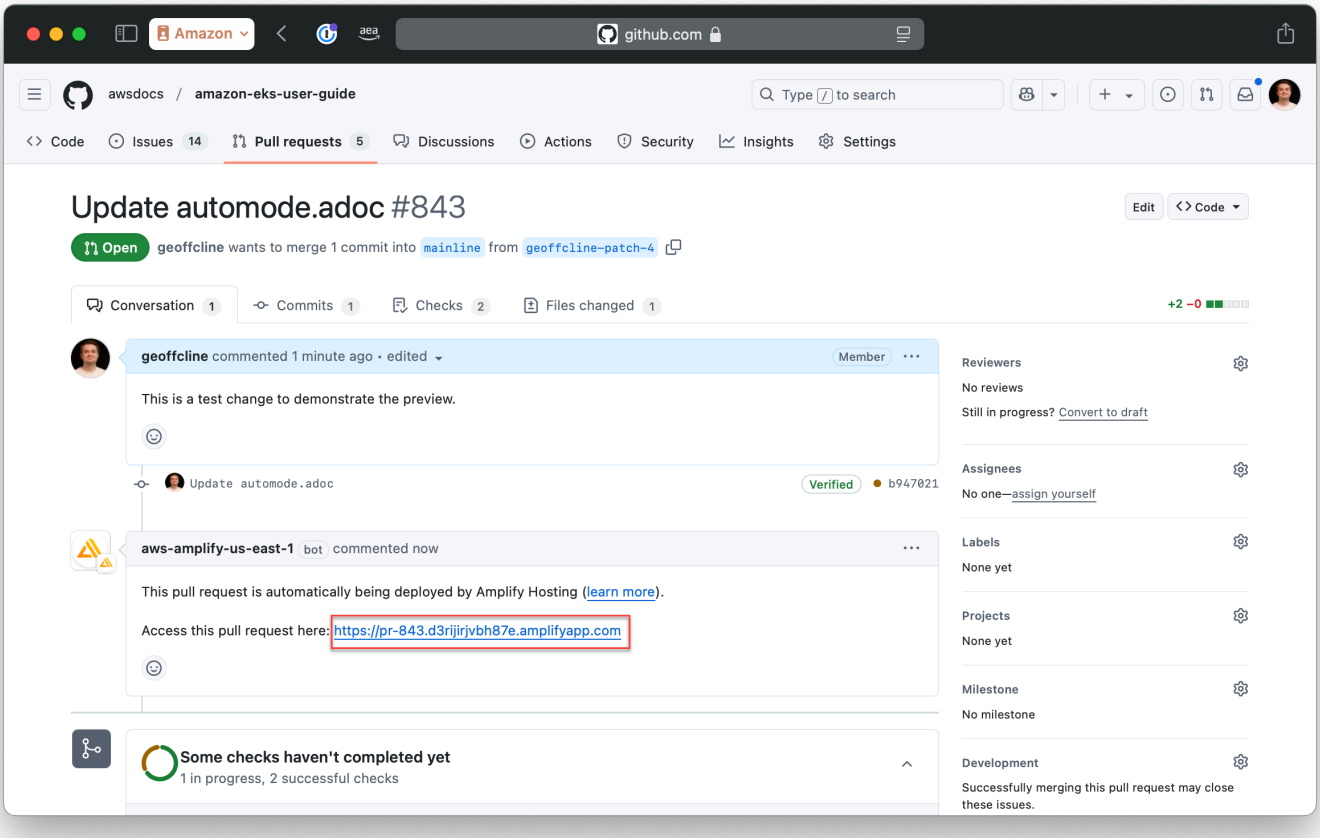
Tipps

- Beginnen Sie mit einer einfachen Anforderung und wiederholen Sie diese, bis Sie den gewünschten Inhalt erhalten.
- Erstellen Sie einen ersten Entwurf der Seitenüberschriften und bitten Sie Q, diese auszufüllen.
- Amazon Q gibt möglicherweise Markdown aus. Das ist in Ordnung. Die AsciiDoc-Tools verstehen die meisten Markdown-Syntaxen

Weitere Informationen zu Amazon Q Developer finden Sie unter Verwendung von Amazon Q Developer in der IDE.

Vorschau des Inhalts von Abrufanfragen

Das Amazon-EKS-Benutzerhandbuch GitHub ist so konfiguriert, dass eine Vorschau der Dokumentensite erstellt und generiert wird. Diese Vorschau enthält nicht das vollständige AWS-Design, überprüft jedoch, ob der Inhalt korrekt aufgebaut ist und die Links funktionieren.



Diese Vorschau wird von AWS Amplify unter einer temporären URL bereitgestellt.

Vorschau anzeigen

Wenn Sie eine Abrufanfrage einreichen, versucht AWS Amplify, eine Vorschau des Inhalts zu erstellen und bereitzustellen.

Wenn die Entwicklung erfolgreich ist, fügt aws-amplify-us-east-1 einen Kommentar zur Abrufanfrage hinzu, der einen Link zur Vorschau enthält. Wählen Sie den Link rechts neben „Auf diese Abrufabfrage zugreifen“ (im Screenshot rot umrandet).

Wenn die Entwicklung fehlschlägt, können die Repo-Administratoren die Protokolle anzeigen und Feedback geben.

> **Note**
>
> Wenn Sie bisher noch nicht beigetragen haben, muss ein Projektbetreuer möglicherweise die Ausführung der Entwicklung genehmigen.

Einschränkungen in der Vorschau

Die Vorschau wird als einzelne große HTML-Datei erstellt. Nach der Veröffentlichung wird sie auf mehreren Seiten angezeigt.

Was funktioniert:

- Querverweise (`xref`)
- Verbindungen zum Internet
- Bilder
- Inhalt gehostet von `samples/`

Was nicht funktioniert:

- Links zu anderen AWS-Inhalten, unter Verwendung von `type="documentation"`. Dies liegt daran, dass dieser Inhalt in der Vorschau-Umgebung nicht vorhanden ist.
- Das Attribut `{aws}` wird nicht korrekt angezeigt. Der Wert hierfür variiert je nach Umgebung.

AsciiDoc-Syntax-Referenz

AsciiDoc ist die bevorzugte Auszeichnungssprache für die AWS-Dokumentation. Obwohl die Tools die Markdown-Syntax (wie Überschriften und Listen) teilweise unterstützen, empfehlen wir, für alle Inhalte die AsciiDoc-Syntax zu verwenden, um Konsistenz und korrekte Darstellung zu gewährleisten.

Dieser Leitfaden behandelt gängige Syntaxelemente, die Sie für Beiträge zur Amazon-EKS-Dokumentation benötigen. Eine erweiterte Syntax und detaillierte Informationen finden Sie in der AsciiDoc-Dokumentation.

Neue Seite

Jedes neue AsciiDoc-Dokument sollte mit der in the section called “Seite erstellen” definierten Struktur beginnen.

Überschriften

Verwenden Sie Überschriften, um Ihre Inhalte hierarchisch zu organisieren:

```
= Page/topic title (level 1)
== Section title (level 2)
=== Level 3 heading
==== Level 4 heading
===== Level 5 heading
====== Level 6 heading
```

Hinweis: Verwenden Sie für Überschriften in der AWS-Dokumentation immer die Groß-/Kleinschreibung.

Text-Formatierung

Formatieren Sie Text, um wichtige Informationen hervorzuheben:

```
Use *bold text* for UI labels.
Use _italic text_ for introducing terms or light emphasis.
Use `monospace text` for code, file names, and commands.
```

Listen

Nicht nummerierte Listen

Erstellen Sie Aufzählungslisten für Elemente ohne bestimmte Reihenfolge:

```
* First item
* Second item
** Nested item
** Another nested item
* Third item
```

Nummerierte Listen

Erstellen Sie nummerierte Listen für aufeinanderfolgende Schritte oder priorisierte Elemente:

```
. First step
. Second step
.. Substep 1
.. Substep 2
. Third step
```

Links

Weitere Informationen zum richtigen Formatieren von Links finden Sie unter the section called “Link einfügen”. Links im Markdown-Stil werden nicht unterstützt.

Codebeispiele

Inline-Code

Verwenden Sie inverse Anführungszeichen für Inline-Code:

```
Use the `kubectl get pods` command to list all pods.
```

Code-Blöcke

Erstellen Sie Code-Blöcke mit Syntaxhervorhebung und Support für Attribute (ähnlich wie bei Entitäten):

```
[source,python,subs="verbatim,attributes"]
----
def hello_world():
    print("Hello, World!")
----
```

Bilder

Fügen Sie Images mit Alt-Text für Barrierefreiheit ein:

```
image::images/image-file.png[Alt text description]
```

Tabellen

Erstellen Sie Tabellen, um Informationen zu organisieren:

```
[%header,cols="2"]
|===
|Header 1
|Header 2

|Cell 1,1
|Cell 1,2

|Cell 2,1
|Cell 2,2
|===
```

Informationen zu komplexeren Tabellen finden Sie in der AsciiDoc-Tabellendokumentation.

Callouts

Verwenden Sie Hervorhebungen, um wichtige Informationen und Hinweise hervorzuheben:

```
NOTE: This is a note callout for general information.

TIP: This is a tip callout for helpful advice.

IMPORTANT: This is an important callout for critical information.
```

Vorversion:

> **Note**
>
> Dies ist eine Hinweis-Hervorhebung.

Andere Dateien einschließen

Inhalte aus anderen Dateien einschließen:

```
include::filename.adoc[]
```

Attribute (ähnlich wie Entitäten)

Verwenden Sie vordefinierte Attribute, um Konsistenz zu gewährleisten. Insbesondere MÜSSEN Sie Attribute für AWS und `arn:aws:` verwenden.

```
{aws} provides Amazon EKS as a managed Kubernetes service.
```

```
[source,bash,subs="verbatim,attributes"]
----
aws iam attach-role-policy \
    --role-name AmazonEKSAutoClusterRole \
    --policy-arn {arn-aws}iam::aws:policy/AmazonEKSClusterPolicy
----
```

Eine Liste der Attribute finden Sie in der `../attributes.txt`-Datei.

Verfahren

Schrittweise Anleitungen formatieren:

```
To create an Amaozon EKS cluster. do the following steps.

. Sign in to the {aws} Management Console.
. Open the Amazon EKS console.
. Choose *Create cluster*.

...
```