Guía del usuario

Amazon EKS

Amazon EKS: Guía del usuario

Table of Contents

..... xxvi
..... xxvii
¿Qué es Amazon EKS? 1
Amazon EKS: administración simplificada de Kubernetes 1
Creación y escalado con Kubernetes: capacidades de Amazon EKS 2
Características de Amazon EKS 3
Servicios relacionados 4
Precios de Amazon EKS 5
Casos de uso comunes 6
Arquitectura 8
Plano de control 8
Computación 8
Capacidades de EKS 10
Conceptos de Kubernetes 11
¿Por qué debería elegir Kubernetes? 12
Clústeres 17
Cargas de trabajo 22
Pasos a seguir a continuación 28
Opciones de implementación 29
Comprensión de las opciones de implementación de Amazon EKS 29
Amazon EKS en la nube 29
Amazon EKS en el centro de datos o en entornos periféricos 30
Amazon EKS Anywhere para entornos aislados 31
Herramientas de Amazon EKS 32
Configuración 33
Pasos a seguir a continuación 33
Configurar AWS CLI 33
Para crear una clave de acceso 34
Configuración de la CLI de AWS 34
Cómo obtener un token de seguridad 35
Cómo verificar la identidad del usuario 35
Pasos a seguir a continuación 36
Configure kubectl y eksctl 36
Instalación o actualización de `kubectl` 36

Paso 1: compruebe si kubectl está instalado ... 37
Paso 2: instalación o actualización de kubectl ... 37
Instalar eksctl ... 52
Siguientes pasos ... 52
Inicio rápido ... 53
En este tutorial: ... 53
Requisitos previos ... 53
Configuración del clúster ... 54
Cree la IngressClass ... 54
Implemente la aplicación de muestra del videojuego 2048 ... 55
Conservación de los datos con el modo automático de Amazon EKS ... 58
Limpieza ... 60
Aprendizaje de Amazon EKS ... 61
Descripción general ... 61
Taller de Amazon EKS ... 62
Tutoriales prácticos de configuración de clústeres de Amazon EKS ... 63
Amazon EKS Samples ... 64
Tutoriales de AWS ... 65
Taller de desarrolladores ... 65
Taller de Terraform ... 65
Formación en AWS Amazon EKS ... 66
Introducción ... 67
Creación de un clúster (modo automático de EKS) ... 67
Creación de un clúster (eksctl) ... 68
Requisitos previos ... 69
Paso 1: creación del clúster y de los nodos de Amazon EKS ... 69
Paso 2: ver los recursos de Kubernetes ... 70
Paso 3: eliminación de sus clústeres y nodos ... 72
Pasos a seguir a continuación ... 73
Creación de un clúster (consola y CLI) ... 73
Requisitos previos ... 74
Paso 1: creación del clúster de Amazon EKS ... 75
Paso 2: configuración del equipo para comunicarse con el clúster ... 77
Paso 3: creación de nodos ... 78
Paso 4: visualización de recursos ... 81
Paso 5: eliminación de recursos ... 81

Siguientes pasos 82
Modo automático de EKS 83
Características 83
Componentes automatizados 84
Configuración 86
Modelo de responsabilidad compartida 87
Crear un clúster 88
CLI de eksctl 89
AWS CLI 91
Consola de administración 99
Cómo habilitar clústeres existentes 101
Referencia para las migraciones 102
Migración de volúmenes de EBS 103
Migración de los equilibradores de carga 104
Habilitación en el clúster 105
Migre desde Karpenter 109
Migración desde MNG 112
Migración de Fargate 113
Ejecución de cargas de trabajo 119
Implemente una carga de trabajo de expansión 120
Implementación del equilibrador de carga 123
Implementación de una carga de trabajo con estado 128
Implementación de carga de trabajo acelerada 133
Configuración 140
Cómo crear una clase de nodos 145
Creación de un grupo de nodos 153
Grupos de nodos con capacidad estática 160
Creación de una clase de ingreso 166
Crear un servicio 175
Creación de una StorageClass 181
Cómo desactivar el modo automático de EKS 192
Actualización de una versión de Kubernetes 193
Revisión de los grupos de nodos integrados 195
Control de la implementación 197
Ejecución de complementos esenciales 198
Uso de políticas de red 200

Etiquetado de subredes 212
Generación del informe del CIS 214
Cifrado de volúmenes raíz 217
Actualización de los controles de AMI 221
Control de las reservas de capacidad 225
Uso de Zonas locales 230
Funcionamiento 234
Instancias administradas 235
Identidad y acceso 240
Red 246
Solución de problemas 249
Agente de supervisión de nodos 251
Obtenga la salida de la consola de una instancia administrada por EC2 mediante la CLI de AWS EC2 251
Obtenga los registros de los nodos mediante contenedores de depuración y la `kubectl` CLI 252
Consulte los recursos asociados al modo automático de EKS en la consola de AWS 253
Consulte los errores de IAM en la cuenta de AWS 254
Solución de problemas de pods que no se pueden programar en el nodo del modo automático 254
Solución de problemas de un nodo que no se une al clúster 254
Cómo compartir volúmenes entre pods 257
Visualización de los eventos de Karpenter en los registros del plano de control 258
Solución de problemas de los controladores incluidos en el modo automático 259
Recursos relacionados 217
Notas de la versión 260
19 de noviembre de 2025 260
19 de noviembre de 2025 260
23 de octubre de 2025 261
1 de octubre de 2025 261
30 de septiembre de 2025 261
29 de septiembre de 2025 261
10 de septiembre de 2025 261
24 de agosto de 2025 261
15 de agosto de 2025 261
6 de agosto de 2025 262

30 de junio de 2025 262
20 de junio de 2025 262
13 de junio de 2025 262
30 de abril de 2025 262
18 de abril de 2025 262
11 de abril de 2025 263
4 de abril de 2025 263
31 de marzo de 2025 263
21 de marzo de 2025 263
14 de marzo de 2025 263
Configuración de los clústeres 264
Cómo crear un clúster automático 264
Requisitos previos 53
Crear un clúster: consola de AWS 266
Crear clúster: AWS CLI 271
Siguientes pasos 99
Creación de un clúster 277
Requisitos previos 53
Paso 1: creación de un rol de IAM del clúster 278
Paso 2: creación de un clúster 279
Paso 3: actualización de kubeconfig 289
Paso 4: configuración del clúster 289
Siguientes pasos 99
Creación de un plano de control aprovisionado 291
Descripción general 97
Casos de uso 152
Niveles de escalado del plano de control 294
Consideraciones 104
Introducción al plano de control aprovisionado 298
Información sobre clústeres 306
Tipos de información del clúster 306
Consideraciones 306
Casos de uso 306
Introducción 308
Consulta de la información del clúster 308
Actualización de una versión de Kubernetes 314

Consideraciones para el modo automático de Amazon EKS 316
Resumen 316
Paso 1: preparación para la actualización 317
Paso 2: revisión de las consideraciones de actualización 318
Paso 3: actualización del plano de control del clúster 319
Paso 4: actualización de los componentes del clúster 321
Actualización a una versión anterior de Kubernetes en un clúster de Amazon EKS 323
Eliminar un clúster 323
Consideraciones 104
Eliminación del clúster (eksctl) 324
Eliminar un clúster (Consola de AWS) 325
Eliminación de un clúster (AWS CLI) 326
Protección de los clústeres de EKS contra la eliminación accidental 327
Acceso al punto de conexión del clúster 329
formato de punto de conexión del clúster IPv6 329
Formato de punto de conexión del clúster IPv4 330
Punto de conexión privado del clúster 330
Modificar el acceso al punto de conexión del clúster 331
Acceso a un servidor de API solo privado 334
Configuración del acceso al punto de conexión 336
Habilitación de la compatibilidad con Windows 339
Consideraciones 104
Requisitos previos 53
Habilitación de la compatibilidad con Windows 341
Implementación de pods de Windows 343
Soporte para una mayor densidad de pods en los nodos de Windows 344
Deshabilitar la compatibilidad con Windows 344
Clústeres privados 345
Requisitos de la arquitectura del clúster 345
Requisitos del nodo 346
Requisitos del pod 347
Escalado automático 350
Modo automático de EKS 350
Soluciones adicionales 350
Información sobre el cambio de zona 351
Descripción del flujo de tráfico de red de este a oeste entre pods 352

Requisitos del cambio de zona de EKS ... 357
Preguntas frecuentes ... 366
Recursos adicionales ... 369
Habilitación del cambio de zona ... 369
Consideraciones ... 370
¿Qué es el controlador de recuperación de aplicaciones de Amazon? ... 370
¿Qué es el cambio de zona? ... 370
¿Qué es el cambio de zona automático? ... 371
¿Qué hace el EKS durante un cambio automático? ... 371
Registre un clúster de EKS con el Controlador de recuperación de aplicaciones de Amazon (ARC) (consola de AWS) ... 371
Siguientes pasos ... 99
Administración de acceso ... 373
Tareas comunes ... 373
Introducción ... 212
Consideraciones para el modo automático de EKS ... 323
Acceso a la API de Kubernetes ... 375
Asociación de las identidades de IAM con los permisos de Kubernetes ... 376
Establecimiento de nodos de autenticación del clúster ... 377
Entradas de los registros ... 378
ConfigMap de aws-auth ... 438
Vinculación del proveedor de OIDC ... 448
Desvinculación del proveedor de OIDC ... 454
Acceso a recursos del clúster ... 454
Permisos necesarios ... 455
Acceso al clúster con kubectl ... 461
Crear el archivo `kubeconfig` de forma automática ... 462
Acceso de la carga de trabajo a AWS ... 463
Tokens de cuenta de servicio ... 464
Complementos de clúster ... 465
Concesión de permisos a AWS Identity and Access Management a cargas de trabajo en clústeres de Amazon Elastic Kubernetes Service ... 466
Credenciales con IRSA ... 471
Pod Identity ... 495
Administración de la computación ... 531
Comparación de opciones de computación ... 531

Grupos de nodos administrados 539
Conceptos de grupos de nodos administrados 540
Tipos de capacidad de grupo de nodos administrado 543
Creación 547
Actualización 559
Actualizar los detalles del comportamiento 564
Plantillas de inicialización 568
Eliminar 586
Nodos autoadministrados 588
Amazon Linux 590
Bottlerocket 599
Windows 603
Ubuntu Linux 612
Métodos de actualización 615
AWS Fargate 628
Consideraciones sobre Fargate de AWS 629
Tabla de comparación de Fargate 632
Introducción 634
Definición de perfiles 640
Eliminar perfiles 645
Detalles de configuración de pod 647
Eventos de aplicación de revisiones del sistema operativo 650
Recopilación de métricas 653
Registro 655
Tipos de instancias de Amazon EC2 667
Número máximo de pods recomendado por Amazon EKS para cada tipo de instancia de Amazon EC2 669
Consideraciones para el modo automático de EKS 323
AMI optimizadas precompiladas 671
Obsolescencia de las AMI AL2 671
Amazon Linux 678
Bottlerocket 688
Ubuntu Linux 694
Windows 695
Estado de los nodos 760
Agente de supervisión de nodos 761

Reparación automática de nodos 761
Problemas de estado de los nodos 764
Cómo ver el estado de los nodos 776
Cómo obtener los registros de nodos 778
Nodos híbridos 781
Características 782
Límites 783
Consideraciones 783
Recursos adicionales 783
Requisitos previos 784
Ejecución de nodos híbridos 854
Configuración 877
Funcionamiento 966
Nodeadm de nodos híbridos 1009
Solución de problemas 1028
Almacenamiento de datos de aplicaciones 1051
Amazon EBS 1051
Consideraciones 1052
Requisitos previos 1052
Paso 1: creación de un rol de IAM 1053
Paso 2: obtención del controlador de CSI de Amazon EBS 1061
Paso 3: implementación de una aplicación de muestra 1062
Amazon EFS 1062
Consideraciones 1062
Requisitos previos 1063
Paso 1: creación de un rol de IAM 1064
Paso 2: obtención del controlador de CSI de Amazon EFS 1070
Paso 3: creación de un sistema de archivos de Amazon EFS 1070
Paso 4: implementación de una aplicación de muestra 1070
Amazon FSx para Lustre 1071
Implementación del controlador 1071
Optimización (EFA) 1079
Optimización (sin EFA) 1095
Amazon FSx para NetApp ONTAP 1103
Amazon FSx para OpenZFS 1104
Amazon File Cache 1104

Mountpoint para Amazon S3 1105
Consideraciones 1105
Implementación del controlador 1106
Cómo eliminar el controlador 1116
Controlador de instantáneas CSI 1118
Configurar redes 1119
Cómo agregar una subred de VPC existente a un clúster de Amazon EKS desde la consola de administración 1119
Requisitos de VPC y subred 1119
Requisitos y consideraciones de la VPC 1120
Requisitos y consideraciones de la subred 1122
Requisitos y consideraciones de la subred compartida 1129
Creación de una VPC 1130
Requisitos previos 53
Subredes públicas y privadas 1131
Solo subredes públicas 1133
Solo subredes privadas 1135
Requisitos del grupo de seguridad 1137
Grupo de seguridad de clúster predeterminado 1137
Restringir el tráfico del clúster 1139
Grupos de seguridad compartidos 1140
Administración de complementos de red 1140
Complementos incorporados 1141
Complementos de red de AWS opcionales 1142
CNI de Amazon VPC 1142
Complementos de CNI alternativos 1269
Multus 1270
AWS Controlador del equilibrador de carga de 1271
CoreDNS 1290
kube-proxy 1309
Cargas de trabajo 1315
Implementación de muestra (Linux) 1315
Requisitos previos 53
Creación de un espacio de nombres de 1316
Cree una implementación de Kubernetes 1316
Crear un servicio 1318

Revise los recursos creados 1319
Ejecute un intérprete de comandos en un pod 1322
Siguientes pasos 1323
Implementación de muestra (Windows) 1323
Requisitos previos 53
Creación de un espacio de nombres de 1316
Cree una implementación de Kubernetes 1316
Crear un servicio 1318
Revise los recursos creados 1327
Ejecute un intérprete de comandos en un pod 1322
Siguientes pasos 1331
Escalador automático vertical de pods 1331
Implementar el escalador automático vertical de pods 1331
Comprobar la instalación del escalador automático vertical de pods 1333
Escalador automático de pods horizontales 1337
Ejecutar una aplicación de prueba del escalador automático de pods horizontales 1338
Equilibrio de carga de red 1341
Requisitos previos 53
Consideraciones 104
Crear un equilibrador de carga de red 1345
(Opcional) Implementación de una aplicación de muestra 1347
Equilibrio de carga de aplicaciones 1351
Requisitos previos 53
Reutilice los ALB con grupos de entrada 1354
(Opcional) Implementación de una aplicación de muestra 1356
Restricción del servicio de IP externas 1359
Copiar una imagen en un repositorio 1361
Visualización de los registros de imágenes de Amazon 1365
Complementos de Amazon EKS 1368
Consideraciones 1369
Espacio de nombres personalizado para complementos 1370
Consideraciones para el modo automático de Amazon EKS 1371
Soporte 1372
Complementos de AWS 1374
Complementos de la comunidad 1396
Complementos del Marketplace 1403

Cómo crear un complemento 1420
Cómo actualizar un complemento 1434
Comprobación de la compatibilidad 1442
Cómo eliminar un complemento 1444
Roles de IAM 1447
Campos que puede personalizar 1454
Verificación de imágenes de contenedor 1458
Capacidades de EKS 1459
Capacidades disponibles 1460
Controladores para Kubernetes (ACK) de AWS 1460
Argo CD 1460
kro (Kube Resource Orchestrator) 1460
Ventajas de las capacidades de EKS 1461
Precios 1462
Cómo funcionan las capacidades de EKS 1462
Casos de uso común 1463
Uso de capacidades 1465
Recursos de capacidades de EKS 1465
Descripción de los estados de las capacidades 1465
Creación de capacidades 1467
Enumeración de capacidades 1467
Descripción de una capacidad 1469
Actualización de la configuración de una capacidad 1470
Eliminación de una capacidad 1471
Siguientes pasos 99
Recursos de Kubernetes 1472
Recursos de Argo CD 1472
Recursos de kro 1473
Recursos de ACK 1475
Límites de recursos 1476
Siguientes pasos 99
Consideraciones 1477
RBAC de Kubernetes y roles de IAM de la capacidad 1477
Patrones de arquitectura de varios clústeres 1478
Comparación de las capacidades de EKS con las soluciones autoadministradas 1480
Controladores de AWS para Kubernetes 1483

Cómo funciona ACK ... 1483
Beneficios de ACK ... 1484
Servicios de AWS compatibles ... 1485
Integración con otras capacidades administradas de EKS ... 1485
Introducción a ACK ... 1486
Creación de una capacidad de ACK ... 1486
Conceptos de ACK ... 1496
Configuración de permisos ... 1504
Consideraciones ... 1510
Solución de problemas ... 1517
Comparación con autoadministrado ... 1522
Argo CD ... 1523
Cómo funciona Argo CD ... 1524
Ventajas de Argo CD ... 1525
Integración con AWS Identity Center ... 1526
Integración con otras capacidades administradas de EKS ... 1485
Introducción a Argo CD ... 1526
Creación de la capacidad Argo CD ... 1526
Conceptos de Argo CD ... 1537
Configuración de permisos ... 1545
Uso de Argo CD ... 1553
Consideraciones sobre Argo CD ... 1591
Solución de problemas ... 1600
Comparación con autoadministrado ... 1605
kro ... 1610
Cómo funciona kro ... 1610
Beneficios de kro ... 1611
Integración con otras capacidades administradas de EKS ... 1485
Introducción a kro ... 1612
Creación de una capacidad de kro ... 1613
Conceptos de kro ... 1624
Configuración de permisos ... 1631
Consideraciones ... 1636
Solución de problemas ... 1642
Comparación con autoadministrado ... 1649
Solución de problemas de capacidades ... 1650

Enfoque general para la solución de problemas 1650
Comprobación de estado de la capacidad 1651
Estados de capacidad comunes 1652
Comprobación del estado de los recursos de Kubernetes 1652
Revisión de los permisos de IAM y el acceso al clúster 1653
Solución de problemas específicos de la capacidad 1655
Problemas comunes en todas las capacidades 1655
Siguientes pasos 99
Administración de clústeres 1657
Supervisión de costos 1657
Visualización de costos por pod 1658
Instalación de Kubecost 1659
Acceso al panel de Kubecost 1660
Más información sobre Kubecost 1662
Servidor de métricas 1671
Consideraciones 104
Implementación como complemento de la comunidad con complementos de Amazon EKS 1672
Implementación con manifiesto 1673
Implementación de aplicaciones con Helm 1674
Etiquetado de recursos 1676
Conceptos básicos de etiquetas 1676
Etiquetado de recursos 1677
Restricciones de las etiquetas 1678
Etiquetado de los recursos para facturación 1678
Uso de etiquetas mediante la consola 1679
Uso de etiquetas mediante la CLI, la API o `eksctl` 1680
Service Quotas 1682
Consulta de las Service Quotas de EKS en la Consola de administración de AWS 1682
Consulta de las cuotas de servicio de EKS con AWS CLI 1683
Cuotas de servicio de Amazon EKS 1684
Cuotas de servicio de AWS Fargate 1684
Seguridad 1686
Prácticas recomendadas 1687
Análisis de vulnerabilidades 1688

Referencia del Center for Internet Security (CIS, Centro para la seguridad de Internet) para Amazon EKS ... 1688
Versiones de la plataforma de Amazon EKS ... 1688
Lista de vulnerabilidades del sistema operativo ... 1689
Detección de nodos con Amazon Inspector ... 1689
Detección de clústeres y nodos con Amazon GuardDuty ... 1689
Validación de la conformidad ... 1689
Consideraciones sobre EKS ... 1691
Seguridad de la infraestructura ... 1691
Resiliencia ... 1696
Prevención de la sustitución confusa entre servicios ... 1697
Consideraciones para Kubernetes ... 1699
Firma de certificados ... 1699
Roles y usuarios de predeterminados ... 1702
Habilitación del cifrado de secretos ... 1707
AWS Secrets Manager ... 1712
Cifrado de sobre predeterminado para todos los datos de la API de Kubernetes ... 1712
Consideraciones para EKS automático ... 1720
Seguridad y autenticación de la API ... 1721
Seguridad de la red ... 1721
Seguridad de las instancias administradas por EC2 ... 1722
Administración automatizada de los recursos ... 1723
Prácticas recomendadas de seguridad ... 1724
Consideraciones para las capacidades de EKS ... 1724
Rol de IAM de capacidad ... 1725
Entradas de acceso de EKS ... 1725
Permisos de Kubernetes adicionales ... 1727
Permisos de IAM que necesita la capacidad ... 1729
Prácticas recomendadas de seguridad ... 1730
Siguientes pasos ... 99
Referencia de IAM ... 1733
Público ... 1733
Autenticación con identidades ... 1734
Administración del acceso con políticas ... 1737
Amazon EKS e IAM ... 1740
Políticas basadas en identidades ... 1745

Roles vinculados a servicios 1752
Rol de IAM de ejecución de pods 1770
Rol de IAM conector 1775
Políticas administradas de AWS 1779
Solución de problemas 1809
Rol de IAM de clúster 1812
Rol de IAM de nodo 1816
Rol de IAM del clúster en modo automático 1822
Rol de IAM de nodo del modo automático 1825
Rol de IAM de capacidad 1828
Supervisión de clústeres 1835
Supervisión y registro en Amazon EKS 1836
Herramientas de supervisión y registro en Amazon EKS 1837
Panel de observabilidad 1841
Resumen 1841
Estado del clúster 1842
Supervisión del plano de control 1842
Información sobre clústeres 1844
Problemas de estado de los nodos 1845
Capacidades de EKS 1845
Observabilidad de la red de contenedores 1845
Casos de uso 152
Características 83
Introducción 308
Requisitos previos y notas importantes 1848
Uso de la AWS CLI, la API de EKS y la API de NFM 1849
Uso de la infraestructura como código (IaC) 1851
¿Cómo funciona? 204
Consideraciones y limitaciones 1861
Métricas de Prometheus 1862
Paso 1: activación de métricas de Prometheus 1863
Paso 2: uso de métricas de Prometheus 1865
Paso 3: administración de raspadores de Prometheus 1865
Implementación mediante Helm 1866
Plano de control 1869
Amazon CloudWatch 1877

Métricas básicas en Amazon CloudWatch 1877
Amazon CloudWatch Observability Operator 1887
Registros del plano de control 1887
Habilitación o deshabilitación de los registros del plano de control 1889
Visualización de registros del plano de control del clúster 1891
AWS CloudTrail 1893
Referencias 1893
Entradas de archivos de registro 1894
Métricas de grupo de Auto Scaling 1897
Operator ADOT 1898
Panel de Amazon EKS 1898
¿Qué es el panel de Amazon EKS? 1898
¿Cómo utiliza el panel AWS Organizations? 1899
Activación del panel de control de EKS mediante la consola de AWS 1900
Vista del panel de EKS 1901
Configuración del panel 1901
Desactivación del panel de EKS mediante la consola de AWS 1903
Solución de problemas con el panel 1903
Configuración de las organizaciones 1904
Trabajar con otros servicios 1910
AWS Backup 1910
AWS CloudFormation 1911
Plantillas de Amazon EKS y AWS CloudFormation 1911
Obtenga más información sobre AWS CloudFormation 1912
AWS CodeConnections 1912
Uso de AWS CodeConnections con Argo CD 1912
Consideraciones sobre el uso de CodeConnections con Argo CD 1913
Amazon Detective 1914
Uso de Amazon Detective con Amazon EKS 1915
Amazon GuardDuty 1915
Zonas locales de AWS 1917
Centro de resiliencia de AWS 1917
AWS Secrets Manager 1918
Uso de AWS Secrets Manager con Argo CD 1918
Amazon Security Lake 1920
Ventajas de usar Security Lake con Amazon EKS 1920

Habilitación de Security Lake en Amazon EKS ... 1921
Análisis de los registros de EKS en Security Lake ... 1921
Amazon VPC Lattice ... 1922
Amazon EKS Connector ... 1923
Consideraciones sobre Amazon EKS Connector ... 1923
Roles de IAM necesarios para Amazon EKS Connector ... 1924
Conexión de un clúster ... 1924
Consideraciones ... 1925
Requisitos previos ... 1925
Paso 1: registro del clúster ... 1925
Paso 2: Instalación del agente de `eks-connector` ... 1928
Pasos a seguir a continuación ... 1930
Concesión de acceso a clústeres ... 1930
Requisitos previos ... 1930
Anulación del registro de un clúster ... 1932
Anulación del registro del clúster de Kubernetes ... 1932
Limpieza de los recursos del clúster de Kubernetes ... 1933
Solución de problemas del conector de EKS ... 1934
Solución de problemas básicos ... 1934
Problema de Helm: 403 Prohibido ... 1936
Error de la consola: El clúster está atascado en el estado Pending ... 1936
Error de la consola: El usuario system:serviceaccount:eks-connector:eks-connector no puede suplantar los usuarios de recursos en el grupo de la API en el ámbito del clúster. ... 1937
Error de la consola: […] está prohibido: el usuario […] no puede publicar el recurso […] en grupo de la API en el ámbito del clúster ... 1937
Error de la consola: Amazon EKS no puede comunicarse con el servidor de la API del clúster de Kubernetes. El clúster debe estar en estado ACTIVE (ACTIVO) para que la conexión se establezca de forma correcta. Intente establecer la conexión en unos minutos. ... 1938
Los pods de Amazon EKS Connector se están bloqueando en bucle ... 1939
Error al iniciar eks-connector: InvalidActivation ... 1939
El nodo del clúster no tiene conectividad de salida ... 1940
Pods de Amazon EKS Connector con el estado `ImagePullBackOff` ... 1941
Preguntas frecuentes ... 1941
Consideraciones de seguridad ... 1942
Responsabilidades de AWS ... 1943

Responsabilidades del cliente 1943
Amazon EKS en AWS Outposts 1945
Cuándo usar cada opción de implementación 1945
Comparación de las opciones de implementación 1946
Ejecución de clústeres locales 1949
Regiones de AWS compatibles 1949
Implementación de un clúster local 1949
Versiones de la plataforma de EKS 1961
Creación de una VPC y de subredes 1981
Preparación para las desconexiones 1985
Consideraciones de capacidad 1990
Solución de problemas de clústeres 1993
Nodos 2005
`eksctl` 2007
Consola de administración de AWS 2008
IA y ML en EKS 2015
¿Por qué elegir EKS para IA/ML? 2015
Casos de uso clave 2016
Casos prácticos 2017
Comience a utilizar el machine learning en EKS 2018
Inferencia en tiempo real 2018
Creación de un clúster 2018
Inferencia de LLM con vLLM 2057
Configuración del clúster 2075
Uso de AMI de GPU de Linux de EKS 2076
Instalación del complemento para dispositivos 2081
Configuración de las AMI de GPU de Windows 2087
Configuración del entrenamiento de clústeres con EFA 2094
Configuración de clústeres de inferencia con Inferentia 2105
Administración de la capacidad 2112
Reserva de GPU para grupos de nodos administrados 2112
Reserva de GPU para nodos autoadministrados 2115
Uso de P6e-GB200 con EKS 2119
Recetas 2130
Limitación para que los pods no se programen en nodos específicos 2130
Recursos 2132

Talleres 2133
prácticas recomendadas 2133
Arquitecturas de referencia 2135
Tutoriales 2135
Herramientas 2137
Protocolo de contexto para modelos (MCP) 2138
Descripción general 97
Ejemplos de integraciones 2139
Introducción 308
Introducción 2140
Herramientas 2157
Amazon Q 2169
Funcionamiento 514
Uso de Amazon Q para la solución de problemas 2170
Consideraciones 104
Más información 2171
Control de versiones 2172
Versiones de Kubernetes 2172
Versiones disponibles con soporte estándar 2173
Versiones disponibles con soporte extendido 2173
Calendario de lanzamiento de Amazon EKS Kubernetes 2173
Obtenga información sobre la versión con AWS CLI 2174
Preguntas frecuentes sobre las versiones de Amazon EKS 2176
Preguntas frecuentes sobre el soporte extendido de Amazon EKS 2178
Versiones con soporte estándar 2181
Versiones con soporte extendido 2186
Visualización del periodo de soporte 2189
Visualización de la política de actualización 2190
Habilitación del soporte extendido 2192
Deshabilitación del soporte extendido 2193
Versiones de la plataforma 2194
Versión de Kubernetes 1.34 2196
Versión de Kubernetes 1.33 2197
Versión de Kubernetes 1.32 2200
Versión de Kubernetes 1.31 2205
Versión de Kubernetes 1.30 2210

Versión de Kubernetes 1.29 ... 2217
Obtenga la versión actual de la plataforma ... 2225
Cambio de la versión de la plataforma ... 2226
Solución de problemas ... 2227
Capacidad insuficiente ... 2227
Los nodos no pueden unirse al clúster ... 2227
Acceso denegado o no autorizado (kubectl) ... 2229
hostname doesn't match ... 2230
getsockopt: no route to host ... 2230
Instances failed to join the Kubernetes cluster ... 2231
Códigos de error del grupo de nodos administrado ... 2231
Not authorized for images ... 2236
El nodo está en estado NotReady ... 2236
Recopilador del registro de EKS ... 2237
La red de tiempo de ejecución del contenedor no está lista ... 2238
Tiempo de espera de protocolo de enlace TLS ... 2239
InvalidClientTokenId ... 2240
Los grupos de nodos deben coincidir con la versión de Kubernetes antes de actualizar el plano de control ... 2240
Al lanzar muchos nodos, hay errores de Too Many Requests ... 2241
Respuesta de error no autorizada HTTP 401 en solicitudes del servidor API de Kubernetes ... 2241
La versión de la plataforma Amazon EKS está más de dos versiones por detrás de la versión de plataforma actual ... 2242
Preguntas frecuentes sobre el estado de los clústeres y los códigos de error con rutas de resolución ... 2245
Proyectos relacionados con Amazon EKS ... 2251
Soporte del software implementado en EKS ... 2251
Herramientas de administración ... 2252
eksctl ... 2252
Controladores de AWS para Kubernetes ... 2252
kro (Kube Resource Orchestrator) ... 2253
Argo CD ... 2253
Flux CD ... 2253
CDK para Kubernetes ... 2253
Red ... 2254
Complemento CNI de Amazon VPC para Kubernetes ... 2254

AWSControlador del balanceador de carga de para Kubernetes 2254
ExternalDNS 2254
Machine learning 2254
Kubeflow 2255
Auto Scaling 2255
Escalador automático del clúster 2255
Karpenter 2255
Escalador 2255
Supervisión 2256
Prometheus 2256
Integración continua/implementación continua 2256
Jenkins X 2256
Nuevas características y hoja de ruta 2257
Historial de documentos 2258
Cómo contribuir 2323
Cómo editar una sola página 2323
Requisitos previos 53
Procedimiento 107
Información general de la solicitud de extracción 2325
Edición de archivos con GitHub 2326
Requisitos previos 53
Procedimiento 107
Visualización de comentarios de estilo 2327
Instalación de Vale 2328
Instalación de la extensión de Vale de VS Code 2328
Sincronización de Vale 2329
Visualización de comentarios de estilo en VS Code 2329
Cómo crear la página 2329
Cómo crear la página 2329
Cómo agregar la página a la navegación 2330
Cómo insertar el enlace 2330
Enlazar a una página o sección de la Guía del usuario de EKS 2331
Enlace a otra guía de la documentación de AWS 2331
Enlace a una página web externa 2331
Cómo crear con Amazon Q 2332
Instale Amazon Q con VS Code 2332

Inicie sesión en Amazon Q 2333
Cómo utilizar Amazon Q para crear contenido 2333
Consejos 2333
Visualización de una vista previa de la solicitud de extracción 2334
Visualización de una vista previa 2335
Limitaciones de la versión preliminar 2335
Sintaxis de AsciiDoc 2336
Nueva página 2336
Encabezados 2336
Formato de texto 2336
Listas 2337
Enlaces 2337
Ejemplos de código 2337
Imágenes 2338
Tablas 2338
Avisos 2338
Inclusión de otros archivos 2339
Atributos (similares a las entidades) 2339
Procedimientos 2339

Ayude a mejorar esta página

Para contribuir a esta guía del usuario, elija el enlace Edit this page on GitHub que se encuentra en el panel derecho de cada página.

¿Qué es Amazon EKS?

Amazon EKS: administración simplificada de Kubernetes

Amazon Elastic Kubernetes Service (EKS) es un servicio de Kubernetes completamente administrado que elimina la complejidad que supone operar clústeres de Kubernetes. Con EKS, es posible:

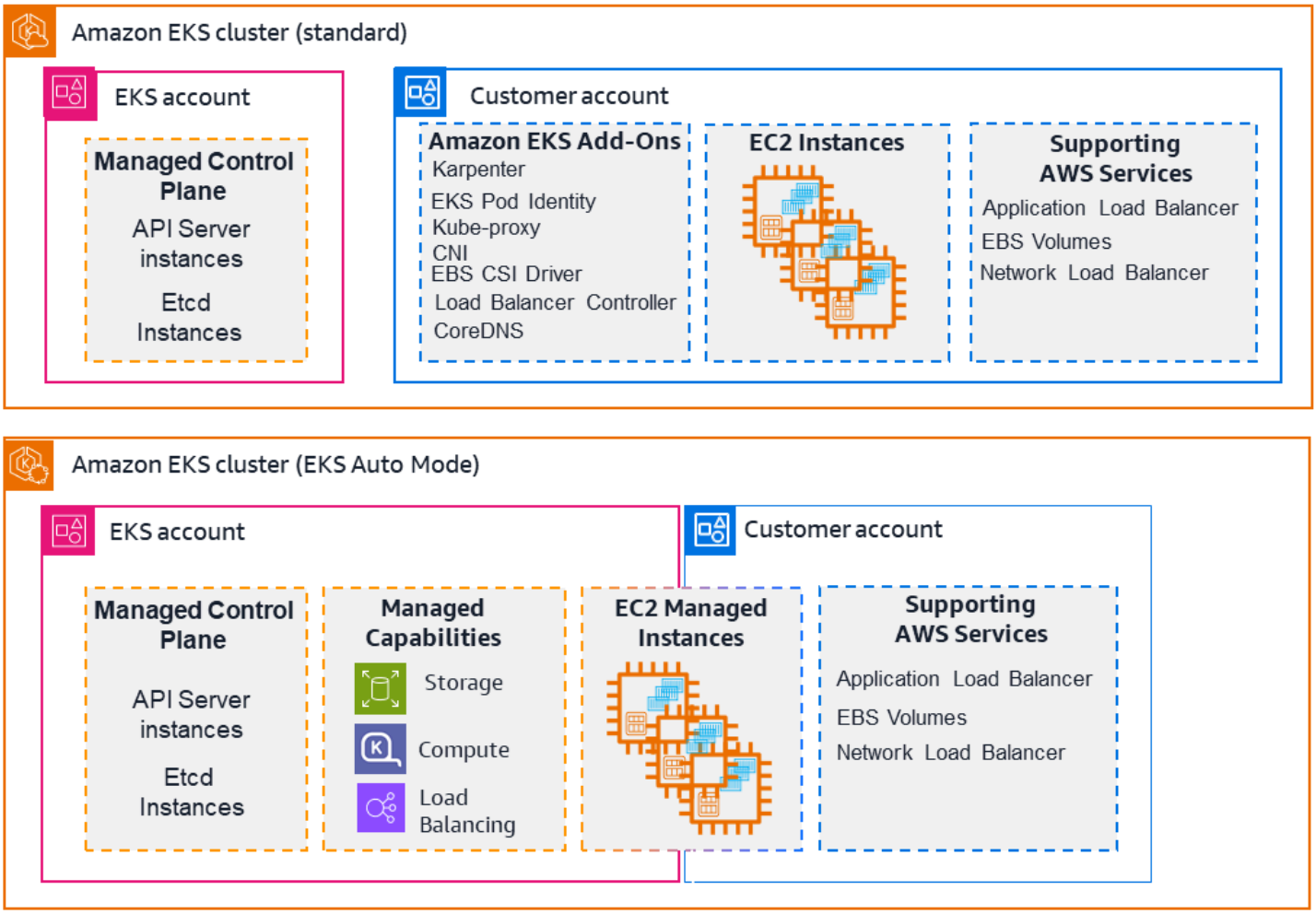
- Implementar aplicaciones más rápido y con menos sobrecarga operativa
- Escalar sin interrupciones para satisfacer las demandas cambiantes de las cargas de trabajo
- Mejorar la seguridad mediante la integración con AWS y actualizaciones automatizadas
- Puede elegir entre EKS estándar o el modo automático de EKS totalmente automatizado

Amazon Elastic Kubernetes Service (Amazon EKS) es la plataforma principal para ejecutar clústeres de Kubernetes, tanto en la nube de Amazon Web Services (AWS) como en sus propios centros de datos (EKS Anywhere y Nodos híbridos de Amazon EKS).

Amazon EKS facilita la creación, protección y administración de clústeres de Kubernetes. Puede resultar más rentable para proporcionar los recursos necesarios durante los picos de demanda que mantener centros de datos propios. Estas son dos de las principales formas de utilizar Amazon EKS:

- EKS estándar: AWS administra el plano de control de Kubernetes cuando se crea un clúster con EKS. Los componentes que administran los nodos, programan las cargas de trabajo, se integran con la nube de AWS y almacenan y escalan la información del plano de control para mantener los clústeres en funcionamiento se administran automáticamente en su nombre.
- Modo automático de EKS: cuando se utiliza la característica Modo automático de EKS, EKS amplía su control para administrar también los nodos (plano de datos de Kubernetes). Simplifica la administración de Kubernetes al aprovisionar automáticamente la infraestructura, seleccionar las instancias de computación óptimas, escalar dinámicamente los recursos, optimizar continuamente los costos, aplicar revisiones a los sistemas operativos e integrarse con los servicios de seguridad de AWS.

El siguiente diagrama ilustra cómo Amazon EKS integra los clústeres de Kubernetes con la nube de AWS, según el método de creación de clúster que elija:



Amazon EKS lo ayuda a eliminar la fricción y acelerar el tiempo de producción, mejora el rendimiento, la disponibilidad y la resiliencia, y refuerza la seguridad del sistema. Para obtener más información, consulte Amazon Elastic Kubernetes Service.

Creación y escalado con Kubernetes: capacidades de Amazon EKS

Amazon EKS no solo lo ayuda a crear y administrar clústeres, sino que también lo ayuda a crear y escalar sistemas de aplicaciones con Kubernetes. Las capacidades de Amazon EKS son servicios de clústeres completamente administrados que amplían la funcionalidad del clúster con herramientas nativas de Kubernetes sin intervención manual, como las siguientes:

- Argo CD: Argo CD ofrece una implementación continuo declarativa y basada en GitOps para sus cargas de trabajo, recursos de AWS e infraestructura en la nube.

- Controladores de AWS para Kubernetes (ACK): ACK permite la creación nativa de Kubernetes y la administración del ciclo de vida de los recursos de AWS, lo que unifica la orquestación de las cargas de trabajo y los flujos de trabajo basados en la infraestructura como código.
- kro (Kube Resource Orchestrator): kro amplía las características nativas de Kubernetes para simplificar la creación, la orquestación y la composición de recursos personalizados, y le proporciona las herramientas para crear sus propios componentes de nube personalizados.

Las capacidades de EKS son recursos en la nube que minimizan la carga operativa que implica instalar, mantener y escalar estos componentes fundamentales de la plataforma en sus clústeres, lo que le permite centrarse en la creación de software en lugar de las operaciones de plataformas del clúster.

Para obtener más información, consulte Capacidades de EKS.

Características de Amazon EKS

Amazon EKS ofrece las siguientes características de nivel alto:

Interfaces de administración

EKS admite una variedad de interfaces para aprovisionar, administrar y mantener clústeres. Algunas de estas son Consola de administración de AWS, las API y los SDK de Amazon EKS, CDK, AWS CLI, la CLI de eksctl, AWS CloudFormation y Terraform. Para obtener más información, consulte Introducción y Configuración de los clústeres.

Herramientas de control de acceso

EKS se basa en las características de Kubernetes y de AWS Identity and Access Management (AWS IAM) para administrar el acceso de los usuarios y las cargas de trabajo. Para obtener más información, consulte the section called “Acceso a la API de Kubernetes” y the section called “Acceso de la carga de trabajo a AWS ”.

Recursos de computación

En el caso de los recursos de computación, EKS permite utilizar la amplia gama de tipos de instancias de Amazon EC2 e innovaciones de AWS, como Nitro y Graviton con Amazon EKS para optimizar la computación de las cargas de trabajo. Para obtener más información, consulte Administración de la computación.

Almacenamiento

El Modo automático de EKS crea de forma automática clases de almacenamiento a partir de volúmenes de EBS. Mediante los controladores de la interfaz de almacenamiento de contenedores (CSI), también puede utilizar Amazon S3, Amazon EFS, Amazon FSx y Amazon File Cache para satisfacer las necesidades de almacenamiento de la aplicación. Para obtener más información, consulte Almacenamiento de datos de aplicaciones.

Seguridad

En Amazon EKS se aplica el modelo de responsabilidad compartida en lo relacionado con la seguridad. Para obtener más información, consulte Prácticas recomendadas de seguridad, Seguridad de la infraestructura y Seguridad de Kubernetes.

Herramientas de supervisión

Utilice el panel de observabilidad para supervisar los clústeres de Amazon EKS. Algunas de las herramientas de supervisión son Prometheus, CloudWatch, CloudTrail y el operador de ADOT. Para obtener más información sobre los paneles, los servidores de métricas y otras herramientas, consulte Costos de clústeres de EKS y Servidor de métricas de Kubernetes.

Capacidades del clúster

EKS proporciona capacidades de clúster administradas para la implementación continua, la administración de los recursos en la nube y la composición de los recursos basadas en innovaciones de código abierto. EKS instala las API de Kubernetes en los clústeres, pero los controladores y otros componentes se ejecutan en EKS y se administran completamente, lo que proporciona parches, escalado y supervisión automatizados. Para obtener más información, consulte Capacidades de EKS.

Compatibilidad y soporte de Kubernetes

Amazon EKS cuenta con la certificación de conformidad con Kubernetes, por lo que puede implementar aplicaciones compatibles con Kubernetes sin necesidad de refactorizarlas y utilizar herramientas y complementos de la comunidad de Kubernetes. EKS ofrece tanto soporte estándar como soporte ampliado para Kubernetes. Para obtener más información, consulte the section called “Versiones de Kubernetes”.

Servicios relacionados

Servicios para utilizar con Amazon EKS

Puede utilizar otros servicios de AWS con los clústeres que implemente mediante Amazon EKS:

Amazon EC2

Obtenga capacidad de computación escalable y bajo demanda con Amazon EC2.

Amazon EBS

Asocie recursos de almacenamiento en bloque escalables y de alto rendimiento con Amazon EBS.

Amazon ECR

Almacene imágenes de contenedores de forma segura con Amazon ECR.

Amazon CloudWatch

Supervise recursos y aplicaciones de AWS en tiempo real con Amazon CloudWatch.

Amazon Prometheus

Realice un seguimiento de las métricas de las aplicaciones en contenedores con Amazon Managed Service para Prometheus.

Elastic Load Balancing

Distribuya el tráfico entrante entre múltiples destinos con Elastic Load Balancing.

Amazon GuardDuty

Detecte amenazas contra los clústeres de EKS con Amazon GuardDuty.

AWS Resilience Hub

Evalúe la resiliencia de los clústeres de EKS con AWS Resilience Hub.

Precios de Amazon EKS

Amazon EKS ofrece precios por clúster, basados en el soporte de versiones de clústeres de Kubernetes, precios para el modo automático de Amazon EKS y precios por vCPU para los Nodos híbridos de Amazon EKS.

Al utilizar Amazon EKS, se paga por separado los recursos de AWS que utiliza para ejecutar las aplicaciones en los nodos de trabajo de Kubernetes. Por ejemplo, si ejecuta nodos de trabajo de Kubernetes como instancias de Amazon EC2 con volúmenes de Amazon EBS y direcciones IPv4

públicas, se cobrará la capacidad de la instancia a través de Amazon EC2, la capacidad del volumen a través de Amazon EBS y la dirección IPv4 a través de Amazon VPC.

Visite las páginas de precios correspondientes a los servicios de AWS que utiliza con las aplicaciones de Kubernetes para obtener información detallada sobre los precios.

- Para ver los precios de los clústeres de Amazon EKS, el modo automático de Amazon EKS, las capacidades de Amazon EKS y los Nodos híbridos de Amazon EKS, consulte Precios de Amazon EKS.
- Para conocer los precios de Amazon EC2, consulte Precios de las instancias bajo demanda de Amazon EC2 y Precios de instancias de spot de Amazon EC2.
- Para conocer los precios de AWS Fargate, consulte Precios de AWS Fargate.
- Puede utilizar savings plans para la computación que se utiliza en los clústeres de Amazon EKS. Para obtener más información, consulte Precio con Savings Plans.

Casos de uso comunes en Amazon EKS

Amazon EKS ofrece sólidos servicios administrados de Kubernetes en AWS que se han diseñado para optimizar las aplicaciones en contenedores. A continuación, se incluyen algunos de los casos de uso más comunes de Amazon EKS y le ayuda a aprovechar sus puntos fuertes para sus necesidades específicas.

Implementación de aplicaciones de alta disponibilidad

: Al usar Elastic Load Balancing, puede asegurarse de que sus aplicaciones estén altamente disponibles en varias zonas de disponibilidad.

Creación de arquitectura de microservicios

: Utilice las características de detección de servicios de Kubernetes con AWS Cloud Map o Amazon VPC Lattice para crear sistemas resilientes.

Automatización de los procesos de lanzamiento de software

: Administre las canalizaciones de integración e implementación continuos (CI/CD) que simplifican el proceso de creación, prueba e implementación automatizadas de aplicaciones. Para una implementación continua declarativa, consulte Implementación continua con Argo CD.

Ejecución de aplicaciones sin servidor

Utilice AWS Fargate con Amazon EKS para ejecutar aplicaciones sin servidor. Esto significa que puede centrarse únicamente en el desarrollo de aplicaciones, mientras que Amazon EKS y Fargate se encargan de la infraestructura subyacente.

Ejecución de cargas de trabajo de machine learning

Amazon EKS es compatible con los marcos de machine learning más populares, como TensorFlow, MXNet y PyTorch. Gracias a la compatibilidad con la GPU, puede gestionar incluso tareas complejas de machine learning de forma eficaz.

Implementación coherente en las instalaciones y en la nube

Para simplificar la ejecución de Kubernetes en entornos en las instalaciones, puede utilizar los mismos clústeres, características y herramientas de Amazon EKS para ejecutar nodos autoadministrados en AWS Outposts o puede emplear los Nodos híbridos de Amazon EKS con una infraestructura propia. En el caso de los entornos autónomos y aislados, puede utilizar Amazon EKS Anywhere para automatizar la administración del ciclo de vida de los clústeres de Kubernetes en una infraestructura propia.

Ejecución de cargas de trabajo rentables de procesamiento por lotes y macrodatos

Utilice instancias de spot para ejecutar sus cargas de trabajo de procesamiento por lotes y macrodatos, como Apache Hadoop y Spark, por una fracción del costo. Esto le permite aprovechar la capacidad no utilizada de Amazon EC2 a precios reducidos.

Administración de recursos de AWS de Kubernetes

Utilice Controladores de AWS para Kubernetes (ACK) para crear y administrar recursos de AWS directamente desde su clúster de Kubernetes mediante las API nativas de Kubernetes.

Creación de abstracciones de ingeniería de plataformas

Cree API de Kubernetes personalizadas que componen varios recursos en abstracciones de nivel superior mediante kro (Kube Resource Orchestrator).

Protección de la aplicación y garantía del cumplimiento

Implemente prácticas de seguridad sólidas y mantenga el cumplimiento con Amazon EKS, que se integra con servicios de seguridad de AWS como AWS Identity and Access Management (IAM), Amazon Virtual Private Cloud (Amazon VPC) y AWS Key Management Service (AWS KMS). Esto garantiza la privacidad y la protección de los datos según los estándares del sector.

Arquitectura de Amazon EKS

Amazon EKS se alinea con la arquitectura general de clústeres de Kubernetes. Para obtener más información, consulte Kubernetes Components en la documentación de Kubernetes. En las siguientes secciones se resumen algunos detalles adicionales de la arquitectura de Amazon EKS.

Plano de control

Amazon EKS garantiza que cada clúster tenga su propio plano de control de Kubernetes. Este diseño mantiene la infraestructura de cada clúster separada, sin superposiciones entre los clústeres o cuentas de AWS. La configuración incluye:

Componentes distribuidos

: El plano de control coloca al menos dos instancias de servidor de la API y tres instancias etcd que se ejecutan en tres zonas de disponibilidad de AWS dentro de una región de AWS.

Rendimiento óptimo

: Amazon EKS supervisa y ajusta activamente las instancias del plano de control para mantener el máximo rendimiento.

Resiliencia

: Si una instancia del plano de control falla, Amazon EKS la reemplaza rápidamente y utiliza una zona de disponibilidad diferente si es necesario.

Tiempo de actividad consistente

: Al ejecutar clústeres en varias zonas de disponibilidad, se ha conseguido un Acuerdo de nivel de servicio (SLA) de disponibilidad de puntos de conexión de servidores API.

Amazon EKS utiliza Amazon Virtual Private Cloud (Amazon VPC) para limitar el tráfico entre los componentes del plano de control dentro de un único clúster. Los componentes del clúster no pueden ver ni recibir comunicaciones de otros clústeres u otras cuentas de AWS, excepto según lo autorizado por las políticas de control de acceso basado en roles (RBAC) de Kubernetes.

Computación

Además del plano de control, un clúster de Amazon EKS tiene un conjunto de máquinas de trabajo denominadas nodos. La selección del tipo de nodo de clúster de Amazon EKS adecuado

es fundamental para cumplir sus requisitos específicos y optimizar la utilización de los recursos. Amazon EKS ofrece los siguientes tipos de nodos principales:

Modo automático de EKS

El modo automático de EKS amplía la administración de AWS más allá del plano de control para incluir el plano de datos, con lo que se automatiza la administración de la infraestructura del clúster. Integra las principales funciones de Kubernetes como componentes integrados, incluidos el escalado automático de computación, las redes, el equilibrio de carga, el DNS, el almacenamiento y la compatibilidad con GPU. El modo automático de EKS administra dinámicamente los nodos en función de las demandas de la carga de trabajo, a través de AMI inmutables con características de seguridad mejoradas. Automatiza las actualizaciones y mejoras sin descuidar los presupuestos de interrupción de los pods, e incluye componentes administrados que, de otro modo, requerirían una administración adicional. Esta opción es idónea para los usuarios que desean aprovechar la experiencia y los conocimientos de AWS en las operaciones cotidianas, minimizar los gastos operativos y centrarse en el desarrollo de aplicaciones en lugar de en la administración de la infraestructura.

AWS Fargate

Fargate es un motor de procesamiento sin servidor para contenedores que elimina la necesidad de administrar las instancias subyacentes. Con Fargate, usted especifica las necesidades de recursos de su aplicación y AWS aprovisiona, escala y mantiene automáticamente la infraestructura. Esta opción es ideal para los usuarios que priorizan la facilidad de uso y desean concentrarse en el desarrollo y la implementación de aplicaciones en lugar de en la administración de la infraestructura.

Karpenter

Karpenter es un escalador automático de clústeres de Kubernetes flexible y de alto rendimiento que ayuda a mejorar la disponibilidad de las aplicaciones y la eficiencia del clúster. Karpenter lanza recursos de computación del tamaño correcto en respuesta a los cambios en la carga de la aplicación. Esta opción puede aprovisionar recursos informáticos justo a tiempo que cumplan con los requisitos de su carga de trabajo.

Grupos de nodos administrados

Los grupos de nodos administrados son una combinación de automatización y personalización para administrar un conjunto de instancias de Amazon EC2 dentro de un clúster de Amazon EKS. AWS se encarga de tareas como la aplicación de parches, la actualización y el escalado de los nodos, lo que facilita los aspectos operativos. Paralelamente, se admiten los argumentos

de `kubelet` personalizados, lo que abre la posibilidad de aplicar políticas avanzadas de administración de CPU y memoria. Además, mejoran la seguridad mediante roles de AWS Identity and Access Management (IAM) para las cuentas de servicio y, al mismo tiempo, reducen la necesidad de permisos independientes por clúster.

Nodos autoadministrados

Los nodos autoadministrados ofrecen un control total sobre sus instancias de Amazon EC2 dentro de un clúster de Amazon EKS. Usted se encarga de administrar, escalar y mantener los nodos, lo que le proporciona un control total sobre la infraestructura subyacente. Esta opción es adecuada para los usuarios que necesitan un control detallado y una personalización de sus nodos y que están dispuestos a invertir tiempo en la administración y el mantenimiento de su infraestructura.

Nodos híbridos de Amazon EKS

Con los Nodos híbridos de Amazon EKS, es posible utilizar la infraestructura en las instalaciones y periférica como nodos en los clústeres de Amazon EKS. Los Nodos híbridos de Amazon EKS unifican la administración de Kubernetes en los entornos y delegan la administración del plano de control de Kubernetes en AWS para las aplicaciones en las instalaciones y periféricas.

Capacidades de EKS

Amazon EKS proporciona capacidades de clúster completamente administradas, ya que instala y administra las API de Kubernetes (con definiciones de recursos personalizados de Kubernetes) en el clúster y, al mismo tiempo, opera controladores y otros componentes en una infraestructura de AWS, independiente del clúster. EKS proporciona la aplicación de parches, el escalado y la supervisión de estas capacidades de forma automatizada y administra completamente su ciclo de vida para reducir la carga de operar servicios en el clúster para la orquestación de cargas de trabajo, la administración de recursos de AWS, etc.

EKS proporciona los siguientes tipos de capacidades:

AWS Controladores para Kubernetes de (ACK)

Controladores de AWS para Kubernetes (ACK) le permite administrar los recursos de AWS mediante las API de Kubernetes, lo que le permite definir los buckets de S3, las bases de datos de RDS, los roles de IAM y otros recursos de AWS como recursos personalizados de Kubernetes. Puede administrar los recursos de AWS junto con sus cargas de trabajo de Kubernetes con las mismas herramientas y flujos de trabajo, con soporte para más de 50 servicios de AWS, como S3, RDS, DynamoDB y Lambda.

Argo CD

Argo CD implementa la implementación continua basada en GitOps para las cargas de trabajo de la aplicación, los recursos de AWS y la configuración del clúster con los repositorios de Git como origen de información verdadera. Argo CD sincroniza automáticamente los clústeres con los repositorios de Git, detecta las desviaciones y concilia continuamente para garantizar que las aplicaciones y los recursos implementados coincidan con el estado deseado de control de versiones. Puede utilizar Argo CD para administrar aplicaciones en un clúster determinado, o implementar y administrar aplicaciones en varios clústeres desde un solo recurso de Argo CD, con una implementación automatizada desde los repositorios de Git siempre que se hagan cambios.

kro (Kube Resource Orchestrator)

kro (Kube Resource Orchestrator) le permite crear API de Kubernetes personalizadas que agrupan varios recursos en abstracciones de nivel superior, lo que permite a los equipos de plataformas definir patrones reutilizables para combinaciones de recursos comunes. De este modo, los equipos de plataformas pueden proporcionar capacidades de autoservicio con barreras de protección adecuadas, lo que permite a los desarrolladores aprovisionar infraestructuras complejas mediante API sencillas y personalizadas, sin dejar de cumplir los estándares organizativos y las prácticas recomendadas.

Conceptos de Kubernetes

Amazon Elastic Kubernetes Service (Amazon EKS) es un servicio administrado de AWS basado en un proyecto de código abierto de Kubernetes. Si bien hay cosas que debe saber sobre cómo el servicio Amazon EKS se integra con la nube de AWS (especialmente cuando crea un clúster de Amazon EKS por primera vez), una vez que esté en funcionamiento, utilizará su clúster de Amazon EKS de la misma manera que lo haría con cualquier otro clúster de Kubernetes. Por lo tanto, para comenzar a administrar clústeres de Kubernetes y a implementar las cargas de trabajo, necesita al menos una comprensión básica de los conceptos de Kubernetes.

Esta página divide los conceptos de Kubernetes en tres secciones: the section called “¿Por qué debería elegir Kubernetes?”, the section called “Clústeres” y the section called “Cargas de trabajo”. La primera sección describe el valor de ejecutar un servicio de Kubernetes, en particular como un servicio administrado como Amazon EKS. La sección Cargas de trabajo describe cómo se crean, almacenan, ejecutan y administran las aplicaciones de Kubernetes. La sección Clústeres

describe los diferentes componentes que componen los clústeres de Kubernetes y cuáles son sus responsabilidades a la hora de crear y mantener los clústeres de Kubernetes.

Temas

- ¿Por qué debería elegir Kubernetes?
- Clústeres
- Cargas de trabajo
- Pasos a seguir a continuación

A medida que vaya leyendo este contenido, los enlaces le llevarán a descripciones más detalladas de los conceptos de Kubernetes, tanto en Amazon EKS como en la documentación de Kubernetes, por si quiere profundizar en alguno de los temas aquí tratados. Para obtener más información sobre cómo Amazon EKS implementa las características de computación y del plano de control de Kubernetes, consulte the section called “Arquitectura”.

¿Por qué debería elegir Kubernetes?

Kubernetes se diseñó para mejorar la disponibilidad y la escalabilidad al ejecutar aplicaciones en contenedores esenciales y con calidad de producción. En lugar de ejecutar Kubernetes en una sola máquina (aunque eso es posible), Kubernetes logra esos objetivos al permitir ejecutar aplicaciones en conjuntos de computadoras que pueden expandirse o contraerse para satisfacer la demanda. Kubernetes incluye características que le facilitan lo siguiente:

- Implementar aplicaciones en varios equipos (mediante contenedores implementados en pods)
- Supervisar el estado de los contenedores y reiniciar los que estén defectuosos
- Escalar los contenedores hacia arriba y hacia abajo en función de la carga
- Actualizar los contenedores con nuevas versiones
- Mover recursos entre contenedores
- Equilibrar el tráfico entre las máquinas

Permitir que Kubernetes automatice este tipo de tareas complejas permite a los desarrolladores de aplicaciones concentrarse en la creación y mejora de sus cargas de trabajo, en lugar de preocuparse por la infraestructura. El desarrollador suele crear archivos de configuración, formateados como archivos YAML, que describen el estado deseado de la aplicación. Esto podría incluir qué

contenedores ejecutar, los límites de recursos, el número de réplicas del pod, la asignación de CPU o memoria, las reglas de afinidad, etc.

Atributos de Kubernetes

Para lograr sus objetivos, Kubernetes cuenta con los siguientes atributos:

- En contenedores: Kubernetes es una herramienta de orquestación de contenedores. Para usar Kubernetes, antes debe tener sus aplicaciones en contenedores. Según el tipo de aplicación, puede ser un conjunto de microservicios, trabajos por lotes o de otras formas. De este modo, sus aplicaciones pueden aprovechar un flujo de trabajo de Kubernetes que abarca un enorme ecosistema de herramientas, en el que los contenedores pueden almacenarse como imágenes en un registro de contenedores, implementarse en un clúster de Kubernetes y ejecutarse en un nodo disponible. Puede crear y probar contenedores individuales en su equipo local con Docker u otro tiempo de ejecución de contenedores, antes de implementarlos en su clúster de Kubernetes.
- Escalable: si la demanda de sus aplicaciones supera la capacidad de las instancias en ejecución de esas aplicaciones, Kubernetes podrá escalar verticalmente. Según sea necesario, Kubernetes puede determinar si las aplicaciones requieren más CPU o memoria y responder mediante la ampliación automática de la capacidad disponible o con una mayor capacidad existente. El escalado se puede llevar a cabo en el pod, si hay suficiente computación disponible para ejecutar más instancias de la aplicación (escalado automático horizontal del pod), o en el nodo, si es necesario instalar más nodos para gestionar el aumento de la capacidad (Escalador automático de clústeres o Karpenter). Como la capacidad ya no es necesaria, estos servicios pueden eliminar los pods innecesarios y cerrar los nodos innecesarios.
- Disponible: si una aplicación o un nodo deja de funcionar o no está disponible, Kubernetes puede mover las cargas de trabajo en ejecución a otro nodo disponible. Para forzar el problema, basta con eliminar una instancia en ejecución de una carga de trabajo o un nodo en el que se estén ejecutando sus cargas de trabajo. La conclusión es que las cargas de trabajo se pueden almacenar en otras ubicaciones si ya no se pueden ejecutar donde están.
- Declarativo: Kubernetes utiliza la conciliación activa para verificar constantemente que el estado que se declara para el clúster coincida con el estado real. Al aplicar objetos de Kubernetes a un clúster, normalmente a través de archivos de configuración con formato YAML, puede, por ejemplo, solicitar que se inicien las cargas de trabajo que desea ejecutar en su clúster. Puede cambiar las configuraciones más adelante para llevar a cabo otras acciones, como usar una versión posterior de un contenedor o asignar más memoria. Kubernetes hará lo necesario para establecer el estado deseado. Esto puede incluir activar o desactivar los nodos, detener y reiniciar las cargas de trabajo o extraer contenedores actualizados.

- Compatible con la composición: dado que una aplicación suele constar de varios componentes, es recomendable poder administrar un conjunto de estos componentes (que suelen estar representados por varios contenedores) de forma conjunta. Si bien Docker Compose ofrece una forma de hacerlo directamente con Docker, el comando Kompose de Kubernetes puede ayudarlo a hacerlo con Kubernetes. Consulte Traducir un archivo de Docker Compose a recursos de Kubernetes para ver un ejemplo de cómo hacerlo.
- Extensible: a diferencia del software propietario, el proyecto de Kubernetes de código abierto está diseñado para que pueda ampliar Kubernetes como desee para satisfacer sus necesidades. Las API y los archivos de configuración están abiertos a modificaciones directas. Se recomienda a terceros desarrollar sus propios controladores para ampliar tanto las características de infraestructura como las características de Kubernetes orientadas al usuario final. Los webhooks le permiten configurar reglas de clúster para hacer cumplir las políticas y adaptarlas a las condiciones cambiantes. Para obtener más ideas sobre cómo ampliar los clústeres de Kubernetes, consulte Extending Kubernetes.
- Portátil: muchas organizaciones han estandarizado sus operaciones en Kubernetes porque les permite administrar todas las necesidades de sus aplicaciones de la misma manera. Los desarrolladores pueden usar las mismas canalizaciones para crear y almacenar aplicaciones en contenedores. Luego, esas aplicaciones se pueden implementar en clústeres de Kubernetes que se ejecutan en las instalaciones, en nubes, en terminales de puntos de venta de restaurantes o en dispositivos de IoT dispersos por los sitios remotos de la empresa. Su naturaleza de código abierto permite a las personas desarrollar estas distribuciones de Kubernetes especiales, junto con las herramientas necesarias para administrarlas.

Administración de Kubernetes

El código fuente de Kubernetes está disponible de forma gratuita, por lo que puede instalarlo con su propio equipo y administrar Kubernetes por su cuenta. Sin embargo, autoadministrar Kubernetes requiere una profunda experiencia operativa, y su mantenimiento requiere tiempo y esfuerzo. Por estas razones, la mayoría de las personas que implementan cargas de trabajo de producción eligen un proveedor de nube (como Amazon EKS) o en las instalaciones (como Amazon EKS Anywhere) con su propia distribución de Kubernetes probada y el apoyo de expertos de Kubernetes. Esto le permite librarse de gran parte del trabajo pesado e indiferenciado necesario para el mantenimiento de sus clústeres, que incluye:

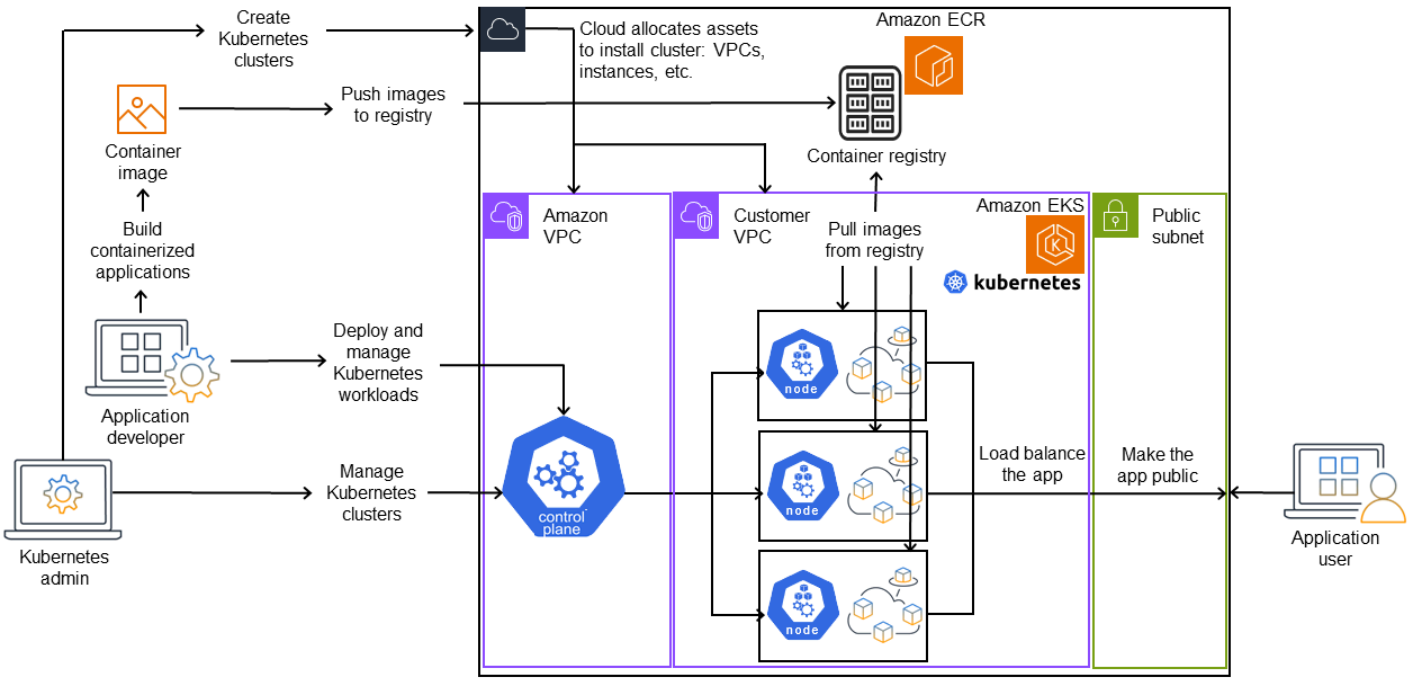
- Hardware: si no tiene hardware disponible para ejecutar Kubernetes según sus necesidades, un proveedor de servicios en la nube como AWS Amazon EKS puede ahorrarle costos iniciales.

Con Amazon EKS, esto significa que puede consumir los mejores recursos de nube que ofrece AWS, incluidas las instancias de computación (Amazon Elastic Compute Cloud), su propio entorno privado (Amazon VPC), la administración central de identidades y permisos (IAM) y el almacenamiento (Amazon EBS). AWS administra las computadoras, las redes, los centros de datos y todos los demás componentes físicos necesarios para ejecutar Kubernetes. Del mismo modo, no tiene que planificar su centro de datos para gestionar la máxima capacidad en los días de mayor demanda. En el caso de Amazon EKS Anywhere u otros clústeres en las instalaciones de Kubernetes, es responsable de administrar la infraestructura utilizada en sus implementaciones de Kubernetes, pero puede confiar en que AWS le ayudará a mantener Kubernetes actualizado.

- Administración del plano de control: Amazon EKS administra la seguridad y la disponibilidad del plano de control de Kubernetes alojado en AWS, que se encarga de programar contenedores, administrar la disponibilidad de las aplicaciones y otras tareas clave, para que pueda centrarse en las cargas de trabajo de sus aplicaciones. Si el clúster se interrumpe, AWS debería disponer de los medios necesarios para restaurarlo a un estado de ejecución. En el caso de Amazon EKS Anywhere, administraría el plano de control.
- Actualizaciones probadas: cuando actualiza sus clústeres, puede confiar en Amazon EKS o Amazon EKS Anywhere para proporcionarle versiones probadas de sus distribuciones de Kubernetes.
- Complementos: hay cientos de proyectos diseñados para ampliar la funcionalidad y trabajar con Kubernetes que puede agregar a la infraestructura de su clúster o utilizarlos para facilitar la ejecución de sus cargas de trabajo. De modo que no tenga que crear y administrar esos complementos por su cuenta, AWS proporciona the section called “Complementos de Amazon EKS” que se pueden usar con los clústeres. Amazon EKS Anywhere ofrece paquetes seleccionados que incluyen compilaciones de muchos proyectos populares de código abierto. Por lo tanto, no tiene que crear el software ni administrar parches de seguridad, correcciones de errores o actualizaciones críticas. Del mismo modo, si los valores predeterminados se ajustan a sus necesidades, es habitual que se necesite muy poca configuración de esos complementos. Consulte the section called “Extensión de clústeres” para obtener más información sobre cómo ampliar su clúster con complementos.

Kubernetes en acción

En el siguiente diagrama se muestran las actividades clave que llevaría a cabo como administrador de Kubernetes o desarrollador de aplicaciones para crear y usar un clúster de Kubernetes. En el proceso, ilustra cómo los componentes de Kubernetes interactúan entre sí, con la nube de AWS como ejemplo del proveedor de nube subyacente.



Un administrador de Kubernetes crea el clúster de Kubernetes mediante una herramienta específica para el tipo de proveedor en el que se creará el clúster. En este ejemplo, se utiliza la nube de AWS como proveedor, que ofrece el servicio administrado de Kubernetes denominado Amazon EKS. El servicio administrado asigna automáticamente los recursos necesarios para crear el clúster, lo que incluye la creación de dos nuevas nubes privadas virtuales (Amazon VPC) para el clúster, la configuración de la red y la asignación de permisos de Kubernetes directamente en las nuevas VPC para la administración de activos en la nube. Además, el servicio administrado garantiza que los servicios del plano de control tengan dónde ejecutarse y asigna cero o más instancias de Amazon EC2 como nodos de Kubernetes para ejecutar cargas de trabajo. AWS administra una Amazon VPC para el plano de control, mientras que la otra Amazon VPC contiene los nodos del cliente que ejecutan las cargas de trabajo.

En el futuro, muchas de las tareas del administrador de Kubernetes se harán mediante herramientas de Kubernetes como `kubectl`. Esa herramienta envía las solicitudes de servicios directamente al plano de control del clúster. Por lo tanto, las formas en que se hacen las consultas y los cambios en el clúster son muy similares a las formas en que se harían en cualquier clúster de Kubernetes.

Un desarrollador de aplicaciones que desee implementar cargas de trabajo en este clúster puede llevar a cabo varias tareas. El desarrollador debe crear la aplicación en una o más imágenes de

contenedor y, a continuación, enviarlas a un registro de contenedores al que pueda acceder el clúster de Kubernetes. AWS ofrece Amazon Elastic Container Registry (Amazon ECR) para ese fin.

Para ejecutar la aplicación, el desarrollador puede crear archivos de configuración con formato YAML que indiquen al clúster cómo ejecutar la aplicación, incluidos los contenedores que deben extraerse del registro y cómo empaquetarlos en pods. El plano de control (programador) programa los contenedores en uno o más nodos y el tiempo de ejecución del contenedor en cada nodo extrae y ejecuta los contenedores necesarios. El desarrollador también puede configurar un equilibrador de carga de aplicaciones para equilibrar el tráfico hacia los contenedores disponibles que se ejecutan en cada nodo y exponer la aplicación al mundo exterior para que esté disponible en una red pública. Una vez hecho esto, cualquier persona que desee utilizar la aplicación puede conectarse al punto de conexión de la aplicación para acceder a ella.

En las siguientes secciones se abordan los detalles de cada una de estas características desde la perspectiva de los clústeres y las cargas de trabajo de Kubernetes.

Clústeres

Si su trabajo consiste en iniciar y administrar clústeres de Kubernetes, debe saber cómo se crean, mejoran, administran y eliminan los clústeres de Kubernetes. También debe saber cuáles son los componentes que componen un clúster y qué debe hacer para mantenerlos.

Las herramientas para administrar los clústeres gestionan la superposición entre los servicios de Kubernetes y el proveedor de hardware subyacente. Por esa razón, la automatización de estas tareas la suele llevar a cabo el proveedor de Kubernetes (como Amazon EKS o Amazon EKS Anywhere) mediante herramientas específicas del proveedor. Por ejemplo, para iniciar un clúster de Amazon EKS puede utilizar `eksctl create cluster`, mientras que para Amazon EKS Anywhere puede usar `eksctl anywhere create cluster`. Tenga en cuenta que, si bien estos comandos crean un clúster de Kubernetes, son específicos del proveedor y no forman parte del proyecto de Kubernetes en sí.

Herramientas de creación y administración de clústeres

El proyecto de Kubernetes ofrece herramientas para crear un clúster de Kubernetes manualmente. Por lo tanto, si desea instalar Kubernetes en una sola máquina o ejecutar el plano de control en una máquina y agregar nodos manualmente, puede usar herramientas de la CLI, como kind, minikube o kubeadm que se enumeran en Install Tools de Kubernetes. Para simplificar y automatizar todo el ciclo de vida de la creación y administración de clústeres, es mucho más fácil utilizar herramientas

compatibles con un proveedor de Kubernetes establecido, como Amazon EKS o Amazon EKS Anywhere.

En la nube de AWS, puede crear clústeres de Amazon EKS mediante herramientas de la CLI, como eksctl, o herramientas más declarativas, como Terraform (consulte Amazon EKS Blueprints for Terraform). También puede utilizar la Consola de administración de AWS para crear un clúster. Consulte Características de Amazon EKS para obtener una lista de todo lo que obtiene con Amazon EKS. Las responsabilidades de Kubernetes que Amazon EKS asume incluyen:

- Plano de control administrado: AWS garantiza que el clúster de Amazon EKS esté disponible y sea escalable, ya que administra el plano de control y hace que esté disponible en todas las zonas de disponibilidad de AWS.
- Administración de nodos: en lugar de agregar nodos manualmente, puede hacer que Amazon EKS cree nodos automáticamente según sea necesario mediante grupos de nodos administrados (consulte the section called “Grupos de nodos administrados”) o Karpenter. Los grupos de nodos administrados tienen integraciones con el Escalado automático de clústeres de Kubernetes. Con las herramientas de administración de nodos, puede aprovechar los ahorros de costos, como las instancias de spot y la consolidación de nodos, y la disponibilidad, con las funciones de programación para establecer cómo se implementan las cargas de trabajo y cómo se seleccionan los nodos.
- Redes de clústeres: mediante plantillas de CloudFormation, `eksctl` configura las redes entre los componentes del plano de control y del plano de datos (nodo) del clúster de Kubernetes. También configura puntos de conexión a través de los cuales se pueden llevar a cabo las comunicaciones internas y externas. Consulte De-mystifying cluster networking for Amazon EKS worker nodes para obtener más información. Las comunicaciones entre pods en Amazon EKS se llevan a cabo mediante Pod Identities de Amazon EKS (consulte the section called “Pod Identity”), lo que permite a los pods aprovechar los métodos de administración de credenciales y permisos en la nube de AWS.
- Complementos: Amazon EKS le ahorra tener que crear y agregar componentes de software que se utilizan habitualmente para admitir clústeres de Kubernetes. Por ejemplo, cuando crea un clúster de Amazon EKS desde la Consola de administración de AWS, se agregan automáticamente el kube-proxy de Amazon EKS (the section called “kube-proxy”), el complemento CNI de Amazon VPC para Kubernetes (the section called “CNI de Amazon VPC”) y los complementos de CoreDNS (the section called “CoreDNS”). Consulte the section called “Complementos de Amazon EKS” para obtener más información sobre estos complementos, incluida una lista de los que están disponibles.

Para ejecutar sus clústeres en sus propias computadoras y redes en las instalaciones, Amazon ofrece Amazon EKS Anywhere. En lugar de que la nube de AWS sea el proveedor, tiene la opción de ejecutar Amazon EKS Anywhere en plataformas como VMware vSphere, bare metal (proveedor de Tinkerbell), Snow, CloudStack o Nutanix con su propio equipo.

Amazon EKS Anywhere se basa en el mismo software de Amazon EKS Distro que utiliza Amazon EKS. Sin embargo, Amazon EKS Anywhere se basa en diferentes implementaciones de la interfaz Cluster API de Kubernetes (CAPI) para administrar todo el ciclo de vida de las máquinas de un clúster de Amazon EKS Anywhere (como CAPV para vSphere y CAPC para CloudStack). Dado que todo el clúster se ejecuta en un equipo que le pertenece, asume la responsabilidad adicional de administrar el plano de control y hacer copias de seguridad de sus datos (consulte `etcd` más adelante en este documento).

Componentes del clúster

Los componentes del clúster de Kubernetes se dividen en dos áreas principales: el plano de control y los nodos de trabajo. Los componentes del plano de control administran el clúster y proporcionan acceso a sus API. Los nodos de trabajo (a veces denominados simplemente nodos) proporcionan los lugares donde se ejecutan las cargas de trabajo reales. Los componentes de los nodos consisten en servicios que se ejecutan en cada nodo para comunicarse con el plano de control y ejecutar contenedores. El conjunto de nodos de trabajo del clúster se denomina plano de datos.

Plano de control

El plano de control consiste en un conjunto de servicios que administran el clúster. Es posible que todos estos servicios se ejecuten en un solo equipo o que estén repartidos en varios equipos. Internamente, se denominan instancias del plano de control (CPI). La forma en que se ejecutan las CPI depende del tamaño del clúster y de los requisitos de alta disponibilidad. A medida que aumenta la demanda en el clúster, un servicio de plano de control puede escalarse para ofrecer más instancias de ese servicio y equilibrar la carga de las solicitudes entre las instancias.

Entre las tareas que llevan a cabo los componentes del plano de control de Kubernetes se incluyen las siguientes:

- Comunicación con los componentes del clúster (servidor de API): el servidor de API (kube-apiserver) expone la API de Kubernetes para que las solicitudes al clúster se puedan efectuar tanto desde dentro como desde fuera del clúster. En otras palabras, las solicitudes para agregar o cambiar los objetos de un clúster (pods, servicios, nodos, etc.) pueden provenir de comandos externos, como las solicitudes de `kubectl` para ejecutar un pod. Del mismo modo, se pueden

hacer solicitudes desde el servidor de API a los componentes del clúster, por ejemplo, una consulta al servicio `kubelet` para conocer el estado de un pod.

- Almacenamiento de datos sobre el clúster (almacén de pares de clave y valor de **`etcd`**): el servicio de `etcd` desempeña la función fundamental de hacer un seguimiento del estado actual del clúster. Si el servicio `etcd` dejara de ser accesible, no podría actualizar ni consultar el estado del clúster, aunque las cargas de trabajo seguirían ejecutándose durante un tiempo. Por ese motivo, los clústeres críticos suelen tener varias instancias del servicio `etcd` con equilibrio de carga que se ejecutan a la vez y hacen copias de seguridad periódicas del almacén de pares de clave-valor de `etcd` en caso de pérdida o corrupción de los datos. Tenga en cuenta que, en Amazon EKS, todo esto se gestiona automáticamente de forma predeterminada. Amazon EKS Anywhere proporciona instrucciones para la copia de seguridad y restauración de etcd. Consulte el Modelo de datos de etcd para aprender cómo `etcd` administra los datos.
- Programación de los pods por nodos (Programador): las solicitudes para iniciar o detener un pod en Kubernetes se dirigen al Programador de Kubernetes (kube-scheduler). Como un clúster puede tener varios nodos capaces de ejecutar el pod, es el programador quien decide en qué nodo (o nodos, en el caso de las réplicas) debe ejecutarse el pod. Si no hay suficiente capacidad disponible para ejecutar el pod solicitado en un nodo existente, la solicitud fallará, a menos que haya tomado otras medidas. Estas disposiciones podrían incluir la habilitación de servicios, como los grupos de nodos administrados (the section called “Grupos de nodos administrados”) o Karpenter, que puedan iniciar automáticamente nuevos nodos para gestionar las cargas de trabajo.
- Mantenimiento de los componentes en el estado deseado (Administrador de controladores): el Administrador de controladores de Kubernetes se ejecuta como un proceso daemon (kube-controller-manager) para observar el estado del clúster y hacer cambios en él para restablecer los estados esperados. En concreto, hay varios controladores que vigilan diferentes objetos de Kubernetes, entre los que se incluyen `statefulset-controller`, `endpoint-controller`, `cronjob-controller` y `node-controller`, entre otros.
- Administración de los recursos de la nube (Administrador de controladores de la nube): el Administrador de controladores de la nube (cloud-controller-manager) gestiona las interacciones entre Kubernetes y el proveedor de servicios en la nube que hace las solicitudes de los recursos del centro de datos subyacente. Los controladores administrados por el administrador de controladores de la nube pueden incluir un controlador de rutas (para establecer rutas de red en la nube), un controlador de servicios (para utilizar servicios de equilibrador de carga en la nube) y un controlador del ciclo de vida de nodos (para mantener los nodos sincronizados con Kubernetes a lo largo de sus ciclos de vida).

Nodos de trabajo (plano de datos)

En el caso de un clúster de Kubernetes de un solo nodo, las cargas de trabajo se ejecutan en la misma máquina que el plano de control. Sin embargo, una configuración más estándar consiste en tener uno o más sistemas de computación independientes (nodos) dedicados a ejecutar cargas de trabajo de Kubernetes.

Al crear un clúster de Kubernetes por primera vez, algunas herramientas de creación de clústeres permiten configurar un número determinado de nodos para agregarlos al clúster (ya sea al identificar sistemas de computación existentes o hacer que el proveedor cree otros nuevos). Antes de agregar cargas de trabajo a esos sistemas, se agregan servicios a cada nodo para implementar estas características:

- Administración de cada nodo (**`kubelet`**): el servidor de API se comunica con el servicio kubelet que se ejecuta en cada nodo para asegurarse de que el nodo esté registrado correctamente y de que los pods solicitados por el Programador estén en ejecución. El kubelet puede leer los manifiestos de los pods y configurar los volúmenes de almacenamiento u otras características que necesiten los pods del sistema local. También puede comprobar el estado de los contenedores que se ejecutan localmente.
- Ejecución de contenedores en un nodo (tiempo de ejecución de contenedores): el tiempo de ejecución de contenedores de cada nodo administra los contenedores solicitados para cada pod asignado al nodo. Esto significa que puede extraer imágenes de contenedores del registro correspondiente, ejecutar el contenedor, detenerlo y responder a las consultas sobre el contenedor. El tiempo de ejecución del contenedor predeterminado es containerd. A partir de la versión 1.24 de Kubernetes, se eliminó la integración especial de Docker (`dockershim`), que podía usarse como tiempo de ejecución del contenedor de Kubernetes. Si bien puede seguir usando Docker para probar y ejecutar contenedores en su sistema local, para usar Docker con Kubernetes tendrá que instalar el motor de Docker en cada nodo para usarlo con Kubernetes.
- Administración de la red entre contenedores (**`kube-proxy`**): para permitir la comunicación entre pods, Kubernetes utiliza una característica denominada servicio para configurar redes de pods que rastrean las direcciones IP y los puertos asociados a dichos pods. El servicio kube-proxy se ejecuta en todos los nodos para permitir que se produzca la comunicación entre los pods.

Extensión de clústeres

Hay algunos servicios que puede agregar a Kubernetes para que sean compatibles con el clúster, pero no se ejecutan en el plano de control. Estos servicios suelen ejecutarse directamente en los

nodos del espacio de nombres kube-system o en su propio espacio de nombres (como suele ocurrir con los proveedores de servicios de terceros). Un ejemplo común es el servicio CoreDNS, que proporciona servicios DNS al clúster. Consulte Discovering built in services para obtener información sobre cómo ver qué servicios de clúster se están ejecutando en kube-system en su clúster.

Hay diferentes tipos de complementos que puede considerar agregar a sus clústeres. Para mantener los clústeres en buen estado, puede agregar características de observabilidad (consulte Supervisión de clústeres) capaces de efectuar tareas como el registro, la auditoría y las métricas. Con esta información, puede solucionar los problemas que se producen, a menudo a través de las mismas interfaces de observabilidad. Algunos ejemplos de estos tipos de servicios son Amazon GuardDuty, CloudWatch (consulte the section called “Amazon CloudWatch”), AWS Distro para OpenTelemetry, el complemento CNI de Amazon VPC para Kubernetes (consulte the section called “CNI de Amazon VPC”), y Supervisión de Kubernetes de Grafana. En cuanto al almacenamiento (consulte Almacenamiento de datos de aplicaciones), los complementos de Amazon EKS incluyen el controlador CSI de Amazon Elastic Block Store (consulte the section called “Amazon EBS”), el controlador CSI de Amazon Elastic File System (consulte the section called “Amazon EFS”) y varios complementos de almacenamiento de terceros, como el controlador CSI de Amazon FSx para NetApp ONTAP the section called “Amazon FSx para NetApp ONTAP”.

Para obtener una lista más completa de los complementos disponibles de Amazon EKS, consulte the section called “Complementos de Amazon EKS”.

Cargas de trabajo

Kubernetes define una carga de trabajo como “una aplicación que se ejecuta en Kubernetes”. Esa aplicación puede consistir en un conjunto de microservicios que se ejecutan como contenedores en pods, o puede ejecutarse como un trabajo por lotes u otro tipo de aplicaciones. El trabajo de Kubernetes consiste en asegurarse de que las solicitudes que haga para configurar o implementar esos objetos se lleven a cabo. Si se dedica a implementar aplicaciones, debería aprender cómo se crean los contenedores, cómo se definen los pods y qué métodos puede usar para implementarlos.

Contenedores

El elemento más básico de la carga de trabajo de una aplicación que se implementa y administra en Kubernetes es un pod. Un pod representa una forma de contener los componentes de una aplicación, además de definir las especificaciones que describen los atributos del pod. Compárelo con algo parecido a un paquete RPM o Deb, que agrupa software para un sistema Linux, pero no se ejecuta en sí mismo como una entidad.

Como el pod es la unidad implementable más pequeña, normalmente contiene un contenedor. Sin embargo, en el caso de que los contenedores estén bien acoplados, puede haber varios contenedores en un mismo pod. Por ejemplo, un contenedor de servidor web puede estar empaquetado en un pod con un contenedor tipo sidecar que puede proporcionar servicios de registro, supervisión u otro servicio que esté estrechamente relacionado con el contenedor del servidor web. En este caso, al estar en el mismo pod, se garantiza que, para cada instancia del pod en ejecución, ambos contenedores se ejecuten siempre en el mismo nodo. Del mismo modo, todos los contenedores de un pod comparten el mismo entorno, y los contenedores de un pod se ejecutan como si estuvieran en el mismo host aislado. El efecto de esto es que los contenedores comparten una única dirección IP que proporciona acceso al pod y los contenedores pueden comunicarse entre sí como si se ejecutaran en su propio host local.

Las especificaciones del pod (PodSpec) definen el estado deseado del pod. Puede implementar un pod individual o varios pods a través de los recursos de carga de trabajo para administrar las plantillas de pod. Los recursos de carga de trabajo incluyen implementaciones (para administrar múltiples réplicas de pods), StatefulSets (para implementar pods que deben ser únicos, como los pods de bases de datos) y DaemonSets (donde un pod debe ejecutarse de forma continua en todos los nodos). Puede obtener más información sobre estos temas más adelante.

Mientras que un pod es la unidad más pequeña que se implemente, un contenedor es la unidad más pequeña que se crea y administra.

Creación de contenedores

En realidad, el pod no es más que una estructura alrededor de uno o más contenedores, en la que cada contenedor contiene el sistema de archivos, los ejecutables, los archivos de configuración, las bibliotecas y otros componentes necesarios para ejecutar realmente la aplicación. Debido a que una empresa llamada Docker Inc. fue la primera en popularizar los contenedores, algunas personas se refieren a los contenedores como contenedores de Docker. Sin embargo, desde entonces, la Open Container Initiative ha definido los tiempos de ejecución, las imágenes y los métodos de distribución de los contenedores para la industria. Si a esto le sumamos el hecho de que los contenedores se crearon a partir de muchas características de Linux existentes, otros suelen denominar contenedores OCI, contenedores Linux o simplemente contenedores.

Cuando crea un contenedor, normalmente comienza con un Dockerfile (literalmente llamado así). Dentro de ese Dockerfile, puede identificar lo siguiente:

- Una imagen base: una imagen de contenedor base es un contenedor que normalmente se crea a partir de una versión mínima del sistema de archivos de un sistema operativo (como Red Hat

Enterprise Linux o Ubuntu) o de un sistema mínimo que se mejora para proporcionar software que ejecute tipos específicos de aplicaciones (como aplicaciones nodejs o python).

- Software de aplicaciones: puede agregar el software de su aplicación a su contenedor de la misma manera que lo agregaría a un sistema Linux. Por ejemplo, en su Dockerfile puede ejecutar `npm` y `yarn` para instalar una aplicación Java o `yum` y `dnf` para instalar paquetes RPM. En otras palabras, si utiliza el comando RUN en un Dockerfile, puede ejecutar cualquier comando que esté disponible en el sistema de archivos de la imagen base para instalar software o configurar software dentro de la imagen contenedora resultante.
- Instrucciones: en la referencia de Dockerfile se describen las instrucciones que puede agregar a un Dockerfile al configurarlo. Estas incluyen instrucciones que se utilizan para crear lo que hay en el propio contenedor (archivos `ADD` o `COPY` del sistema local), identificar los comandos que se van a ejecutar cuando se ejecuta el contenedor (`CMD` o `ENTRYPOINT`) y conectar el contenedor al sistema en el que se ejecuta (identificando el `USER` que se va a ejecutar, el `VOLUME` local que se va a montar o los puertos a `EXPOSE`).

Si bien el comando `docker` y el servicio se han utilizado tradicionalmente para crear contenedores (`docker build`), otras herramientas disponibles para crear imágenes de contenedores incluyen podman y nerdctl. Consulte Creación de imágenes de contenedores de mejor calidad o Descripción general de Docker Build para obtener información sobre cómo crear contenedores.

Almacenamiento de contenedores

Una vez que haya creado la imagen del contenedor, puede almacenarla en un registro de distribución de contenedores de su estación de trabajo o en un registro de contenedores público. La ejecución de un registro de contenedores privado en su estación de trabajo le permite almacenar las imágenes de los contenedores de forma local y ponerlas a su disposición de forma inmediata.

Para almacenar las imágenes de los contenedores de una forma más pública, puede enviarlas a un registro de contenedores público. Los registros públicos de contenedores proporcionan una ubicación central para almacenar y distribuir las imágenes de los contenedores. Algunos ejemplos de registros de contenedores públicos son Amazon Elastic Container Registry, el registro Red Hat Quay y el registro Docker Hub.

Al ejecutar cargas de trabajo en contenedores en Amazon Elastic Kubernetes Service (Amazon EKS), recomendamos que extraiga copias de las imágenes oficiales de Docker que se almacenan en Amazon Elastic Container Registry. Amazon ECR ha estado almacenando estas imágenes desde 2021. Puede buscar imágenes de contenedores populares en la Galería pública de Amazon ECR

y, específicamente, para las imágenes de Docker Hub, puede buscar en la Galería de Docker de Amazon ECR.

Ejecución de contenedores

Como los contenedores se crean en un formato estándar, un contenedor puede ejecutarse en cualquier máquina que pueda ejecutar un contenedor en tiempo de ejecución (por ejemplo, Docker) y cuyo contenido coincida con la arquitectura de la máquina local (por ejemplo, `x86_64` o `arm`). Para probar un contenedor o simplemente ejecutarlo en el escritorio local, puede usar los comandos `docker run` o `podman run` para iniciar un contenedor en el servidor local. Sin embargo, en Kubernetes, cada nodo de trabajo tiene implementado un tiempo de ejecución del contenedor y depende de Kubernetes que se solicite que un nodo ejecute un contenedor.

Una vez que se ha asignado un contenedor para que se ejecute en un nodo, este comprueba si la versión solicitada de la imagen del contenedor ya existe en el nodo. Si no es así, Kubernetes indica al tiempo de ejecución del contenedor que extraiga ese contenedor del registro de contenedores correspondiente y, a continuación, lo ejecute localmente. Tenga en cuenta que la imagen de un contenedor hace referencia al paquete de software que se mueve entre su portátil, el registro del contenedor y los nodos de Kubernetes. Un contenedor hace referencia a una instancia en ejecución de esa imagen.

Pods

Una vez que los contenedores estén listos, trabajar con los pods incluye configurar, implementar y hacer que estos sean accesibles.

Configuración de pods

Cuando define un pod, le asigna un conjunto de atributos. Esos atributos deben incluir al menos el nombre del pod y la imagen del contenedor para que se ejecuten. Sin embargo, hay muchas otras cosas que también querrá configurar con las definiciones de su pod (consulte la página PodSpec para obtener más información sobre lo que puede incluir un pod). Entre ellos se incluyen:

- Almacenamiento: cuando un contenedor en ejecución se detiene y se elimina, el almacenamiento de datos en ese contenedor desaparecerá, a menos que configure un almacenamiento más permanente. Kubernetes admite muchos tipos de almacenamiento diferentes y los abstrae bajo el nombre de volúmenes. Los tipos de almacenamiento incluyen CephFS, NFS, iSCSI, etc. Incluso puede usar un dispositivo de bloques local desde la computadora local. Con uno de esos tipos de almacenamiento disponibles en su clúster, puede montar el volumen de almacenamiento

en un punto de montaje seleccionado del sistema de archivos del contenedor. Un volumen persistente es aquel que sigue existiendo después de eliminar el pod, mientras que un volumen efímero se elimina cuando se elimina el pod. Si el administrador del clúster creó diferentes clases de almacenamiento para el clúster, es posible que tenga la opción de elegir los atributos del almacenamiento que va a utilizar, por ejemplo, si el volumen se elimina o se recupera después de su uso, si se ampliará si se necesita más espacio e incluso si cumple con ciertos requisitos de rendimiento.

- Secretos: al poner los secretos a disposición de los contenedores en las especificaciones del pod, puede proporcionar los permisos que esos contenedores necesitan para acceder a los sistemas de archivos, las bases de datos u otros activos protegidos. Las claves, las contraseñas y los tokens son algunos de los elementos que se pueden almacenar como secretos. El uso de secretos elimina la necesidad de almacenar esta información en las imágenes de contenedores, ya que únicamente es necesario que los secretos estén disponibles para los contenedores en ejecución. Los ConfigMaps son similares a los secretos. `ConfigMap` tiende a contener información menos crítica, como pares de clave-valor para configurar un servicio.
- Recursos de contenedores: los objetos para seguir configurando los contenedores pueden adoptar la forma de configuración de recursos. Para cada contenedor, puede solicitar la cantidad de memoria y CPU que puede usar, así como establecer límites a la cantidad total de esos recursos que puede usar el contenedor. Consulte Resource Management for Pods and Containers para ver ejemplos.
- Interrupciones: los pods se pueden interrumpir de forma involuntaria (un nodo deja de funcionar) o de forma voluntaria (se desea una actualización). Al configurar un presupuesto de interrupciones para los pods, puede controlar en cierta medida la disponibilidad de su aplicación en caso de que se produzcan interrupciones. Para ver ejemplos, consulte Specifying a Disruption Budget for your application.
- Espacios de nombres: Kubernetes proporciona diferentes formas de aislar las cargas de trabajo y los componentes de Kubernetes entre sí. Ejecutar todos los pods de una aplicación concreta en el mismo espacio de nombres es una forma habitual de proteger y administrar esos pods juntos. Puede crear sus propios espacios de nombres para usarlos o elegir no indicar ninguno (lo que hace que Kubernetes utilice el espacio de nombres `default`). Los componentes del plano de control de Kubernetes normalmente se ejecutan en el espacio de nombres kube-system.

La configuración que acabamos de describir normalmente se recopila en un archivo YAML para aplicarla al clúster de Kubernetes. En el caso de los clústeres de Kubernetes personales, puede almacenar estos archivos YAML en el sistema local. Sin embargo, dado que los clústeres

y las cargas de trabajo son más importantes, GitOps es una forma popular de automatizar el almacenamiento y las actualizaciones de los recursos de carga de trabajo e infraestructura de Kubernetes.

Los objetos que se utilizan para recopilar e implementar la información del pod se definen mediante uno de los siguientes métodos de implementación.

Implementación de pods

El método que elija para implementar los pods depende del tipo de aplicación que planee ejecutar con ellos. Aquí tiene algunas opciones:

- Aplicaciones sin estado: una aplicación sin estado no guarda los datos de la sesión de un cliente, por lo que no es necesario volver a hacer referencia a lo que ocurrió en una sesión anterior. Esto facilita la sustitución de los pods por otros nuevos en caso de que no funcionen correctamente, o bien el traslado de un sitio a otro sin guardar su estado. Si ejecuta una aplicación sin estado (como un servidor web), puede usar una Implementación para implementar Pods y Replicasets. Un ReplicaSet define cuántas instancias de un pod desea que se ejecuten simultáneamente. Aunque puede ejecutar un ReplicaSet directamente, es habitual ejecutar réplicas directamente dentro de una implementación, para definir cuántas réplicas de un pod deben ejecutarse a la vez.
- Aplicaciones con estado: una aplicación con estado es aquella en la que la identidad del pod y el orden en que se lanzan los pods son importantes. Estas aplicaciones necesitan un almacenamiento persistente que sea estable y deben implementarse y escalarse de manera coherente. Para implementar una aplicación con estado en Kubernetes, puede usar StatefulSets. Un ejemplo de una aplicación que normalmente se ejecuta como StatefulSet es una base de datos. Dentro de un StatefulSet, puede definir las réplicas, el pod y sus contenedores, los volúmenes de almacenamiento que se van a montar y las ubicaciones del contenedor donde se almacenan los datos. Consulte Run a Replicated Stateful Application para ver un ejemplo de una base de datos que se implementa como ReplicaSet.
- Aplicaciones por nodo: hay ocasiones en las que desea ejecutar una aplicación en cada nodo del clúster de Kubernetes. Por ejemplo, su centro de datos puede requerir que todos los equipos ejecuten una aplicación de monitoreo o un servicio de acceso remoto concreto. Para Kubernetes, puede utilizar un DaemonSet para garantizar que la aplicación seleccionada se ejecute en todos los nodos del clúster.
- Las aplicaciones se ejecutan hasta completarse: hay algunas aplicaciones que desea ejecutar para completar una tarea concreta. Esto podría incluir uno que publique informes de estado mensuales o que elimine datos antiguos. Se puede usar un objeto Job para configurar una

aplicación para que se inicie y ejecute y, a continuación, se cierre cuando finalice la tarea. Un objeto CronJob permite configurar una aplicación para que se ejecute a una hora, minuto, día del mes, mes o día de la semana específicos, mediante una estructura definida por el formato crontab de Linux.

Hacer que las aplicaciones sean accesibles desde la red

Dado que las aplicaciones solían implementarse como un conjunto de microservicios que se desplazaban a diferentes lugares, Kubernetes necesitaba una forma de que esos microservicios pudieran encontrarse entre sí. Además, para que otras personas pudieran acceder a una aplicación fuera del clúster de Kubernetes, este necesitaba una forma de exponer esa aplicación en direcciones y puertos externos. Estas características relacionadas con las redes se llevan a cabo con los objetos Servicio y Entrada, respectivamente:

- Servicios: dado que un pod puede moverse a diferentes nodos y direcciones, otro pod que necesite comunicarse con el primer pod podría tener dificultades para localizar dónde está. Para resolver este problema, Kubernetes le permite representar una aplicación como un servicio. Con un servicio, puede identificar un pod o un conjunto de pods con un nombre concreto y, a continuación, indicar qué puerto expone el servicio de esa aplicación desde el pod y qué puertos podría utilizar otra aplicación para contactar con ese servicio. Otro pod de un clúster puede simplemente solicitar un servicio por su nombre y Kubernetes dirigirá esa solicitud al puerto adecuado de una instancia del pod que ejecute ese servicio.
- Entrada: una entrada es lo que permite que las aplicaciones representadas por los servicios de Kubernetes estén disponibles para los clientes que se encuentran fuera del clúster. Las características básicas de entrada incluyen un equilibrador de carga (administrado por la entrada), el controlador de entrada y reglas para dirigir las solicitudes desde el controlador al servicio. Hay varios controladores de entrada que puede elegir con Kubernetes.

Pasos a seguir a continuación

Comprender los conceptos básicos de Kubernetes y su relación con Amazon EKS lo ayudará a navegar por la documentación de Amazon EKS y la documentación de Kubernetes para encontrar la información que necesita a fin de administrar los clústeres de Amazon EKS e implementar cargas de trabajo en esos clústeres. Para comenzar a usar Amazon EKS, elija una de las siguientes opciones:

- the section called “Creación de un clúster (eksctl)”
- the section called “Creación de un clúster ”

- the section called “Implementación de muestra (Linux)”
- Administración de clústeres

Implementación de clústeres de Amazon EKS en entornos en las instalaciones y en la nube

Comprensión de las opciones de implementación de Amazon EKS

Amazon Elastic Kubernetes Service (Amazon EKS) es un servicio de Kubernetes completamente administrado que permite ejecutar Kubernetes sin problemas en la nube y en los entornos en las instalaciones.

En la nube, Amazon EKS automatiza la administración de la infraestructura de clústeres de Kubernetes para el plano de control y los nodos de Kubernetes. Esto resulta esencial a la hora de programar contenedores, administrar la disponibilidad de las aplicaciones, escalar recursos dinámicamente, optimizar la computación, almacenar datos de clústeres y llevar a cabo otras funciones críticas. Con Amazon EKS, obtiene el rendimiento, la escalabilidad, la fiabilidad y la disponibilidad sólidos propios de la infraestructura de AWS, junto con integraciones nativas con servicios de AWS de redes, seguridad, almacenamiento y observabilidad.

Para simplificar la ejecución de Kubernetes en los entornos en las instalaciones, puede utilizar los mismos clústeres, características y herramientas de Amazon EKS para the section called “Nodos” o Nodos híbridos de Amazon EKS en una infraestructura propia, o bien puede utilizar Amazon EKS Anywhere para entornos autónomos y aislados.

Amazon EKS en la nube

Puede utilizar Amazon EKS con computación en regiones de AWS, Zonas locales de AWS y Zonas de AWS Wavelength. Con Amazon EKS en la nube, AWS administra plenamente la seguridad, escalabilidad y disponibilidad del plano de control de Kubernetes en la región AWS. Cuando ejecuta aplicaciones con computación en regiones de AWS, se beneficia de todas las características que ofrecen AWS y Amazon EKS, incluido el modo automático de Amazon EKS, que automatiza completamente la administración de la infraestructura de clústeres de Kubernetes para computación, almacenamiento y redes en AWS con un solo clic. Al ejecutar aplicaciones con computación en Zonas locales de AWS y Zonas de AWS Wavelength, puede utilizar nodos autoadministrados de Amazon EKS para conectar instancias de Amazon EC2 para la computación en clústeres y puede utilizar los demás servicios de AWS disponibles en Zonas locales de AWS y Zonas de AWS

Wavelength. Para obtener más información, consulte las características de las Zonas locales de AWS y las características de las Zonas de AWS Wavelength.

	Amazon EKS en las regiones de AWS	Amazon EKS en Zonas locales o de Wavelength
Administración del plano de control de Kuberenetes	Administradas por AWS	Administradas por AWS
Ubicación del plano de control de Kubernetes	Regiones de AWS	Regiones de AWS
Plano de datos de Kubernetes	• Modo automático de Amazon EKS • Grupos de nodos administrados de Amazon EKS • Nodos autoadministrados de Amazon EC2 • AWS Fargate	• Grupos de nodos administrados de Amazon EKS (solo Zonas locales) • Nodos autoadministrados de Amazon EC2
Ubicación del plano de datos de Kubernetes	Regiones de AWS	Zonas locales o de Wavelength de AWS

Amazon EKS en el centro de datos o en entornos periféricos

Si necesita ejecutar aplicaciones en centros de datos propios o en entornos periféricos, puede utilizar Amazon EKS en AWS Outposts o los Nodos híbridos de Amazon EKS. Puede utilizar nodos autoadministrados con instancias de Amazon EC2 en AWS Outposts para la computación en clúster, o bien puede utilizar Nodos híbridos de Amazon EKS con una infraestructura propia en las instalaciones o periférica para la computación en clúster. AWS Outposts es una infraestructura administrada por AWS que se ejecuta en los centros de datos o en instalaciones de coubicación, mientras que los Nodos híbridos de Amazon EKS se ejecutan en máquinas físicas o virtuales que se administran en entornos en las instalaciones o periféricos. Amazon EKS en AWS Outposts y los Nodos híbridos de Amazon EKS requieren una conexión fiable desde los entornos en las instalaciones a una región de AWS, y puede utilizar los mismos clústeres, características y herramientas de Amazon EKS que utiliza para ejecutar aplicaciones en la nube. Cuando se ejecuta

en AWS Outposts, puede implementar alternativamente todo el clúster de Kubernetes en AWS Outposts con clústeres locales de Amazon EKS en AWS Outposts.

	Nodos híbridos de Amazon EKS	Amazon EKS en AWS Outposts
Administración del plano de control de Kuberenetes	Administradas por AWS	Administradas por AWS
Ubicación del plano de control de Kubernetes	Regiones de AWS	Regiones de AWS o AWS Outposts
Plano de datos de Kubernetes	Máquinas físicas o virtuales administradas por el cliente	Nodos autoadministrados de Amazon EC2
Ubicación del plano de datos de Kubernetes	Centro de datos del cliente o entorno periférico	Centro de datos del cliente o entorno periférico

Amazon EKS Anywhere para entornos aislados

Amazon EKS Anywhere simplifica la administración de clústeres de Kubernetes mediante la automatización de aquellas tareas arduas que no marcan la diferencia, como la configuración de la infraestructura y las operaciones del ciclo de vida de los clústeres de Kubernetes en entornos en las instalaciones y periféricos. A diferencia de Amazon EKS, Amazon EKS Anywhere es un producto administrado por el cliente y los clientes son responsables de las operaciones del ciclo de vida del clúster y del mantenimiento de los clústeres de Amazon EKS Anywhere. Amazon EKS Anywhere se basa en la API de clústeres del subproyecto Kubernetes (CAPI) y es compatible con diversas infraestructuras, como VMware vSphere, bare metal, Nutanix, Apache CloudStack y AWS Snow. Amazon EKS Anywhere se puede ejecutar en entornos aislados y ofrece integraciones opcionales con servicios regionales de AWS para la observabilidad y la administración de identidades.
Para recibir soporte para Amazon EKS Anywhere y acceso a complementos de Kubernetes proporcionados por AWS, puede adquirir Suscripciones empresariales de Amazon EKS Anywhere.

	Amazon EKS Anywhere
Administración del plano de control de Kuberenetes	Administrados por el cliente

	Amazon EKS Anywhere
Ubicación del plano de control de Kubernetes	Centro de datos del cliente o entorno periférico
Plano de datos de Kubernetes	Máquinas físicas o virtuales administradas por el cliente
Ubicación del plano de datos de Kubernetes	Centro de datos del cliente o entorno periférico

Herramientas de Amazon EKS

Puede utilizar el conector de Amazon EKS para registrar y conectar cualquier clúster de Kubernetes que cumpla los requisitos a AWS y verlo en la consola de Amazon EKS. Una vez conectado, puede ver el estado, la configuración y las cargas de trabajo del clúster en la consola de Amazon EKS. Puede utilizar esta característica para ver los clústeres conectados en la consola de Amazon EKS, pero el conector de Amazon EKS no permite realizar operaciones de administración ni de modificación en los clústeres conectados a través de la consola de Amazon EKS.

Amazon EKS Distro es la distribución de AWS de los componentes subyacentes de Kubernetes que impulsan todas las ofertas de Amazon EKS. Incluye los componentes principales que necesita un clúster de Kubernetes para funcionar, como los componentes del plano de control de Kubernetes (etcd, kube-apiserver, kube-scheduler, kube-controller-manager) y los componentes de red (CoreDNS, kube-proxy, complementos de CNI). Amazon EKS Distro se puede utilizar para autoadministrar clústeres de Kubernetes con las herramientas que elija. Las implementaciones de Amazon EKS Distro no están cubiertas por los planes de soporte de AWS.

Configuración para usar Amazon EKS

Para comenzar a administrar sus clústeres de Amazon EKS, debe instalar varias herramientas. Utilice lo siguiente para configurar las credenciales, crear y modificar clústeres y trabajar con ellos una vez que estén en ejecución:

- Configurar AWS CLI: consiga que AWS CLI configure y administre los servicios que necesita para trabajar con los clústeres de Amazon EKS. En concreto, necesita que AWS CLI configure las credenciales, pero también las necesita con otros servicios de AWS.
- Configurar kubectl y eksctl: La CLI de `eksctl` interactúa con AWS para crear, modificar y eliminar clústeres de Amazon EKS. Una vez que el clúster esté activo, utilice el comando de código abierto `kubectl` para administrar los objetos de Kubernetes dentro de los clústeres de Amazon EKS.
- Configuración de un entorno de desarrollo (opcional): considere la posibilidad de agregar las siguientes herramientas:
  - Herramienta de implementación local: si es la primera vez que usa Kubernetes, considere la posibilidad de instalar una herramienta de implementación local como minikube o kind. Estas herramientas le permiten tener un clúster de Amazon EKS en su máquina local para probar aplicaciones.
  - Administrador de paquetes: Helm es un administrador de paquetes de Kubernetes muy popular que simplifica la instalación y administración de paquetes complejos. Con Helm, es más fácil instalar y administrar paquetes como el controlador del equilibrador de carga de AWS en su clúster de Amazon EKS.

Pasos a seguir a continuación

- Configurar AWS CLI
- Configurar kubectl y eksctl
- Inicio rápido: implementación de una aplicación web y almacenamiento de datos

Configurar AWS CLI

La AWS CLI es una herramienta de línea de comandos para trabajar con los servicios de AWS, incluido Amazon EKS. También se usa para autenticar a los roles o usuarios de IAM para acceder al clúster de Amazon EKS y a otros recursos de AWS desde su máquina local. Para aprovisionar

recursos en AWS desde la línea de comandos, debe obtener un ID de clave de acceso de AWS y una clave secreta para utilizarlos en la línea de comandos. A continuación, debe configurar estas credenciales en la AWS CLI. Si aún no ha instalado la AWS CLI, consulte Instalar o actualizar la última versión de la AWS CLI en la Guía del usuario de la interfaz de la línea de comandos de AWS.

Para crear una clave de acceso

1. Inicie sesión en el Consola de administración de AWS.
2. Para cuentas de un solo usuario o de varios usuarios:
   - Cuentas de un solo usuario: en la esquina superior derecha, elija el nombre de usuario de AWS para abrir el menú de navegación. Por ejemplo, elija **`webadmin`** .
   - Cuenta de varios usuarios: elija IAM en la lista de servicios. En el panel de IAM, seleccione Usuarios y elija el nombre del usuario.
3. Elija Credenciales de seguridad.
4. En Clave de acceso, elija Crear clave de acceso.
5. Elija Interfaz de la línea de comandos (CLI) y, a continuación, haga clic en Siguiente.
6. Elija Create access key (Crear clave de acceso).
7. Elija Descargar archivo .csv.

Configuración de la CLI de AWS

Después de instalar la AWS CLI, siga estos pasos para configurarla: Para obtener más información, consulte Configurar la AWS CLI, en la Guía del usuario de la interfaz de la línea de comandos de AWS.

1. En una ventana de terminal, ingrese el siguiente comando:

   ```
   aws configure
   ```

   Si lo desea, puede configurar un perfil con nombre, como `--profile cluster-admin`. Si configura un perfil con nombre en la AWS CLI, deberá transferir siempre este indicador en los siguientes comandos.
2. Introduzca las credenciales de AWS. Por ejemplo:

   ```
   Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
   ```

```
Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: region-code
Default output format [None]: json
```

Cómo obtener un token de seguridad

Si es necesario, ejecute el siguiente comando para obtener un token de seguridad nuevo para la AWS CLI. Para obtener más información, consulte get-session-token en la Referencia de comandos de la AWS CLI.

De forma predeterminada, el token será válido durante 15 minutos. Para cambiar el tiempo de espera predeterminado de la sesión, transfiera el indicador `--duration-seconds`. Por ejemplo:

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

Este comando devuelve las credenciales de seguridad temporales de una sesión de la AWS CLI. Debería ver la siguiente respuesta de salida:

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

Cómo verificar la identidad del usuario

Si es necesario, ejecute el siguiente comando para verificar las credenciales de AWS para su identidad de usuario de IAM (por ejemplo, *`ClusterAdmin`*) para la sesión de terminal.

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

Este comando devuelve el Nombre de recurso de Amazon (ARN) de la entidad de IAM que está configurada para la AWS CLI. Debería ver la siguiente respuesta de ejemplo de salida:

```
{
```

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

Pasos a seguir a continuación

- Configure kubectl y eksctl
- Inicio rápido: implementación de una aplicación web y almacenamiento de datos

Configuración de **kubectl** y **eksctl**

Después de instalar AWS CLI, conviene instalar otras dos herramientas para crear y administrar los clústeres de Kubernetes:

- `kubectl`: la herramienta de la línea de comandos `kubectl` es la principal que utilizará para administrar los recursos del clúster de Kubernetes. Esta página describe cómo descargar y configurar el binario `kubectl` que coincide con la versión del clúster de Kubernetes. Consulte Instalación o actualización de kubectl.
- `eksctl`: la herramienta de la línea de comandos `eksctl` está diseñada para crear clústeres de EKS en la nube de AWS o en las instalaciones (con EKS Anywhere), así como para modificar y eliminar esos clústeres. Consulte Instalación de eksctl.

Instalación o actualización de **kubectl**

Este tema le ayuda a descargar e instalar o actualizar, el `kubectl` binario en su dispositivo. El dato binario es idéntico a las versiones de comunidad ascendente. El dato binario no es exclusivo de Amazon EKS o de AWS. Siga los pasos que se indican a continuación para obtener la versión específica de `kubectl`que necesita, aunque muchos creadores simplemente ejecutan `brew install kubectl` para instalarla.

> **Note**
>
> Debe utilizar una versión de `kubectl` con una diferencia de versión de menos de un número que el plano del control del clúster de Amazon EKS. Por ejemplo, un cliente de `kubectl` `1.32` trabaja con los clústeres `1.31`, `1.32` y `1.33` de Kubernetes.

Paso 1: compruebe si **kubectl** está instalado

Determine si ya tiene `kubectl` instalado en su dispositivo.

```
kubectl version --client
```

Si tiene `kubectl` instalado en la ruta de su dispositivo, el resultado de ejemplo incluye información similar a la siguiente. Si desea actualizar la versión que tiene instalada actualmente con una versión posterior, complete el siguiente paso y asegúrese de instalar la nueva versión en la misma ubicación en la que se encuentra la versión actual.

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

Si no recibe un resultado, entonces no tiene `kubectl` instalado o no está instalado en una ubicación que esté en la ruta de acceso del dispositivo.

Paso 2: instalación o actualización de **kubectl**

Instalación o actualización de `kubectl` en uno de los siguientes sistemas operativos:

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

> **Note**
>
> Si las descargas desde las regiones de AWS mencionadas en esta sección son lentas en su región de AWS, considere la posibilidad de configurar CloudFront para distribuir el contenido. Para obtener más información, consulte Introducción a una distribución básica de CloudFront.

macOS

Siga los pasos que se indican a continuación para instalar `kubectl` en macOS. Estos pasos incluyen:

- Elegir y descargar el binario para la versión de Kubernetes que desee.
- Opcionalmente, comprobar la suma de comprobación del binario.
- Agregar ejecución a los permisos del binario.
- Copiar el binario en una carpeta en la RUTA.
- Opcionalmente, agregar el directorio del binario a la RUTA.

Procedimiento:

1. Descargue el dato binario para la versión de Kubernetes del clúster de Amazon S3.
   - 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. (Opcional) Compruebe el dato binario descargado con la suma de comprobación de `SHA-256` de su dato binario.

   a. Descargue la suma de comprobación de `SHA-256` de la versión de Kubernetes de su clúster.

   - 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. Verifique la suma de comprobación de `SHA-256` del dato binario descargado.

```
openssl sha1 -sha256 kubectl
```

c. Asegúrese de que la suma de comprobación generada en la salida coincida con la suma de comprobación del archivo de `kubectl.sha256` descargado.

3. Aplique permisos de ejecución al binario.

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

4. Copie el binario en una carpeta en `PATH`. Si ya ha instalado una versión de `kubectl`, recomendamos que cree un `$HOME/bin/kubectl` y se asegure de que `$HOME/bin` venga en primer lugar en su `$PATH`.

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

5. (Opcional) Agregue la ruta `$HOME/bin` a su archivo de inicialización del shell para que se configure cuando abra un shell.

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

Linux (amd64)

Siga los pasos que se indican a continuación para instalar `kubectl` en Linux (amd64). Estos pasos incluyen:

- Elegir y descargar el binario para la versión de Kubernetes que desee.
- Opcionalmente, comprobar la suma de comprobación del binario.
- Agregar ejecución a los permisos del binario.
- Copiar el binario en una carpeta en la RUTA.
- Opcionalmente, agregar el directorio del binario a la RUTA.

Procedimiento:

1. Descargue el dato binario de `kubectl` para la versión de Kubernetes del clúster de Amazon S3.
   - 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. (Opcional) Compruebe el dato binario descargado con la suma de comprobación de `SHA-256` de su dato binario.

   a. Descargue la suma de comprobación de `SHA-256` para la versión de Kubernetes del clúster desde Amazon S3, mediante el comando de la plataforma de hardware de su dispositivo.

   - 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. Controle la suma de comprobación de `SHA-256` del archivo binario descargado con uno de los siguientes comandos.

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

o

```
openssl sha1 -sha256 kubectl
```

c. Para la primera, debería ver `kubectl: OK`, para la segunda, puede comprobar que la suma de comprobación generada en la salida coincida con la suma de comprobación del archivo `kubectl.sha256` descargado.

3. Aplique permisos de ejecución al binario.

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

4. Copie el binario en una carpeta en `PATH`. Si ya ha instalado una versión de `kubectl`, recomendamos que cree un `$HOME/bin/kubectl` y se asegure de que `$HOME/bin` venga en primer lugar en su `$PATH`.

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

5. (Opcional) Agregue la ruta `$HOME/bin` a su archivo de inicialización del shell para que se configure cuando abra un shell.

> **Note**
>
> En este paso, se presupone que usa el shell Bash; si utiliza otro shell, cambie el comando para utilizar su archivo de inicialización del shell específico.

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

Linux (arm64)

Siga los pasos que se indican a continuación para instalar `kubectl` en Linux (arm64). Estos pasos incluyen:

- Elegir y descargar el binario para la versión de Kubernetes que desee.

- Opcionalmente, comprobar la suma de comprobación del binario.
- Agregar ejecución a los permisos del binario.
- Copiar el binario en una carpeta en la RUTA.
- Opcionalmente, agregar el directorio del binario a la RUTA.

Procedimiento:

1. Descargue el dato binario de `kubectl` para la versión de Kubernetes del clúster de Amazon S3.
   - 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. (Opcional) Compruebe el dato binario descargado con la suma de comprobación de `SHA-256` de su dato binario.

   a. Descargue la suma de comprobación de `SHA-256` para la versión de Kubernetes del clúster desde Amazon S3, mediante el comando de la plataforma de hardware de su dispositivo.

   - 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. Controle la suma de comprobación de `SHA-256` del archivo binario descargado con uno de los siguientes comandos.

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

o

```
openssl sha1 -sha256 kubectl
```

c. Para la primera, debería ver `kubectl: OK`, para la segunda, puede comprobar que la suma de comprobación generada en la salida coincida con la suma de comprobación del archivo `kubectl.sha256` descargado.

3. Aplique permisos de ejecución al binario.

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

4. Copie el binario en una carpeta en `PATH`. Si ya ha instalado una versión de `kubectl`, recomendamos que cree un `$HOME/bin/kubectl` y se asegure de que `$HOME/bin` venga en primer lugar en su `$PATH`.

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

5. (Opcional) Agregue la ruta `$HOME/bin` a su archivo de inicialización del shell para que se configure cuando abra un shell.

> **Note**
>
> En este paso, se presupone que usa el shell Bash; si utiliza otro shell, cambie el comando para utilizar su archivo de inicialización del shell específico.

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

Windows

Siga los pasos que se indican a continuación para instalar `kubectl` en Windows. Estos pasos incluyen:

- Elegir y descargar el binario para la versión de Kubernetes que desee.
- Opcionalmente, comprobar la suma de comprobación del binario.
- Copiar el binario en una carpeta en la RUTA.
- Opcionalmente, agregar el directorio del binario a la RUTA.

Procedimiento:

1. Abra un terminal de PowerShell.
2. Descargue el dato binario de `kubectl` para la versión de Kubernetes del clúster de Amazon S3.
   - 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. (Opcional) Compruebe el dato binario descargado con la suma de comprobación de SHA-256 de su dato binario.

   a. Descargue la suma de comprobación de SHA-256 para la versión de Kubernetes del clúster para Windows.

   - 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. Verifique la suma de comprobación de `SHA-256` del dato binario descargado.

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

c. Asegúrese de que la suma de comprobación generada en la salida coincida con la suma de comprobación del archivo de `kubectl.sha256` descargado. La salida de PowerShell debe ser una cadena de caracteres equivalente en mayúsculas.

4. Copie el binario en una carpeta en `PATH`. Si tiene un directorio existente en su `PATH` que utiliza para utilidades de línea de comandos, copie el binario en ese directorio. De lo contrario, lleve a cabo los pasos que figuran a continuación.

   a. Cree un nuevo directorio para los binarios de línea de comandos, como `C:\bin`.

   b. Copie el binario `kubectl.exe` en el nuevo directorio.

   c. Edite la variable de entorno `PATH` del usuario o sistema para agregar el nuevo directorio a su `PATH`.

   d. Cierre el terminal de PowerShell y abra uno nuevo para obtener la nueva variable `PATH`.

5. Una vez que instale `kubectl`, puede comprobar la versión.

```
kubectl version --client
```

6. Al instalar por primera vez `kubectl`, aún no está configurado para comunicarse con ningún servidor. Trataremos esta configuración según sea necesario en otros procedimientos. Si alguna vez necesita actualizar la configuración para comunicarse con un clúster en particular, puede ejecutar el siguiente comando. Reemplace *`region-code`* por la región de AWS en la que se encuentra el clúster. Reemplace *`my-cluster`* por el nombre de su clúster.

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

7. Considere configurar el autocompletado, que le permite usar la tecla de tabulación para completar los subcomandos de `kubectl` después de escribir las primeras letras. Consulte Kubectl autocomplete en la documentación de Kubernetes para obtener más información.

Instalar **`eksctl`**

La CLI de `eksctl` se usa para trabajar con clústeres de EKS. Automatiza muchas tareas individuales. Consulte Instalación en la documentación de `eksctl` para obtener instrucciones sobre cómo instalar `eksctl`. En el caso de Linux, utilice las instrucciones de UNIX.

Al usar `eksctl`, la entidad principal de seguridad de IAM que está utilizando debe contar con permisos para trabajar con los roles de IAM de Amazon EKS, los roles vinculados al servicio, AWS CloudFormation, además de una VPC y recursos relacionados. Para obtener más información, consulte Acciones y Uso de roles vinculados a servicios en la Guía del usuario de IAM. Debe completar todos los pasos de esta guía como el mismo usuario. Ejecute el siguiente comando para comprobar el usuario actual:

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

Siguientes pasos

- Inicio rápido: implementación de una aplicación web y almacenamiento de datos

Inicio rápido: implementación de una aplicación web y almacenamiento de datos

Este tutorial de inicio rápido lo guía por los pasos que se deben seguir para implementar la aplicación de muestra del videojuego 2048 y conservar sus datos en un clúster del modo automático de Amazon EKS mediante eksctl.

El modo automático de Amazon EKS simplifica la administración de clústeres mediante la automatización de tareas rutinarias como el almacenamiento en bloques, las redes, el equilibrio de carga y el escalado automático de computación. Durante la configuración, gestiona la creación de nodos con instancias administradas de EC2, equilibradores de carga de aplicación y volúmenes de EBS.

En resumen, implementará un ejemplo de carga de trabajo con las anotaciones personalizadas necesarias para una integración sin problemas con los servicios de AWS.

En este tutorial:

Utilizará la plantilla del clúster de `eksctl` que aparece a continuación para crear un clúster con el modo automático de EKS para el aprovisionamiento automatizado de nodos.

- Configuración de la VPC: al utilizar la plantilla del clúster de eksctl que aparece a continuación, eksctl crea automáticamente una nube privada virtual (VPC) IPv4 para el clúster. De forma predeterminada, eksctl configura una VPC que aborda todos los requisitos de red, además de crear puntos de conexión públicos y privados.
- Administración de instancias: el modo automático de EKS agrega o elimina nodos dinámicamente en el clúster de EKS en función de la demanda de las aplicaciones de Kubernetes.
- Persistencia de los datos: utilice la capacidad de almacenamiento en bloques del modo automático de EKS para garantizar la persistencia de los datos de la aplicación, incluso en situaciones en las que el pod se reinicie o se produzca un error.
- Acceso a aplicaciones externas: utilice la capacidad de equilibrio de carga del modo automático de EKS para aprovisionar dinámicamente un equilibrador de carga de aplicación (ALB).

Requisitos previos

Antes de comenzar, asegúrese de haber realizado las siguientes tareas:

- Configuración de su entorno para Amazon EKS
- Instalación de la versión más reciente de eksctl

Configuración del clúster

En esta sección, creará un clúster mediante el modo automático de EKS para el aprovisionamiento dinámico de nodos.

Cree un archivo denominado `cluster-config.yaml` y pegue en él el siguiente contenido. Sustituya `region-code` por una región válida (por ejemplo, `us-east-1`).

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

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

autoModeConfig:
  enabled: true
```

Ahora, estamos listos para crear el clúster.

Cree el clúster de EKS con mediante `cluster-config.yaml`:

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

> **Important**
>
> Si no usa eksctl para crear el clúster, debe etiquetar manualmente las subredes de la VPC.

Cree la IngressClass

Cree una `IngressClass` de Kubernetes para el modo automático de EKS. La IngressClass define cómo el modo automático de EKS gestiona los recursos de ingreso. En este paso se configura la capacidad de equilibrio de carga del modo automático de EKS. Al crear recursos de ingreso para las aplicaciones, el modo automático de EKS utiliza esta IngressClass para aprovisionar y administrar

automáticamente los equilibradores de carga, con lo que integra las aplicaciones de Kubernetes con los servicios de equilibrio de carga de AWS.

Guarde el siguiente archivo yaml como `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
```

Aplique la IngressClass al clúster:

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

Implemente la aplicación de muestra del videojuego 2048

En esta sección, explicamos los pasos para implementar el popular “juego 2048” como una aplicación de muestra dentro del clúster. El manifiesto proporcionado incluye anotaciones personalizadas para el equilibrador de carga de aplicación (ALB). Estas anotaciones se integran con EKS y le indican que gestione el tráfico HTTP entrante como “orientado a Internet” y lo dirija al servicio correspondiente en el espacio de nombres `game-2048` mediante el tipo de destino “ip”.

> **Note**
>
> La imagen `docker-2048` del ejemplo es una imagen de contenedor `x86_64` y no se ejecutará en otras arquitecturas.

1. Cree un espacio de nombres de Kubernetes llamado `game-2048` con la marca `--save-config`.

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

   Debería ver la siguiente respuesta de salida:

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

2. Implemente la aplicación de ejemplo del juego 2048.

```
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
```

   Este manifiesto configura una implementación, un servicio y una entrada de Kubernetes para el espacio de nombres `game-2048`, lo que crea los recursos necesarios para implementar y exponer la aplicación de `game-2048` dentro del clúster. Incluye la creación de un servicio denominado `service-2048` que expone la implementación en el puerto `80` y un recurso Ingress denominado `ingress-2048` que define las reglas de enrutamiento para el tráfico HTTP entrante y las anotaciones para un equilibrador de carga de aplicaciones (ALB) con acceso a Internet. Debería ver la siguiente respuesta de salida:

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

3. Ejecute el siguiente comando para obtener el recurso Ingress para el espacio de nombres `game-2048`.

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

   Debería ver la siguiente respuesta de salida:

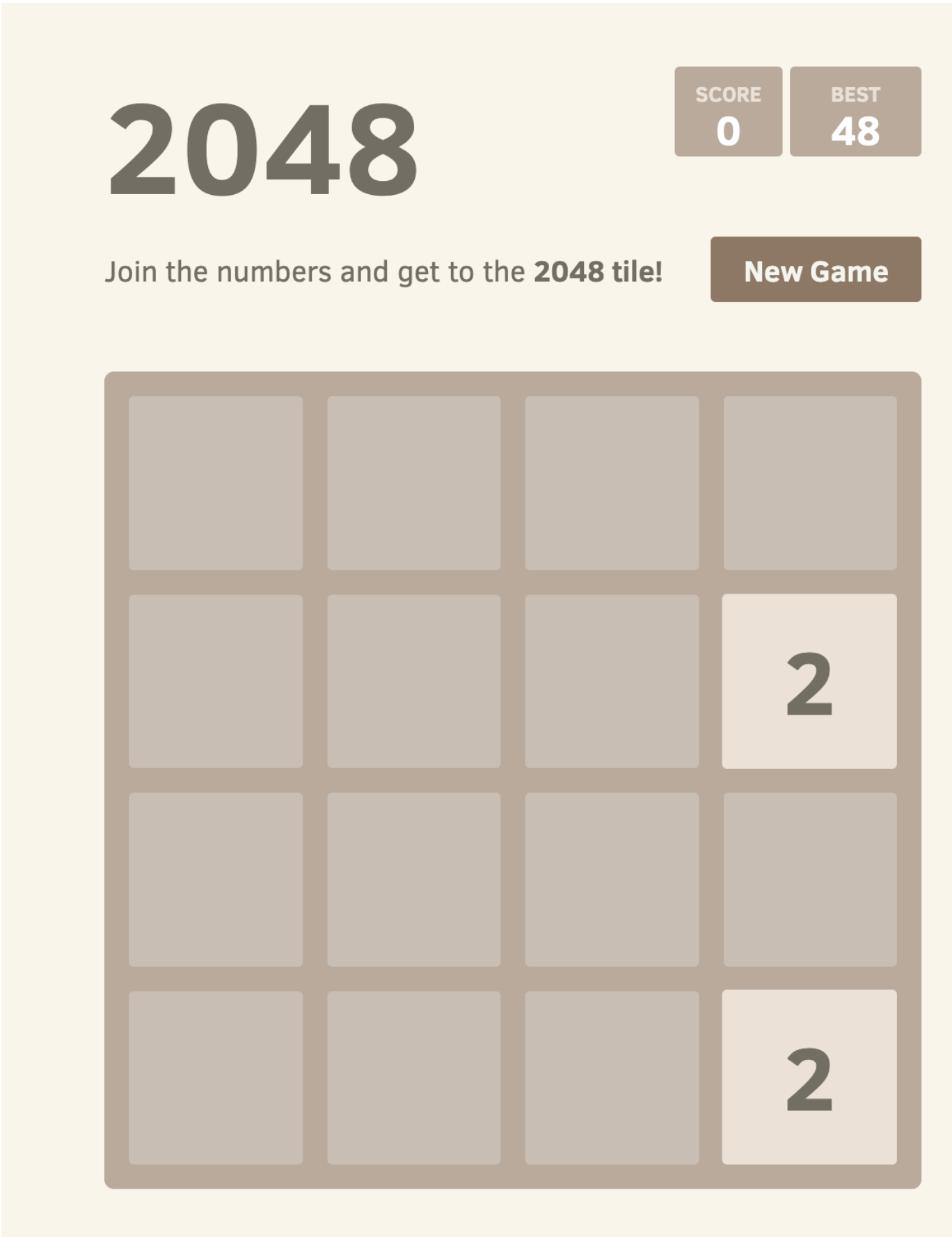
```
NAME           CLASS   HOSTS   ADDRESS
                     PORTS   AGE
ingress-2048   alb     *       k8s-game2048-ingress2-eb379a0f83-378466616.region-
code.elb.amazonaws.com   80      31s
```

   Deberá esperar varios minutos para que el equilibrador de carga de aplicación (ALB) se aprovisione antes de comenzar con los siguientes pasos.

4. Abra un navegador web y escriba la `ADDRESS` del paso anterior para acceder a la aplicación web. Por ejemplo:

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

   Debería ver el juego 2048 en el navegador. Juegue.


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

Conservación de los datos con el modo automático de Amazon EKS

Ahora que el videojuego 2048 se encuentra en funcionamiento en el clúster de Amazon EKS, es hora de asegurarse de que los datos del videojuego se conservan de forma segura mediante la capacidad de almacenamiento en bloque del modo automático de Amazon EKS.

1. Cree un archivo denominado `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. Aplicación de `StorageClass`:

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

3. Cree una reclamación de volumen persistente (PVC) para solicitar el almacenamiento de los datos de su juego. Cree un archivo llamado `ebs-pvc.yaml` y agregue el siguiente contenido:

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

4. Aplique el PVC en su clúster:

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

   Debería ver la siguiente respuesta de salida:

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

5. Ahora, tiene que actualizar la implementación del juego 2048 para usar este PVC para almacenar datos. La siguiente implementación está configurada para usar el PVC para almacenar los datos del juego. Cree un archivo llamado `ebs-deployment.yaml` que contenga lo siguiente:

```
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. Aplique la implementación actualizada:

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

Debería ver la siguiente respuesta de salida:

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

Tras seguir estos pasos, el videojuego 2048 deberá estar configurado en el clúster para conservar los datos mediante la capacidad de almacenamiento en bloque del modo automático de Amazon EKS. Esto garantiza que el progreso y los datos del juego estén seguros incluso en caso de que se produzca un fallo en el pod o en el nodo.

Si le ha gustado este tutorial, envíenos sus comentarios para que podamos ofrecerle más tutoriales de inicio rápido específicos para cada caso práctico, como este.

Limpieza

Para evitar incurrir en cargos futuros, debe eliminar manualmente la pila de CloudFormation asociada para borrar todos los recursos creados durante esta guía, incluida la red de la VPC.

Elimine la pila de CloudFormation:

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

Aprendizaje de Amazon EKS con ejemplos

Descripción general

Esta guía del usuario de Amazon EKS contiene procedimientos de uso general para crear su primer clúster de EKS desde la línea de comandos o la Consola de administración de AWS y una referencia sólida para todos los componentes principales de Amazon EKS. Sin embargo, como administrador o desarrollador de clústeres de Amazon EKS, puede obtener una comprensión más profunda de Amazon EKS si sigue las rutas de aprendizaje que existen en sitios ajenos a esta guía. Los siguientes sitios le ayudarán a:

- Configurar tipos específicos de clústeres. Los tipos de clústeres específicos pueden basarse en los tipos de carga de trabajo o en los requisitos de seguridad. Por ejemplo, es posible que desee ajustar un clúster para que ejecute cargas de trabajo por lotes, de machine learning o que requieran un uso intensivo de recursos informáticos.
- Mejorar sus clústeres. Puede agregar características avanzadas a su clúster para ofrecer aspectos como la observabilidad, el almacenamiento flexible, el escalado automático o la creación de redes de clústeres especializadas.
- Automatizar actualizaciones. Con características como GitOps, puede configurarlas para aprovisionar la infraestructura del clúster y las cargas de trabajo de forma automática, en función de los cambios que se produzcan en esos componentes de sus repositorios de Git.
- Usar herramientas avanzadas de configuración de clústeres. Si bien `eksctl` proporciona una forma rápida de crear un clúster, existen otras herramientas que pueden facilitar la configuración y la actualización de clústeres más complejos. Estas incluyen herramientas como Terraform y CloudFormation.

Para comenzar su ruta de aprendizaje de Amazon EKS, le recomendamos que visite algunos de los sitios que se describen en esta página. Si tiene problemas en el camino, también hay recursos que le ayudarán a superarlos. Por ejemplo, el Centro de conocimiento de re:Post le permite buscar en la base de datos de soporte los problemas de soporte relacionados con Amazon EKS. Además, la Guía de prácticas recomendadas de Amazon EKS ofrece consejos sobre las mejores formas de configurar sus clústeres aptos para producción.

Taller de Amazon EKS

El taller de Amazon EKS, que comienza con una comprensión básica de Kubernetes y los contenedores, es una plataforma de aprendizaje para explicar a un administrador de clústeres las características importantes de Amazon EKS. Estas son las formas en las que puede participar en el taller de Amazon EKS:

- Conceptos básicos de Amazon EKS: vea el vídeo de la página de introducción para obtener información sobre cómo Amazon EKS implementa las características de Kubernetes en la nube de AWS. Si necesita información aún más básica sobre Kubernetes, vea el vídeo What is Kubernetes.
- Configuración de Amazon EKS: si tiene una cuenta de AWS, la sección Configuración le ayuda a configurar un entorno de CloudShell para crear un clúster. Ofrece la posibilidad de elegir entre eksctl (una línea de comandos sencilla para la creación de clústeres) y Terraform (un enfoque más basado en la infraestructura como código para crear un clúster) para crear su clúster de Amazon EKS.
- Introducción a Amazon EKS: pruebe una tienda web sencilla en la sección Aplicación de muestra. Puede usarla en todos los demás ejercicios. En esta sección, también puede obtener información sobre cómo empaquetar imágenes de contenedores y cómo se administran los microservicios mediante los pods, implementaciones, servicios, conjuntos de estados y espacios de nombres de Kubernetes. A continuación, use Kustomize para implementar cambios en los manifiestos de Kubernetes.
- Conceptos básicos de Amazon EKS: mediante características de AWS como el controlador de equilibrador de carga de AWS, el taller le muestra cómo exponer sus aplicaciones al mundo. En cuanto al almacenamiento, el taller muestra cómo utilizar Amazon EBS para el almacenamiento de bloques, Amazon EFS para el almacenamiento de sistemas de archivos y Amazon FSx para NetApp ONTAP para administrar los sistemas de archivos ONTAP en AWS. Para la administración de nodos, el taller le ayudará a configurar grupos de nodos administrados.
- Características avanzadas de Amazon EKS: las características más avanzadas que se ofrecen en el taller de Amazon EKS incluyen laboratorios para configurar:
  - Escalado automático: incluye el escalado automático de nodos (con el Escalador automático de clústeres o Karpenter) y el escalado automático de la carga de trabajo (con el Escalador automático de pods horizontales y el Escalador automático proporcional de clústeres).
  - Observabilidad: obtenga información sobre el registro, OpenSearch, Información de contenedores en Amazon EKS y Visibilidad de los costos con Kubecost en un conjunto de Laboratorios de observabilidad.

- Seguridad: este conjunto de Laboratorios de seguridad le permite explorar Administración de secretos, Amazon GuardDuty, Estándares de seguridad de pods y Administración de políticas de Kyverno.
- Redes: conozca las características de redes de Amazon EKS en los laboratorios de Redes, que incluyen Amazon VPC CNI (complementos de red compatibles) y Amazon VPC Lattice (para configurar clústeres en cuentas de VC y de usuario).
- Automatización: los laboratorios de Automatización lo guían a través de los métodos de GitOps para administrar sus clústeres y proyectos, como AWS Controllers for Kubernetes y Crossplane para administrar los planos de control de Amazon EKS.

Tutoriales prácticos de configuración de clústeres de Amazon EKS

Un conjunto de tutoriales de configuración de clústeres de Amazon EKS en el sitio de la comunidad de AWS puede ayudarle a crear clústeres de Amazon EKS con fines especiales y a mejorarlos de diversas maneras. Los tutoriales se dividen en tres tipos diferentes:

Creación de clústeres

Estos tutoriales le ayudan a crear clústeres que se pueden utilizar para fines especiales. Las características especiales incluyen la capacidad de ejecutar:

- Aplicaciones escalables basadas en IPv6 en todo el mundo
- Tareas asíncronas por lotes
- Microservicios de alto tráfico
- Escalado automático con Karpenter en Fargate
- Cargas de trabajo financieras
- Grupos de nodos administrados por Windows

Mejora de los clústeres

Una vez que tenga un clúster existente, puede ampliarlo y mejorarlo de manera que pueda ejecutar cargas de trabajo especializadas y, de otro modo, mejorar los clústeres. Estos tutoriales incluyen formas de:

- Proporcionar soluciones de almacenamiento con EFS CSI

- Proporcionar almacenamiento dinámico de bases de datos con EBS CSI
- Exponer aplicaciones en clústeres IPv4 mediante el controlador de equilibrador de carga de AWS
- Exponer aplicaciones en clústeres IPv6 mediante el controlador de equilibrador de carga de AWS

Optimización de los servicios de AWS

Con estos tutoriales, podrá integrar mejor sus clústeres con los servicios de AWS. Estos tutoriales incluyen aquellos que le ayudarán a:

- Administrar registros de DNS para microservicios con ExternalDNS
- Supervisar aplicaciones con CloudWatch
- Administrar las tareas asíncronas con el almacenamiento de SQS y EFS
- Consumir secretos de AWS Secrets Manager de las cargas de trabajo
- Configurar mTLS con Fargate, NGINX y ACM PCA

Amazon EKS Samples

El repositorio Amazon EKS Samples almacena los manifiestos para utilizarlos con Amazon EKS. Estos manifiestos le dan la oportunidad de probar diferentes tipos de aplicaciones en Amazon EKS o crear tipos específicos de clústeres de Amazon EKS. Los ejemplos incluyen manifiestos para:

- Crear un clúster de AWS Amazon EKS Fargate
- Crear un clúster con un rol de IAM existente
- Agregar un grupo de nodos administrado por Ubuntu a un clúster
- Hacer una copia de seguridad y restauración del almacenamiento de pods con instantáneas de volumen
- Recuperar los volúmenes de EBS montados como PVC con varias cuentas
- Habilitar el protocolo proxy para el controlador de entrada NGINX con los equilibradores de carga clásicos
- Configurar el inicio de sesión en Fargate para AWS OpenSearch
- Ejecutar el SDK de Python con un proveedor de identidades federado web
- Implementar una aplicación de muestra en un controlador CSI de NFS
- Utilizar instantáneas de volumen para StatefulSets

- Implementar pods en todos los nodos de diferentes zonas de disponibilidad

Tenga en cuenta que estos ejemplos son únicamente para fines de aprendizaje y pruebas y no están pensados para usarse en producción.

Tutoriales de AWS

El sitio AWS Tutoriales publica algunos tutoriales de Amazon EKS, pero también ofrece una herramienta de búsqueda para encontrar otros tutoriales publicados en AWS (como Comunidad de AWS). Los tutoriales de Amazon EKS publicados directamente en este sitio incluyen:

- Implementar una aplicación web de contenedores en Amazon EKS
- Ejecutar clústeres de Kubernetes por menos dinero (instancias de Amazon EKS y de spot)
- Cómo optimizar los costos de los trabajos de Jenkins en Kubernetes

Taller de desarrolladores

Si es desarrollador de software y desea crear o refactorizar aplicaciones para que se ejecuten en Amazon EKS, el taller para desarrolladores de Amazon EKS es un buen punto de partida. El taller no solo lo ayuda a crear aplicaciones en contenedores, sino que también lo ayuda a implementar esos contenedores en un registro de contenedores (ECR) y, desde allí, en un clúster de Amazon EKS.

Comience con el taller de Python de Amazon EKS para llevar a cabo el proceso de refactorización de una aplicación de Python y, a continuación, configure su entorno de desarrollo para prepararlo para la implementación de la aplicación. Revise las secciones sobre contenedores, Kubernetes y Amazon EKS para prepararse para ejecutar sus aplicaciones en contenedores en esos entornos.

Taller de Terraform

Si bien `eksctl` es una herramienta sencilla para crear un clúster, para los tipos más complejos de infraestructura como código de las implementaciones de Amazon EKS, Terraform es una popular herramienta de creación y administración de clústeres de Amazon EKS. El taller Amazon EKS de Terraform enseña cómo usar Terraform para crear una VPC de AWS, crear clústeres de Amazon EKS y añadir mejoras opcionales a su clúster. En concreto, hay una sección para crear un clúster privado de Amazon EKS

Formación en AWS Amazon EKS

AWS ofrece formación formal para obtener información sobre Amazon EKS. Un curso de formación de tres días titulado Running Containers on Amazon Elastic Kubernetes Service enseña:

- Conceptos básicos de Kubernetes y Amazon EKS
- Cómo crear clústeres de Amazon EKS
- Protección de Amazon EKS con la autorización RBAC de AWS IAM y Kubernetes
- Herramientas de automatización de GitOps
- Herramientas de supervisión
- Técnicas para mejorar los costos, la eficiencia y la resiliencia

Introducción a Amazon EKS

Asegúrese de tener configurado el uso de Amazon EKS antes de seguir las guías de introducción. Para obtener más información, consulte Configuración.

Existen dos guías de introducción disponibles para crear un clúster de Kubernetes nuevo con nodos en Amazon EKS:

- Introducción a Amazon EKS (eksctl): esta guía de introducción lo ayuda a instalar todos los recursos necesarios a fin de comenzar a utilizar Amazon EKS mediante `eksctl`, una utilidad de línea de comandos sencilla para crear y administrar clústeres de Kubernetes en Amazon EKS. Al final del tutorial, contará con un clúster de Amazon EKS en ejecución en el que puede implementar aplicaciones. Esta es la forma más sencilla y rápida de comenzar a utilizar Amazon EKS.
- Introducción a Amazon EKS Consola de administración de AWS y AWS CLI: esta guía de introducción lo ayuda a crear todos los recursos necesarios para comenzar a utilizar Amazon EKS con la Consola de administración de AWS y la AWS CLI. Al final del tutorial, contará con un clúster de Amazon EKS en ejecución en el que puede implementar aplicaciones. En esta guía, creará cada recurso necesario para un clúster de Amazon EKS de forma manual. Los procedimientos proporcionan una visibilidad de cómo se crea cada recurso y cómo interactúan entre sí.

También ofrecemos las siguientes referencias:

- Para ver una colección de tutoriales prácticos, consulte Configuración del clúster de EKS en la comunidad de AWS.
- Para ver ejemplos de código, consulte Ejemplos de código para Amazon EKS con las SDK deAWS.

Introducción a Amazon EKS: modo automático de EKS

Al igual que ocurre en otras experiencias de introducción de EKS, al crear el primer clúster con el modo automático de EKS, se delega la administración del clúster en sí a AWS. Sin embargo, el modo automático de EKS mejora la automatización de EKS al delegar la responsabilidad de muchos servicios esenciales para configurar la infraestructura de carga de trabajo (nodos, redes y diversos servicios), lo que simplifica la administración de los nodos y su escalado vertical para satisfacer las demandas de la carga de trabajo.

Elija una de las siguientes formas de crear un clúster con el modo automático de EKS:

- the section called " AWS CLI": utilice la interfaz de la línea de comandos de `aws` para crear un clúster.
- the section called "Consola de administración": utilice la Consola de administración de AWS para crear un clúster.
- the section called "CLI de eksctl": utilice la interfaz de la línea de comandos de `eksctl` para crear un clúster.

Si compara diferentes enfoques para crear el primer clúster de EKS, debe saber que al utilizar el modo automático de EKS, AWS asume responsabilidades adicionales de administración del clúster, incluida la configuración de componentes para:

- Iniciar y escalar los nodos a medida que la demanda de la carga de trabajo aumente o disminuya.
- Actualizar periódicamente el propio clúster (plano de control), los sistemas operativos de los nodos y los servicios que se ejecutan en los nodos.
- Elegir la configuración predeterminada que define aspectos como el tamaño y la velocidad del almacenamiento de los nodos, así como la configuración de la red de pods.

Para obtener más información sobre lo que se obtiene con los clústeres del modo automático de EKS, consulte Modo automático de EKS.

Introducción a Amazon EKS: **`eksctl`**

> **Note**
>
> En este tema se explica cómo comenzar sin el modo automático de EKS.
> El modo automático de EKS automatiza las tareas rutinarias de computación en clústeres, almacenamiento y redes. Descubra cómo comenzar a utilizar el modo automático de Amazon EKS.

Esta guía lo ayuda a crear todos los recursos necesarios a fin de comenzar a utilizar Amazon Elastic Kubernetes Service (Amazon EKS) mediante `eksctl`, una utilidad de línea de comandos sencilla para crear y administrar clústeres de Kubernetes en Amazon EKS. Al final de este tutorial, contará con un clúster de Amazon EKS en ejecución en el que puede implementar aplicaciones.

Los procedimientos de esta guía crean varios recursos de forma automática que debe establecer manualmente al crear el clúster mediante la Consola de administración de AWS. Si prefiere crear la mayoría de los recursos de forma manual y comprender mejor cómo interactúan entre sí, utilice la Consola de administración de AWS para crear el clúster y la informática. Para obtener más información, consulte the section called “Creación de un clúster (consola y CLI)”.

Requisitos previos

Antes de comenzar este tutorial, debe instalar y configurar las herramientas de la AWS CLI, kubectl y eksctl tal y como se describe en Configuración para usar Amazon EKS.

Paso 1: creación del clúster y de los nodos de Amazon EKS

> **Important**
>
> Para comenzar de la manera más sencilla y rápida posible, este tema incluye pasos a fin de crear un clúster y nodos con la configuración predeterminada. Antes de crear un clúster y nodos para su uso en producción, recomendamos que conozca toda la configuración e implemente un clúster y nodos con la configuración que satisfaga sus requisitos. Para obtener más información, consulte the section called “Creación de un clúster ” y Administración de la computación. Algunos ajustes de configuración solo se pueden habilitar al crear el clúster y los nodos.

Puede crear un clúster con uno de los siguientes tipos de nodos. Para obtener más información sobre cada tipo, consulte Administración de la computación. Después de implementar el clúster, puede agregar otros tipos de nodos.

- Fargate – Linux: seleccione este tipo de nodo si desea ejecutar aplicaciones de Linux en the section called “ AWS Fargate”. Fargate es un motor de computación sin servidor que le permite implementar pods de Kubernetes sin administrar instancias de Amazon EC2.
- Nodos administrados – Linux: seleccione este tipo de nodo si desea ejecutar aplicaciones de Amazon Linux en instancias de Amazon EC2. Aunque no se trata en esta guía, también puede agregar nodos Autoadministrados de Windows y Bottlerocket a su clúster.

Cree su clúster de Amazon EKS con el comando siguiente. Puede reemplazar `my-cluster` con su propio valor. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas

y minúsculas) y guiones. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster. Reemplace *region-code* por cualquier región de AWS en la que se admita Amazon EKS. Para ver una lista de las regiones de AWS, consulte Puntos de conexión y cuotas de Amazon EKS en la Guía de referencia general de AWS.

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
```

La creación del clúster tarda varios minutos. Durante la creación, verá varias líneas de salida. La última línea de salida es similar a la siguiente línea de ejemplo.

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

`eksctl` creó un archivo de configuración `kubectl` en `~/.kube/config` o agregó la configuración del clúster nuevo dentro de un archivo de configuración en `~/.kube/config` en su equipo.

Después de que se complete la creación del clúster, consulte la pila de AWS CloudFormation denominada `eksctl-`*`my-cluster`*`-cluster` en la consola de AWS CloudFormation para ver todos los recursos que se crearon.

Paso 2: ver los recursos de Kubernetes

1. Visualización de los nodos del clúster.

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

   Un ejemplo de salida sería el siguiente.

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
```

Para obtener más información acerca de lo que ve en la salida, consulte the section called “Acceso a recursos del clúster”.

2. Visualización de las cargas de trabajo que se ejecutan en el clúster.

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

Un ejemplo de salida sería el siguiente.

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>
```

Para obtener más información acerca de lo que ve en la salida, consulte the section called “Acceso a recursos del clúster”.

Paso 3: eliminación de sus clústeres y nodos

Cuando haya terminado con el clúster y los nodos que creó para este tutorial, deberá eliminarlos con el siguiente comando. Si quiere hacer más con este clúster antes de eliminarlo, consulte the section called “Pasos a seguir a continuación”.

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

Pasos a seguir a continuación

Los siguientes temas de documentación lo ayudarán a ampliar la funcionalidad de su clúster.

- Implemente una aplicación de muestra en su clúster.
- La entidad principal de IAM que creó el clúster es la única entidad principal que puede hacer llamadas al servidor de API de Kubernetes con `kubectl` o la Consola de administración de AWS. Si desea que otras entidades principales de IAM tengan acceso al clúster, debe agregarlas. Para obtener más información, consulte the section called “Acceso a la API de Kubernetes” y the section called “Permisos necesarios”.
- Antes de implementar un clúster para su uso en producción, le recomendamos que se familiarice con toda la configuración de clústeres y nodos. Algunos ajustes (como habilitar el acceso SSH a los nodos de Amazon EC2) deben establecerse cuando se crea el clúster.
- Para aumentar la seguridad del clúster, configure el complemento de interfaz de red de contenedores de Amazon VPC a fin de utilizar roles de IAM para cuentas de servicio.

Introducción a Amazon EKS: Consola de administración de AWS y AWS CLI

> **Note**
>
> En este tema se explica cómo comenzar sin el modo automático de EKS. Utiliza grupos de nodos administrados para implementar nodos.
> El modo automático de EKS automatiza las tareas rutinarias de computación en clústeres, almacenamiento y redes. Descubra cómo comenzar a utilizar el modo automático de Amazon EKS. El modo automático de EKS es el método preferente para implementar nodos.

Esta guía lo ayuda a crear todos los recursos necesarios para comenzar a utilizar Amazon Elastic Kubernetes Service (Amazon EKS) mediante la Consola de administración de AWS y la AWS CLI. En esta guía, creará cada recurso de forma manual. Al final de este tutorial, contará con un clúster de Amazon EKS en ejecución en el que puede implementar aplicaciones.

Los procedimientos de esta guía le dan una visibilidad completa sobre cómo se crea cada recurso y cómo interactúan los recursos entre sí. Si prefiere que la mayoría de los recursos se creen de forma automática, utilice la CLI de `eksctl` para crear el clúster y los nodos. Para obtener más información, consulte the section called “Creación de un clúster (eksctl)”.

Requisitos previos

Antes de comenzar este tutorial, debe instalar y configurar las siguientes herramientas y recursos que necesitará para crear y administrar un clúster de Amazon EKS.

- AWS CLI: una herramienta de línea de comandos para trabajar con servicios de AWS, incluido Amazon EKS. Para obtener más información, consulte Instalación en la Guía del usuario de la interfaz de la línea de comandos de AWS. Después de instalar la AWS CLI, recomendamos que también la configure. Para obtener más información, consulte Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. Tenga en cuenta que necesita la v2 de la AWS CLI para utilizar la opción update-kubeconfig que se muestra en esta página.
- **`kubectl`**: una herramienta de línea de comandos para trabajar con clústeres de Kubernetes. Para obtener más información, consulte the section called “Configure kubectl y eksctl”.
- Permisos de IAM necesarios: la entidad principal de seguridad de IAM que está utilizando debe contar con permisos para trabajar con los roles de IAM de Amazon EKS, los roles vinculados al servicio, AWS CloudFormation, una VPC y recursos relacionados. Para obtener más información, consulte Acciones y Uso de roles vinculados a servicios en la Guía del usuario de IAM. Debe completar todos los pasos de esta guía como el mismo usuario. Ejecute el siguiente comando para comprobar el usuario actual:

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

Le recomendamos que siga los pasos de este tema en un intérprete de comandos Bash. Si no está utilizando un intérprete de comandos Bash, algunos comandos de script, como los caracteres de continuación de línea y la forma en que se establecen y utilizan las variables, requieren ajustes para su intérprete de comandos. Además, las reglas de entrecomillado y escape de su intérprete de comandos pueden ser diferentes. Para obtener más información, consulte Uso de entrecomillado de cadenas en la AWS CLI de la Guía del usuario de la interfaz de la línea de comandos de AWS.

Paso 1: creación del clúster de Amazon EKS

> **Important**
>
> Para comenzar de la manera más sencilla y rápida posible, en este tema se incluye pasos a fin de crear un clúster con la configuración predeterminada. Antes de crear un clúster para su uso en producción, recomendamos que conozca toda la configuración e implemente un clúster con la configuración que satisfaga sus requisitos. Para obtener más información, consulte the section called “Creación de un clúster ”. Algunos ajustes de configuración solo se pueden habilitar al crear el clúster.

1. Cree una Amazon VPC con subredes privadas y públicas que cumplan con los requisitos de Amazon EKS. Reemplace *region-code* por cualquier región de AWS en la que se admita Amazon EKS. Para ver una lista de las regiones de AWS, consulte Puntos de conexión y cuotas de Amazon EKS en la Guía de referencia general de AWS. Puede reemplazar *my-eks-vpc-stack* por el nombre que elija.

   ```
   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**
   >
   > Para obtener una lista de todos los recursos que el comando anterior crea, abra la consola de AWS CloudFormation en https://console.aws.amazon.com/cloudformation/. Elija la pila *my-eks-vpc-stack* y, a continuación, elija la pestaña Resources (Recursos).

2. Cree un rol de IAM de clúster y adjúntelo a la política administrada de IAM de Amazon EKS. Los clústeres de Kubernetes administrados por Amazon EKS realizan llamadas a otros servicios de AWS en su nombre para administrar los recursos que utiliza con el servicio.

   a. Copie el siguiente contenido en un archivo con el nombre *eks-cluster-role-trust-policy.json*.

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

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

b. Cree el rol.

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

c. Adjunte la política administrada de IAM por Amazon EKS requerida al rol.

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

3. Abra la consola de Amazon EKS en eks/home#/clusters.

   Asegúrese de que la región de AWS que se muestra en la parte superior derecha de la consola sea la región de AWS en la que desea crear el clúster. De lo contrario, elija el menú desplegable junto al nombre de la región de AWS y elija la región de AWS que desea utilizar.

4. Elija Create cluster. Si no ve esta opción, elija Clústeres en el panel de navegación izquierdo primero.

5. En la página Configure cluster (Configurar clúster), haga lo siguiente:

   a. Seleccione Configuración personalizada y desactive Utilizar el modo automático de EKS. (Si prefiere un clúster del modo automático de EKS, consulte the section called “Consola de administración” en su lugar).

   b. Ingrese un Nombre para el clúster, como *my-cluster*. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster.

   c. En Cluster Service Role (Rol de servicio de clúster), elija *myAmazonEKSClusterRole*.

d. Conserve el resto de la configuración con sus valores predeterminados y elija Next (Siguiente).

6. En la página Specify networking (Especificar redes), haga lo siguiente:
   a. Elija el ID de la VPC que creó en un paso anterior en la lista desplegable de VPC. Es algo similar a *`* | my-eks-vpc-stack-VPC`*.
   b. Seleccione, en la lista desplegable Subredes, las subredes creadas en un paso anterior. Las subredes serán algo como *`* | my-eks-vpc-stack-*`*.
   c. Elija el grupo de seguridad creado en un paso anterior en la lista desplegable Grupos de seguridad adicionales. Es algo como *`* | my-eks-vpc-stack-ControlPlaneSecurityGroup-*`*.
   d. Conserve el resto de la configuración con sus valores predeterminados y elija Siguiente.
7. En la página Configurar observabilidad, elija Siguiente.
8. En la página Seleccionar complementos, elija Siguiente.

   Para obtener más información sobre los complementos, consulte the section called "Complementos de Amazon EKS".
9. En la página Configurar las opciones de complementos seleccionados, elija Siguiente.
10. En la página Review and create (Revisar y crear), elija Create (Crear).

    A la derecha del nombre del clúster, el estado del clúster es En creación durante varios minutos hasta que se complete el proceso de aprovisionamiento del clúster. No siga con el paso siguiente hasta que el estado sea Activo.

> **Note**
>
> Es posible que reciba un error que indique que una de las zonas de disponibilidad de la solicitud no tiene capacidad suficiente para crear un clúster de Amazon EKS. Si esto ocurre, el mensaje de error indicará las zonas de disponibilidad que admiten un clúster nuevo. Intente crear el clúster de nuevo con al menos dos subredes ubicadas en las zonas de disponibilidad admitidas para su cuenta. Para obtener más información, consulte the section called "Capacidad insuficiente".

Paso 2: configuración del equipo para comunicarse con el clúster

En esta sección creará un archivo de `kubeconfig` para el clúster. La configuración de este archivo permite a la CLI de `kubectl` comunicarse con el clúster.

Antes de continuar, asegúrese de que la creación del clúster se haya completado correctamente en el paso 1.

1. Creación o actualización de un archivo de `kubeconfig` para el clúster. Reemplace *`region-code`* con la región de AWS en la creó el clúster. Reemplace *`my-cluster`* por el nombre de su clúster.

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

   De forma predeterminada, el archivo de `config` se crea en `~/.kube` o la configuración del clúster nuevo se agrega a un archivo de `config` existente en `~/.kube`.

2. Pruebe la configuración.

   ```
   kubectl get svc
   ```

   > **Note**
   >
   > Si recibe cualquier error de tipo de recurso o autorización, consulte the section called "Acceso denegado o no autorizado (`kubectl`)" en el tema de solución de problemas.

   Un ejemplo de salida sería el siguiente.

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

Paso 3: creación de nodos

> **Important**
>
> Para comenzar de la forma más sencilla y rápida posible, en este tema se incluyen pasos para crear nodos principalmente con configuraciones predeterminadas. Antes de crear nodos para su uso en producción, recomendamos que conozca toda la configuración e implemente nodos con la configuración que satisfaga sus requisitos. Para obtener más información,

consulte Administración de la computación. Algunos ajustes de configuración solo se pueden habilitar al crear los nodos.

Con este procedimiento se configura el clúster para utilizar grupos de nodos administrados en la creación de nodos, para lo cual se especifican las subredes y el rol de IAM de nodo que se crearon en los pasos anteriores. Permite ejecutar aplicaciones de Amazon Linux en instancias de Amazon EC2.

Para obtener más información sobre las diferentes formas de configurar los nodos en EKS, consulte Administración de la computación. Después de implementar el clúster, puede agregar otros tipos de nodos. Aunque no se trata en esta guía, también puede agregar nodos Autoadministrados de Windows y Bottlerocket a su clúster.

Creación de un grupo de nodos administrados de Linux de EC2

1. Cree un rol de IAM de nodo y adjúntelo a la política administrada de IAM de Amazon EKS. El daemon de `kubelet` del nodo de Amazon EKS realiza llamadas a las API de AWS en su nombre. Los nodos reciben permisos de dichas llamadas de API a través de políticas asociadas y de un perfil de instancias de IAM.
   a. Copie el siguiente contenido en un archivo denominado `node-role-trust-policy.json`.

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

   b. Cree el rol de IAM de nodo.

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

c. Adjunte las políticas de IAM administradas requeridas al rol.

```
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. Abra la consola de Amazon EKS en eks/home#/clusters.

e. Elija el nombre del clúster que creó en Paso 1: crear el clúster de Amazon EKS, por ejemplo, *my-cluster*.

f. En la página ***my-cluster*** , haga lo siguiente:

g. Elija la pestaña Compute (Computación).

h. Elija Add Node Group (Agregar grupo de nodos).

2. En la página Configure Node Group (Configurar grupo de nodos), haga lo siguiente:

   a. En Nombre, ingrese un nombre único para el grupo de nodos administrados, por ejemplo, *my-nodegroup*. El nombre del grupo de nodos no puede tener más de 63 caracteres. Debe empezar por una letra o un dígito, pero también puede incluir guiones y guiones bajos como caracteres no iniciales.

   b. En Node IAM role name (Nombre de rol de IAM de nodo), elija el rol *myAmazonEKSNodeRole* que creó en el paso anterior. Recomendamos que cada grupo de nodos utilice su propio rol de IAM exclusivo.

   c. Elija Siguiente.

3. En la página Set compute and scaling configuration (Establecer la configuración de computación y escalado), acepte los valores predeterminados y elija Next (Siguiente).

4. En la página Specify networking (Especificar redes), acepte los valores predeterminados y elija Next (Siguiente).

5. En la página Revisar y crear, revise la configuración del grupo de nodos administrados y elija Crear.

6. Al cabo de varios minutos, el Status (Estado) en la Node Group configuration (Configuración del grupo de nodos) cambiará de Creating (En creación) a Active (Activo). No siga con el paso siguiente hasta que el estado sea Activo.

Paso 4: visualización de recursos

Puede ver sus nodos y cargas de trabajo de Kubernetes.

1. En el panel de navegación izquierdo, elija Clusters (Clústeres). En la lista de Clusters (Clústeres), elija el nombre del clúster que ha creado, como *my-cluster*.
2. En la página ***my-cluster*** , elija lo siguiente:
   a. Pestaña Computación: verá la lista de Nodos implementados para el clúster. Puede elegir el nombre de un nodo para obtener más información sobre él.
   b. Pestaña Recursos: consulte todos los recursos de Kubernetes que se implementan de forma predeterminada en un clúster de Amazon EKS. Seleccione cualquier tipo de recurso en la consola de para obtener más información sobre él.

Paso 5: eliminación de recursos

Cuando haya terminado con el clúster y los nodos que creó para este tutorial, deberá eliminar los recursos creados. Si desea hacer más con este clúster antes de eliminar los recursos, consulte the section called “Siguientes pasos”.

1. Elimine cualquier perfil de grupo de nodos que haya creado.
   a. Abra la consola de Amazon EKS en eks/home#/clusters.
   b. En el panel de navegación izquierdo, elija Clusters (Clústeres). En la lista de clústeres, elija *my-cluster*.
   c. Elija la pestaña Computación.
   d. Si ha creado un grupo de nodos, elija el grupo de nodos *my-nodegroup* y, luego, elija Delete (Eliminar). Ingrese *my-nodegroup* y, a continuación, seleccione Eliminar.
   e. No continúe hasta que se hayan eliminado los perfiles de grupos de nodos.
2. Elimine el clúster.
   a. En el panel de navegación izquierdo, elija Clusters (Clústeres). En la lista de clústeres, elija *my-cluster*.
   b. Seleccione Delete cluster (Eliminar clúster).
   c. Ingrese *my-cluster* y, a continuación, seleccione Eliminar. No continúe hasta que se elimine el clúster.
3. Elimine la pila de AWS CloudFormation de la VPC que ha creado.

   a. Abra la consola de CloudFormation en https://console.aws.amazon.com/cloudformation/.
   b. Elija la pila *my-eks-vpc-stack* y, luego, elija Delete (Eliminar).
   c. En el cuadro de diálogo de confirmación Eliminar ***my-eks-vpc-stack***, elija Eliminar pila.
4. Elimine los roles de IAM que ha creado.
   a. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
   b. En el panel de navegación izquierdo, elija Roles.
   c. Seleccione cada rol que haya creado de la lista (***myAmazonEKSClusterRole***, así como *myAmazonEKSNodeRole*). Elija Delete (Eliminar), ingrese el texto de confirmación solicitado y, luego, elija Delete (Eliminar).

Siguientes pasos

Los siguientes temas de documentación lo ayudarán a ampliar la funcionalidad de su clúster.

- La entidad principal de IAM que creó el clúster es la única entidad principal que puede hacer llamadas al servidor de API de Kubernetes con `kubectl` o la Consola de administración de AWS. Si desea que otras entidades principales de IAM tengan acceso al clúster, debe agregarlas. Para obtener más información, consulte the section called “Acceso a la API de Kubernetes” y the section called “Permisos necesarios”.
- Implemente una aplicación de muestra en su clúster.
- Antes de implementar un clúster para su uso en producción, le recomendamos que se familiarice con toda la configuración de clústeres y nodos. Algunos ajustes (como habilitar el acceso SSH a los nodos de Amazon EC2) deben establecerse cuando se crea el clúster.
- Para aumentar la seguridad del clúster, configure el complemento de interfaz de red de contenedores de Amazon VPC a fin de utilizar roles de IAM para cuentas de servicio.

Automatización de la infraestructura de clústeres con el modo automático de EKS

El modo automático de EKS amplía la administración de AWS de los clústeres de Kubernetes más allá del propio clúster, para permitir que AWS también configure y administre la infraestructura que permite el buen funcionamiento de las cargas de trabajo. Puede delegar decisiones clave de infraestructura y aprovechar la experiencia y los conocimientos de AWS para las operaciones cotidianas. La infraestructura de clúster administrada por AWS incluye numerosas capacidades de Kubernetes como componentes principales, en lugar de complementos, como el escalado automático de computación, las redes de pods y servicios, el equilibrio de carga de aplicaciones, el DNS de clúster, el almacenamiento en bloque y la compatibilidad con GPU.

Para comenzar, puede implementar un nuevo clúster de modo automático de EKS o habilitar el modo automático de EKS en un clúster existente. Puede implementar, actualizar o modificar los clústeres del modo automático de EKS mediante eksctl, la AWS CLI, la Consola de administración de AWS, las API de EKS o las herramientas de infraestructura como código que prefiera.

Con el modo automático de EKS, aún podrá utilizar las herramientas compatibles con Kubernetes que prefiera. El modo automático de EKS se integra con servicios de AWS, como Amazon EC2, Amazon EBS y ELB, y aprovecha los recursos en la nube AWS que siguen las prácticas recomendadas. Estos recursos se escalan automáticamente, se optimizan en costos y se actualizan periódicamente para ayudar a minimizar los costos operativos y los gastos generales.

Características

El modo automático de EKS ofrece las siguientes características de alto nivel:

Optimización de la administración de clústeres de Kubernetes: el modo automático de EKS optimiza la administración de EKS al proporcionar clústeres listos para la producción con una sobrecarga operativa mínima. Con el modo automático de EKS, puede ejecutar cargas de trabajo exigentes y dinámicas con confianza, sin necesidad de tener conocimientos profundos sobre EKS.

Disponibilidad de las aplicaciones: El modo automático de EKS agrega o elimina nodos dinámicamente en el clúster de EKS en función de la demanda de las aplicaciones de Kubernetes. Esto minimiza la necesidad de planificar manualmente la capacidad y garantiza la disponibilidad de las aplicaciones.

Eficiencia: el modo automático de EKS se diseñó para optimizar los costos de computación a la vez que se adhiere a la flexibilidad definida por el NodePool y los requisitos de la carga de trabajo. Además, termina las instancias que no se utilizan y consolida las cargas de trabajo en otros nodos con el fin de mejorar la rentabilidad.

Seguridad: el modo automático de EKS utiliza AMI que se tratan como inmutables para los nodos. Estas AMI aplican software bloqueado, habilitan controles de acceso obligatorios SELinux y proporcionan sistemas de archivos raíz de solo lectura. Además, los nodos lanzados por el modo automático de EKS tienen una vida útil máxima de 21 días (que se puede reducir), tras lo cual se sustituyen automáticamente por nuevos nodos. Este enfoque mejora la postura de seguridad mediante la rotación periódica de los nodos, en línea con las prácticas recomendadas ya adoptadas por un gran número de clientes.

Actualizaciones automatizadas: el modo automático de EKS mantiene actualizados el clúster de Kubernetes, los nodos y los componentes relacionados con las revisiones más recientes, a la vez que respeta los presupuestos de interrupción de pods (PDB) y los presupuestos de interrupción de NodePool (NDB) configurados. Es posible que sea necesario intervenir hasta los 21 días de vida útil máxima si los presupuestos de interrupción de pods de bloqueo u otras configuraciones impiden las actualizaciones.

Componentes administrados: el modo automático de EKS incluye Kubernetes y características en la nube de AWS como componentes principales que, de otro modo, tendrían que administrarse como complementos. Esto incluye soporte integrado para asignaciones de direcciones IP de pods, políticas de red de pods, servicios de DNS locales, complementos de GPU, comprobadores de estado y almacenamiento EBS CSI.

NodePools y NodeClasses personalizables: si la carga de trabajo requiere cambios en las configuraciones de almacenamiento, computación o redes, puede crear NodePools y NodeClasses personalizados mediante el modo automático de EKS. Si bien no se deben editar los NodePools y NodeClasses predeterminados, puede agregar nuevos NodePools o NodeClasses personalizados junto con las configuraciones predeterminadas para satisfacer requisitos específicos.

Componentes automatizados

El modo automático de EKS optimiza el funcionamiento de los clústeres de Amazon EKS mediante la automatización de los componentes clave de la infraestructura. La habilitación del modo automático de EKS reduce aún más las tareas de administración de los clústeres de EKS.

A continuación, se enumeran los componentes del plano de datos que están automatizados:

- Computación: en el caso de muchas cargas de trabajo, gracias al modo automático de EKS, se podrá desentender de muchos aspectos de la computación de los clústeres de EKS. Entre ellos se incluyen:
  - Nodos: los nodos del modo automático de EKS se diseñaron para ser tratados como aparatos. El modo automático de EKS se encarga de lo siguiente:
    - Elige una AMI adecuada que esté configurada con numerosos servicios necesarios para ejecutar las cargas de trabajo sin intervención.
    - Restringe el acceso a los archivos en la AMI mediante el modo de aplicación forzada de SELinux y un sistema de archivos raíz en modo solo lectura.
    - Impide el acceso directo a los nodos mediante la denegación de acceso SSH o SSM.
    - Incluye compatibilidad con GPU, a través de complementos y controladores de kernel independientes para GPU NVIDIA y Neuron, con lo que se consiguen cargas de trabajo de alto rendimiento.
    - Gestiona automáticamente los avisos de interrupción de las instancias de spot de EC2 y los eventos de estado de las instancias de EC2
  - Escalado automático: al basarse en el escalado automático de Karpenter, el modo automático de EKS supervisa los pods no programables y hace posible que se implementen nuevos nodos para ejecutar esos pods. A medida que se terminan las cargas de trabajo, el modo automático de EKS interrumpe y termina dinámicamente los nodos cuando dejan de ser necesarios, lo que optimiza el uso de los recursos.
  - Actualizaciones: al tener el control sobre los nodos, se agiliza la capacidad del modo automático de EKS a la hora de proporcionar revisiones de seguridad y actualizaciones tanto del sistema operativo como de los componentes, según sea necesario. Estas actualizaciones se diseñaron para minimizar las interrupciones de las cargas de trabajo. El modo automático de EKS impone una vida útil máxima del nodo de 21 días para garantizar que el software y las API se mantengan actualizados.
- Equilibrio de carga: el modo automático de EKS agiliza el equilibrio de carga mediante la integración con el servicio Elastic Load Balancing de Amazon, con lo que se automatiza el aprovisionamiento y la configuración de los equilibradores de carga para los servicios Kubernetes y los recursos de Ingress. Admite características avanzadas para los equilibradores de carga de aplicaciones y de red, administra su ciclo de vida y los escala para satisfacer las demandas de los clústeres. Esta integración aporta una solución de equilibrio de carga lista para producción que se adhiere a las prácticas recomendadas de AWS, con lo que es posible centrarse en las aplicaciones en lugar de en la administración de la infraestructura.

- Almacenamiento: el modo automático de EKS configura el almacenamiento efímero en su nombre y configura los tipos y tamaños de los volúmenes, las políticas de cifrado y las políticas de eliminación tras la terminación del nodo.
- Redes: el modo automático de EKS automatiza las tareas críticas de red para la conectividad de los pods y los servicios. Esto incluye la compatibilidad con IPv4/IPv6 y el uso de bloques de CIDR secundarios para ampliar los espacios de direcciones IP.
- Administración de identidades y accesos: no es necesario instalar el agente de Pod Identity de EKS en los clústeres del modo automático de EKS.

Para obtener más información sobre estas propiedades, consulte the section called “Funcionamiento”.

Configuración

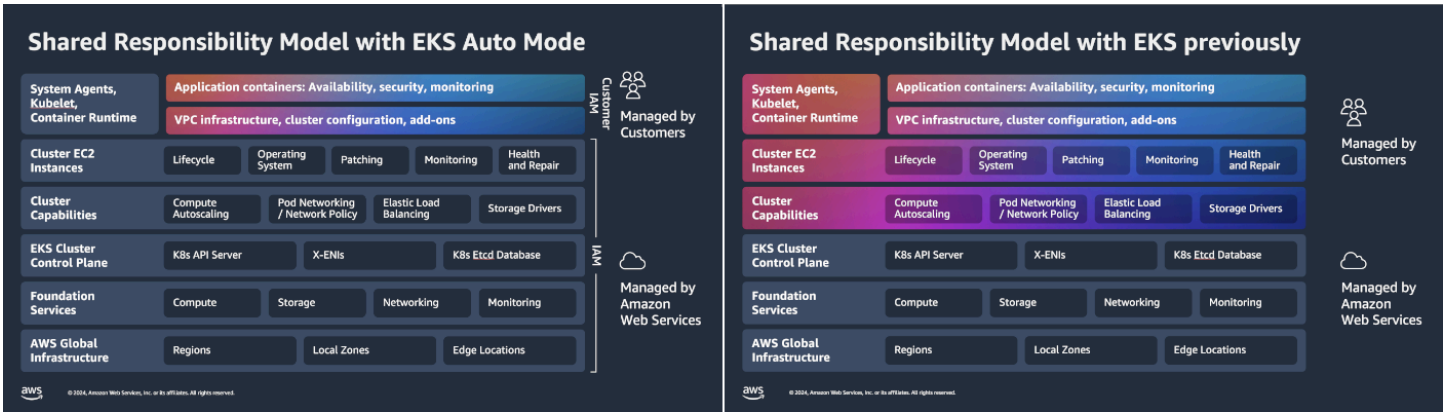
Aunque el modo automático de EKS administrará eficazmente la mayoría de los servicios del plano de datos sin necesidad de que intervenga, es posible que en ocasiones desee cambiar el modo en que se comportan algunos de esos servicios. Puede modificar la configuración de los clústeres del modo automático de EKS de las siguientes maneras:

- Daemonsets de Kubernetes: en lugar de modificar los servicios instalados en los nodos, puede utilizar DaemonSets de Kubernetes. Los Daemonsets se diseñaron para ser administrados por Kubernetes, pero se ejecutan en cada nodo del clúster. De este modo, puede agregar servicios especiales para supervisar o vigilar los nodos de otro modo.
- NodePools y NodeClasses personalizados: los NodePools y NodeClasses predeterminados son configurados por el modo automático de EKS y no se deben editar. Para personalizar el comportamiento de los nodos, se pueden crear NodePools o NodeClasses adicionales para casos de uso como:
  - Selección de tipos de instancia específicos (por ejemplo, procesadores acelerados o instancias de spot de EC2).
  - Aislar cargas de trabajo por motivos de seguridad o seguimiento de costos.
  - Configuración de ajustes de almacenamiento efímero, como IOPS, tamaño y rendimiento.
- Equilibrio de carga: algunos servicios, como el equilibrio de carga, que el modo automático de EKS ejecuta como objetos de Kubernetes, se pueden configurar directamente en los clústeres del modo automático de EKS.

Para obtener más información sobre las opciones de configuración del modo automático de EKS, consulte the section called “Configuración”.

Modelo de responsabilidad compartida

El modelo de responsabilidad compartida de AWS define las responsabilidades de seguridad y cumplimiento entre AWS y los clientes. Las imágenes y el texto que aparecen a continuación comparan y contrastan cómo difieren las responsabilidades del cliente y de AWS entre el modo automático de EKS y el modo estándar de EKS.



El modo automático de EKS transfiere gran parte de la responsabilidad compartida de la infraestructura de Kubernetes de los clientes a AWS. Con el modo automático de EKS, AWS asume una mayor responsabilidad por la seguridad de la nube, que antes era responsabilidad del cliente y ahora es compartida. Los clientes ahora pueden centrarse más en sus aplicaciones mientras AWS administra la infraestructura subyacente.

Responsabilidad del cliente

Con el modo automático de EKS, los clientes siguen siendo responsables de los contenedores de aplicaciones, incluida la disponibilidad, la seguridad y la supervisión. También mantienen el control sobre la infraestructura de VPC y la configuración del clúster de EKS. Este modelo permite a los clientes concentrarse en las preocupaciones específicas de la aplicación mientras delegan la administración de la infraestructura del clúster a AWS. Las características opcionales por nodo se pueden incluir en los clústeres mediante complementos de AWS.

Responsabilidad de AWS

Con el modo automático de EKS, AWS amplía su responsabilidad para incluir la administración de varios componentes críticos adicionales, en comparación con los que ya se administraban en los

clústeres de EKS que no utilizan el modo automático. En concreto, el modo automático de EKS se encarga de la configuración, la administración, la seguridad y el escalado de las instancias de EC2 lanzadas, así como de las capacidades de los clústeres para el equilibrio de carga, la administración de direcciones IP, la política de redes y el almacenamiento en bloque. AWS administra los siguientes componentes en el modo automático de EKS:

- Instancias de EC2 lanzadas en el modo automático: AWS gestiona el ciclo de vida completo de los nodos al aprovechar las instancias administradas por Amazon EC2. Las instancias administradas de EC2 asumen la responsabilidad de la configuración, la aplicación de parches, la supervisión y el mantenimiento del estado del sistema operativo. En este modelo, AWS es responsable tanto de la propia instancia como del sistema operativo huésped que se ejecuta en ella. Los nodos utilizan variantes de las AMI de Bottlerocket optimizadas para ejecutar contenedores. Las AMI de Bottlerocket tienen un software bloqueado, sistemas de archivos raíz inmutables y un acceso seguro a la red (para evitar las comunicaciones directas a través de SSH o SSM).
- Capacidades de clúster: AWS administra el escalado automático de cómputo, las redes de pods con la aplicación de políticas de red, la integración de Elastic Load Balancing y la configuración de los controladores de almacenamiento.
- Plano de control de los clústeres: AWS sigue administrando el servidor de la API de Kubernetes, los ENI multicuenta y la base de datos etcd, como ocurre con el EKS estándar.
- Servicios básicos e infraestructura global: AWS conserva la responsabilidad de los servicios subyacentes de computación, almacenamiento, redes y supervisión, así como de la infraestructura global de regiones, zonas locales y ubicaciones periféricas.

Cómo crear un clúster con el modo automático de Amazon EKS

En este capítulo se describe cómo crear un clúster de Amazon EKS con el modo automático habilitado mediante diversas herramientas e interfaces. El modo automático simplifica la creación de clústeres al configurar y administrar automáticamente la infraestructura de computación, redes y almacenamiento del clúster. Aprenderá a crear un clúster del modo automático mediante la AWS CLI, la Consola de administración de AWS o la herramienta de línea de comandos eksctl.

> **Note**
>
> El modo automático de EKS requiere la versión 1.29 o superior de Kubernetes.

Elija la herramienta que mejor se adapte a sus necesidades: la Consola de administración de AWS proporciona una interfaz visual ideal para explorar las características del modo automático de EKS y crear clústeres individuales. AWS CLI es ideal para tareas de creación de scripts y automatización, especialmente al integrar la creación de clústeres en flujos de trabajo existentes o en canalizaciones de CI/CD. La CLI de eksctl ofrece una experiencia nativa de Kubernetes y se recomienda para usuarios familiarizados con las herramientas de Kubernetes que deseen realizar operaciones simplificadas en la línea de comandos con configuraciones predeterminadas adecuadas.

Antes de comenzar, asegúrese de tener instalados y configurados los requisitos previos necesarios, incluidos los permisos de IAM correspondientes para crear clústeres de EKS. Para obtener información sobre cómo instalar herramientas de CLI, como `kubectlaws` y `eksctl`, consulte Configuración.

Puede usar la AWS CLI, la Consola de administración de AWS o la CLI de eksctl para crear un clúster con el modo automático de Amazon EKS.

Temas

- Creación de un clúster de modo automático de EKS con la CLI de eksctl
- Cómo crear un clúster de modo automático de EKS con AWS CLI
- Creación de un clúster del modo automático de EKS con la Consola de administración de AWS

Creación de un clúster de modo automático de EKS con la CLI de eksctl

En este tema se muestra cómo crear un clúster de modo automático de Amazon EKS mediante la interfaz de la línea de comandos (CLI) de eksctl. Puede crear un clúster de modo automático o bien mediante la ejecución de un único comando de la CLI o a través de la aplicación de un archivo de configuración YAML. Ambos métodos ofrecen la misma funcionalidad, aunque el enfoque YAML ofrece un control más detallado de la configuración del clúster.

La CLI de eksctl simplifica el proceso de creación y administración de clústeres de modo automático de EKS al encargarse de la creación y configuración de los recursos de AWS subyacentes. Antes de continuar, asegúrese de que cuenta con las credenciales y permisos de AWS necesarios configurados en la máquina local. En esta guía se presupone que conoce los conceptos básicos de Amazon EKS y que ya ha instalado las herramientas de la CLI necesarias.

> **Note**
>
> Debe instalar la versión `0.195.0` o superior de eksctl. Para obtener más información, consulte Versiones de eksctl en GitHub.

Creación de un clúster de modo automático de EKS con un comando de la CLI

Debe tener las herramientas `aws` y `eksctl` instaladas. Debe iniciar sesión en AWS CLI con permisos suficientes para administrar recursos de AWS, incluidos: instancias de EC2, redes de EC2, clústeres de EKS y roles de IAM. Para obtener más información, consulte Configuración.

Ejecute el siguiente comando para crear un nuevo clúster de modo automático de EKS con

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

Creación de un clúster de modo automático de EKS con un archivo YAML

Debe tener las herramientas `aws` y `eksctl` instaladas. Debe iniciar sesión en AWS CLI con permisos suficientes para administrar recursos de AWS, incluidos: instancias de EC2, redes de EC2, clústeres de EKS y roles de IAM. Para obtener más información, consulte Configuración.

Revise las opciones de configuración del modo automático de EKS en el recurso ClusterConfig de ejemplo que aparece a continuación. Para conocer la especificación completa de ClusterConfig, consulte la documentación de eksctl.

AWS sugiere habilitar el modo automático de EKS. Si es la primera vez que crea un clúster de modo automático de EKS, deje el `nodeRoleARN` sin especificar para crear un rol de IAM de nodo para el modo automático de EKS. Si ya tiene un rol de IAM de nodo en la cuenta de AWS, AWS sugiere que se vuelva a utilizar.

AWS sugiere no especificar ningún valor para `nodePools`. El modo automático de EKS creará grupos de nodos predeterminados. Puede utilizar la API de Kubernetes para crear grupos de nodos adicionales.

```
# 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
```

Guarde el archivo `ClusterConfig` como `cluster.yaml` y use el siguiente comando para crear el clúster:

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

Cómo crear un clúster de modo automático de EKS con AWS CLI

Los clústeres del modo automático de EKS automatizan las tareas rutinarias de administración de clústeres para computación, almacenamiento y redes. Por ejemplo, los clústeres del modo automático de EKS detectan automáticamente cuándo se necesitan nodos adicionales y aprovisionan nuevas instancias de EC2 a fin de satisfacer las demandas de la carga de trabajo.

En este tema se ofrece orientación para crear un nuevo clúster de modo automático de EKS mediante AWS CLI y, opcionalmente, implementar una carga de trabajo de muestra.

Requisitos previos

- La versión más reciente de la Interfaz de Línea de Comandos de AWS (AWS CLI) instalada y configurada en el dispositivo. Para comprobar su versión actual, utilice `aws --version`. Para

instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS.

  - Inicie sesión en la CLI con permisos de IAM suficientes para crear recursos de AWS, como políticas de IAM, roles de IAM y clústeres de EKS.
- La herramienta de línea de comandos kubectl instalada en el dispositivo. AWS sugiere que utilice la misma versión de kubectl que la versión de Kubernetes del clúster de EKS. Para instalar o actualizar kubectl, consulte the section called “Configure kubectl y eksctl”.

Cómo especificar las subredes de la VPC

El modo automático de Amazon EKS implementa nodos en las subredes de la VPC. Al crear un clúster de EKS, debe especificar las subredes de la VPC en las que se implementarán los nodos. Puede utilizar las subredes de la VPC predeterminadas en la cuenta de AWS o crear una VPC dedicada para cargas de trabajo críticas.

- AWS sugiere crear una VPC dedicada para el clúster. Información sobre cómo the section called “Creación de una VPC”.
- La consola de EKS ayuda a crear una nueva VPC. Información sobre cómo the section called “Consola de administración”.
- Como alternativa, puede utilizar la VPC predeterminada de la cuenta de AWS. Siga las siguientes instrucciones para encontrar los ID de subred.

Para encontrar los ID de subred de la VPC predeterminada

Uso de AWS CLI:

1. Ejecute el siguiente comando para enumerar la VPC predeterminada y sus subredes:

```
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. Guarde el resultado y anote los ID de subred.

   Código de salida de ejemplo:

```
----------------------------------------
|             DescribeSubnets          |
```

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

Roles de IAM para clústeres de modo automático de EKS

Rol de IAM de clúster

El modo automático de EKS requiere un rol de IAM de clúster para realizar acciones en la cuenta de AWS, como el aprovisionamiento de nuevas instancias de EC2. Debe crear este rol para conceder a EKS los permisos necesarios. AWS recomienda asociar las siguientes políticas administradas de AWS al rol de IAM del clúster:

- AmazonEKSComputePolicy
- AmazonEKSBlockStoragePolicy
- AmazonEKSLoadBalancingPolicy
- AmazonEKSNetworkingPolicy
- AmazonEKSClusterPolicy

Rol de IAM del nodo

Al crear un clúster de modo automático de EKS, se especifica un rol de IAM del nodo. Cuando el modo automático de EKS crea nodos para procesar cargas de trabajo pendientes, a cada nuevo nodo de instancia de EC2 se le asigna el rol de IAM del nodo. Este rol permite que el nodo se comunique con EKS, pero generalmente las cargas de trabajo que se ejecutan en el nodo no acceden a este.

Si desea conceder permisos a las cargas de trabajo que se ejecutan en un nodo, utilice EKS Pod Identity. Para obtener más información, consulte the section called “Pod Identity”.

Debe crear este rol y asociar la siguiente política administrada de AWS:

- AmazonEKSWorkerNodeMinimalPolicy
- AmazonEC2ContainerRegistryPullOnly

Rol vinculado a servicio

El modo automático de EKS también requiere un rol vinculado al servicio, que AWS crea y configura automáticamente. Para obtener más información, consulte AWSServiceRoleForAmazonEKS.

Cómo crear un rol de IAM de clúster de modo automático de EKS

Paso 1: Creación de la política de confianza

Cree una política de confianza que permita al servicio de Amazon EKS asumir el rol. Guarde la política como `trust-policy.json`:

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

Paso 2: Creación del rol de IAM

Utilice la política de confianza para crear el rol de IAM del clúster:

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

Paso 3: Cómo anotar el ARN del rol

Recupere y guarde el ARN del nuevo rol para utilizarlo en pasos posteriores:

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

Paso 4: Asociación de las políticas requeridas

Asocie las siguientes políticas administradas por AWS al rol de IAM del clúster para conceder los permisos necesarios:

AmazonEKSClusterPolicy:

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

AmazonEKSComputePolicy:

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

AmazonEKSBlockStoragePolicy:

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

AmazonEKSLoadBalancingPolicy:

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

AmazonEKSNetworkingPolicy:

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

Cómo crear un rol de IAM de nodo del modo automático de EKS

Paso 1: Creación de la política de confianza

Cree una política de confianza que permita al servicio de Amazon EKS asumir el rol. Guarde la política como `node-trust-policy.json`:

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

Paso 2: Creación del rol de IAM del nodo

Utilice el archivo node-trust-policy.json del paso anterior para definir qué entidades pueden asumir el rol. Ejecute el siguiente comando para crear el rol de IAM del nodo:

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

Paso 3: Cómo anotar el ARN del rol

Después de crear el rol, recupere y guarde el ARN del rol de IAM del nodo. Necesitará este ARN en los pasos siguientes. Utilice el siguiente comando para obtener el ARN:

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

Paso 4: Asociación de las políticas requeridas

Asocie las siguientes políticas administradas por AWS al rol de IAM del nodo para proporcionar los permisos necesarios:

AmazonEKSWorkerNodeMinimalPolicy:

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

AmazonEC2ContainerRegistryPullOnly:

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

Cómo crear un clúster de modo automático de EKS

Descripción general

Para crear un clúster de modo automático de EKS mediante AWS CLI, necesitará los siguientes parámetros:

- `cluster-name`: el nombre del clúster.
- `k8s-version`: la versión de Kubernetes (p. ej., 1.31).
- `subnet-ids`: los ID de subred identificados en los pasos anteriores.
- `cluster-role-arn`: ARN del rol de IAM del clúster.
- `node-role-arn`: ARN del rol de IAM del nodo.

Configuraciones de clúster predeterminadas

Revise las características y los valores predeterminados antes de crear el clúster:

- `nodePools`: el modo automático de EKS incluye grupos de nodos de uso general y predeterminados del sistema. Más información sobre los grupos de nodos.

Nota: Los grupos de nodos del modo automático de EKS se diferencian de los grupos de nodos administrados de Amazon EKS, pero pueden coexistir en el mismo clúster.

- `computeConfig.enabled`: automatiza las tareas de computación rutinarias, como la creación y eliminación de instancias de EC2.
- `kubernetesNetworkConfig.elasticLoadBalancing.enabled`: automatiza las tareas de equilibrio de carga, incluida la creación y eliminación de equilibradores de carga elásticos.
- `storageConfig.blockStorage.enabled`: automatiza las tareas de almacenamiento, como la creación y eliminación de volúmenes de Amazon EBS.
- `accessConfig.authenticationMode`: requiere entradas de acceso de EKS. Más información sobre los modos de autenticación de EKS.

Ejecute el comando

Utilice el siguiente comando para crear el clúster:

```
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\"
      }
    }
```

Cómo comprobar el estado del clúster

Paso 1: Comprobación de la creación del clúster

Ejecute el siguiente comando para comprobar el estado del clúster. La creación del clúster generalmente tarda unos 15 minutos:

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

Paso 2: Actualización de kubeconfig

Una vez que el clúster esté listo, actualice el archivo kubeconfig local para permitir que `kubectl` se comunique con el clúster. Esta configuración utiliza AWS CLI para la autenticación.

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

Paso 3: Verificación de los grupos de nodos

Enumere los grupos de nodos del clúster con el siguiente comando:

```
kubectl get nodepools
```

Siguientes pasos

- Aprenda a implementar una carga de trabajo de muestra en el nuevo clúster de modo automático de EKS.

Creación de un clúster del modo automático de EKS con la Consola de administración de AWS

La creación de un clúster del modo automático de EKS en la Consola de administración de AWS requiere menos configuración que otras opciones. EKS se integra con AWS IAM y las redes de VPC para ayudar a crear los recursos asociados a un clúster de EKS.

Existen dos opciones para crear un clúster en la consola:

- Configuración rápida (con el modo automático de EKS)
- Configuración personalizada

En este tema, aprenderá a crear un clúster de modo automático de EKS mediante la opción de configuración rápida.

Creación de un modo automático de EKS mediante la opción de configuración rápida

Debe iniciar sesión en la Consola de administración de AWS con permisos suficientes para administrar recursos de AWS, incluidos los siguientes: instancias de EC2, redes de EC2, clústeres de EKS y roles de IAM.

1. Vaya a la consola de EKS
2. Haga clic en Crear clúster
3. Confirme que la opción de configuración rápida esté seleccionada
4. Determine los siguientes valores o utilice los valores predeterminados para un clúster de prueba.
   - Nombre del clúster
   - Versión de Kubernetes
5. Seleccione el rol IAM del clúster. Si es la primera vez que crea un clúster de modo automático de EKS, utilice la opción Crear rol recomendado.
   - Opcionalmente, puede volver a utilizar un único rol de IAM de clúster en la cuenta de AWS para todos los clústeres de modo automático de EKS.
   - El rol de IAM de clúster incluye los permisos necesarios para que el modo automático de EKS administre los recursos, incluidas las instancias de EC2, los volúmenes de EBS y los equilibradores de carga de EC2.
   - La opción Crear rol recomendado completa previamente todos los campos con valores recomendados. Seleccione Siguiente y, a continuación, Crear. El rol usará el nombre de `AmazonEKSAutoClusterRole` sugerido.
   - Si ha creado recientemente un nuevo rol, utilice el icono Actualizar para recargar el menú desplegable de selección de roles.
6. Seleccione el rol de IAM del nodo. Si es la primera vez que crea un clúster de modo automático de EKS, utilice la opción Crear rol recomendado.
   - Opcionalmente, puede volver a utilizar un único rol de IAM de nodo en la cuenta de AWS para todos los clústeres de modo automático de EKS.
   - El rol de IAM de nodo incluye los permisos necesarios para que los nodos del modo automático se conecten al clúster. El rol de IAM del nodo debe incluir permisos para recuperar imágenes de ECR para los contenedores.
   - La opción Crear rol recomendado completa previamente todos los campos con valores recomendados. Seleccione Siguiente y, a continuación, Crear. El rol usará el nombre de `AmazonEKSAutoNodeRole` sugerido.

- Si ha creado recientemente un nuevo rol, utilice el icono Actualizar para recargar el menú desplegable de selección de roles.

7. Seleccione la VPC para el clúster de modo automático de EKS. Elija Crear VPC para crear una nueva VPC para EKS, o elija una VPC que haya creado previamente para EKS.
   - Si utiliza la Consola de la VPC para crear una nueva VPC, AWS sugiere crear al menos una puerta de enlace NAT por zona de disponibilidad. De lo contrario, puede utilizar todos los demás valores predeterminados.
   - Para obtener más información y detalles sobre los requisitos del clúster de IPv6, consulte the section called “Creación de una VPC”.
8. (opcional) El modo automático de EKS completa automáticamente las subredes privadas de la VPC seleccionada. Puede eliminar las subredes no deseadas.
   - EKS selecciona automáticamente las subredes privadas de la VPC según las prácticas recomendadas. Opcionalmente, puede seleccionar subredes adicionales de la VPC, como subredes públicas.
9. (opcional) Seleccione Ver valores predeterminados de configuración rápida para revisar todos los valores de configuración del nuevo clúster. En la tabla se indica que algunos valores no se pueden editar una vez creado el clúster.
10. Seleccione Create cluster (Crear clúster). Tenga en cuenta que la creación del clúster puede tardar quince minutos en completarse.

Siguientes pasos

- Aprenda a Implementar una carga de trabajo de muestra en el clúster de modo automático de EKS

Cómo habilitar el modo automático de EKS en clústeres de EKS existentes

Puede habilitar el modo automático de EKS en clústeres de EKS existentes.

AWS admite las siguientes migraciones:

- Migración de Karpenter a los nodos del modo automático de EKS. Para obtener más información, consulte the section called “Migre desde Karpenter”.

- Migración de los grupos de nodos administrados de EKS a los nodos del modo automático de EKS. Para obtener más información, consulte the section called “Migración desde MNG”.
- Migración de EKS Fargate al modo automático de EKS. Para obtener más información, consulte the section called “Migración de Fargate”.

AWS no admite las siguientes migraciones:

- Migración de volúmenes del controlador CSI de EBS (mediante el complemento de Amazon EKS) al controlador CSI de EBS del modo automático de EKS (administrado por el modo automático de EKS). Las PVC hechas con uno no se pueden montar sobre el otro, ya que utilizan dos aprovisionadores de volumen Kubernetes diferentes.
  - `eks-auto-mode-ebs-migration-tool` (proyecto AWS Labs) permite la migración entre la StorageClass estándar de CSI de EBS (`ebs.csi.aws.com`) y la StorageClass de CSI de EBS automática de EKS (`ebs.csi.eks.amazonaws.com`). Tenga en cuenta que la migración requiere eliminar y volver a crear los recursos existentes de PersistentVolumeClaim y PersistentVolume, por lo que la validación en un entorno que no sea de producción es esencial antes de la implementación.
- Migración de los equilibradores de carga del controlador de equilibrio de carga de AWS al modo automático de EKS

  Puede instalar el controlador del equilibrador de carga de AWS en un clúster del modo automático de Amazon EKS. Utilice las opciones `IngressClass` o `loadBalancerClass` para asociar los recursos de servicio e ingreso al controlador del equilibrador de carga o al modo automático de EKS.
- Migración de clústeres de EKS con CNI alternativas u otras configuraciones de red no compatibles

Referencia para las migraciones

Utilice la siguiente referencia para las migraciones para configurar los recursos de Kubernetes de modo que sean propiedad de controladores autoadministrados o del modo automático de EKS.

Funcionalidad	Recurso	Campo	Autoadministrado	Modo automático de EKS
Almacenamiento en bloque	`StorageClass`	`provisioner`	`ebs.csi.aws.com`	`ebs.csi.eks.amazonaws.com`
Equilibrio de carga	`Service`	`loadBalancerClass`	`service.k8s.aws/nlb`	`eks.amazonaws.com/nlb`
Equilibrio de carga	`IngressClass`	`controller`	`ingress.k8s.aws/alb`	`eks.amazonaws.com/alb`
Equilibrio de carga	`IngressClassParams`	`apiversion`	`elbv2.k8s.aws/v1beta1`	`eks.amazonaws.com/v1`
Equilibrio de carga	`TargetGroupBinding`	`apiversion`	`elbv2.k8s.aws/v1beta1`	`eks.amazonaws.com/v1`
Computación	`NodeClass`	`apiVersion`	`karpenter.sh/v1`	`eks.amazonaws.com/v1`

Migración de volúmenes de EBS

Al migrar las cargas de trabajo al modo automático de EKS, es necesario gestionar la migración de volúmenes de EBS debido a los diferentes aprovisionadores de controladores CSI:

- Aprovisionador de modo automático de EKS: `ebs.csi.eks.amazonaws.com`
- Aprovisionador de CSI de EBS de código abierto: `ebs.csi.aws.com`

Siga estos pasos para migrar los volúmenes persistentes:

1. Modificar la política de retención de volúmenes: cambie las versiones de la plataforma (PV) existentes `persistentVolumeReclaimPolicy` a `Retain` para garantizar que no se elimine el volumen de EBS subyacente.
2. Eliminar la PV de Kubernetes: elimine el recurso de PV anterior y mantenga intacto el volumen real de EBS.
3. Crear una nueva PV con aprovisionamiento estático: cree una nueva PV que haga referencia al mismo volumen de EBS, pero que funcione con el controlador CSI de destino.
4. Adjuntar a una PVC nueva: cree una nueva PVC que haga referencia específicamente a su PV utilizando el campo `volumeName`.

Consideraciones

- Asegúrese de detener las aplicaciones antes de comenzar la migración.
- Realice copias de seguridad de los datos antes de iniciar el proceso de migración.
- Este proceso debe realizarse para cada volumen persistente.
- La carga de trabajo debe actualizarse para usar la nueva PVC.

Migración de los equilibradores de carga

No se pueden transferir directamente los equilibradores de carga existentes desde el controlador del equilibrador de carga de AWS autoadministrado al modo automático de EKS. En su lugar, debe implementar una estrategia de implementación azul/verde. Esto implica mantener la configuración del equilibrador de carga existente y, al mismo tiempo, crear nuevos equilibradores de carga en el controlador administrado.

Para minimizar las interrupciones del servicio, recomendamos un enfoque de cambio de tráfico basado en el DNS. En primer lugar, cree nuevos equilibradores de carga mediante el modo automático de EKS y, al mismo tiempo, mantenga operativa la configuración actual. A continuación, utilice el enrutamiento de DNS (como Route 53) para desplazar gradualmente el tráfico de los equilibradores de carga antiguos a los nuevos. Una vez que el tráfico se haya migrado correctamente y haya verificado la nueva configuración, podrá retirar los equilibradores de carga y el controlador autoadministrado antiguos.

Cómo habilitar el modo automático de EKS en un clúster existente

En este tema se describe cómo habilitar el modo automático de Amazon EKS en los clústeres de Amazon EKS existentes. Para habilitar el modo automático en un clúster existente, es necesario actualizar los permisos de IAM y configurar los ajustes principales del modo automático de EKS. Una vez habilitado, podrá comenzar a migrar las cargas de trabajo de computación existentes para aprovechar las operaciones simplificadas y la administración automatizada de la infraestructura del modo automático.

Important

Compruebe que tiene instalada la versión mínima requerida de algunos complementos de Amazon EKS antes de habilitar el modo automático de EKS. Para obtener más información, consulte the section called “Versiones del complemento requeridas”.

Antes de comenzar, asegúrese de que dispone de acceso de administrador al clúster de Amazon EKS, así como de los permisos para modificar los roles de IAM. Los pasos descritos en este tema sirven de guía para habilitar el modo automático mediante la Consola de administración de AWS o la AWS CLI.

Consola de administración de AWS

Debe iniciar sesión en la consola de AWS con permiso para administrar los recursos de IAM, EKS y EC2.

Note

El rol de IAM de clúster de un clúster de EKS no se podrá modificar una vez creado el clúster. El modo automático de EKS requiere permisos adicionales para este rol. Debe asociar políticas adicionales al rol actual.

Actualización del rol de IAM de clúster

1. Abra la página de información general del clúster en la Consola de administración de AWS.
2. En el ARN del rol de IAM de clúster, seleccione Ver en IAM.
3. En la lista desplegable Agregar permisos, seleccione Asociar políticas.

4. Utilice el cuadro de búsqueda para encontrar y seleccionar las siguientes políticas:
   - `AmazonEKSComputePolicy`
   - `AmazonEKSBlockStoragePolicy`
   - `AmazonEKSLoadBalancingPolicy`
   - `AmazonEKSNetworkingPolicy`
   - `AmazonEKSClusterPolicy`
5. Seleccione Agregar permisos.
6. En la pestaña Relaciones de confianza, seleccione Editar política de confianza
7. Inserte la siguiente política de confianza del rol de IAM del clúster y seleccione Actualizar política

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

Cómo habilitar el modo automático de EKS

1. Abra la página de información general del clúster en la Consola de administración de AWS.
2. En Modo automático de EKS, seleccione Administrar
3. Active el Modo automático de EKS.
4. En el menú desplegable Grupo de nodos de EKS, seleccione los grupos de nodos predeterminados que desee crear.
   - Obtenga más información sobre los grupos de nodos en el modo automático de EKS. Para obtener más información, consulte the section called “Creación de un grupo de nodos”.

5. Si ha creado previamente un rol de IAM de nodo de modo automático de EKS en esta cuenta de AWS, selecciónelo en el menú desplegable Rol de IAM de nodo. Si no ha creado este rol anteriormente, seleccione Crear rol recomendado y siga los pasos.

AWS CLI

Requisitos previos

- El rol de clúster de IAM del clúster de EKS existente debe incluir permisos suficientes para el modo automático de EKS, como las siguientes políticas:
  - `AmazonEKSComputePolicy`
  - `AmazonEKSBlockStoragePolicy`
  - `AmazonEKSLoadBalancingPolicy`
  - `AmazonEKSNetworkingPolicy`
  - `AmazonEKSClusterPolicy`
- El rol de IAM de clúster debe tener una política de confianza actualizada que incluya la acción `sts:TagSession`. Para obtener más información sobre cómo crear un rol de IAM de clúster, consulte the section called “ AWS CLI”.
- `aws` CLI instalada, sesión iniciada y una versión suficiente. Debe tener permiso para administrar recursos de IAM, EKS y EC2. Para obtener más información, consulte Configuración.

Procedimiento

Utilice los siguientes comandos para habilitar el modo automático de EKS en un clúster existente.

> **Note**
>
> Las capacidades de computación, almacenamiento en bloque y equilibrio de carga se deben habilitar o desactivar en la misma solicitud.

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

Versiones del complemento requeridas

Si desea habilitar el modo automático de EKS en un clúster existente, es posible que deba actualizar algunos complementos. Tenga en cuenta:

- Esto se aplica únicamente a los clústeres existentes que realicen la transición al modo automático de EKS.
- Los nuevos clústeres creados con el modo automático de EKS habilitado no necesitan estas actualizaciones.

Si tiene alguno de los siguientes complementos instalados, asegúrese de que tienen la versión mínima especificada:

Nombre del complemento	Versión mínima requerida
Complemento CNI de Amazon VPC para 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
Controlador CSI de Amazon EBS	v1.37.0-eksbuild.1
Controlador de instantáneas CSI	v8.1.0-eksbuild.2
Agente de Pod Identity de EKS	v1.3.4-eksbuild.1

Para obtener más información, consulte the section called “Cómo actualizar un complemento”.

Siguientes pasos

- Para migrar las cargas de trabajo de Administración de grupo de nodos, consulte the section called “Migración desde MNG”.

- Para migrar desde Karpenter autoadministrado, consulte the section called “Migre desde Karpenter”.

Migración desde Karpenter al modo automático de EKS mediante kubectl

En este tema se explica el proceso para migrar cargas de trabajo de Karpenter al modo automático de Amazon EKS mediante kubectl. La migración se puede realizar de forma gradual, lo que le permite trasladar las cargas de trabajo a su propio ritmo y, al mismo tiempo, mantener la estabilidad del clúster y la disponibilidad de las aplicaciones durante la transición.

El enfoque paso a paso que se describe a continuación permite ejecutar Karpenter y el modo automático de EKS paralelamente durante el periodo de migración. Esta estrategia de doble operación ayuda a garantizar una transición fluida, ya que permite validar el comportamiento de la carga de trabajo en el modo automático de EKS antes de desmantelar completamente Karpenter. Puede migrar aplicaciones individualmente o en grupos, lo que proporciona flexibilidad para acomodar los requisitos operativos específicos y el nivel de tolerancia al riesgo.

Requisitos previos

Antes de iniciar la migración, asegúrese de que dispone de:

- Karpenter v1.1 o posterior instalado en el clúster. Para obtener más información, consulte Upgrading to 1.1.0+ en la documentación de Karpenter.
- `kubectl` instalado y conectado al clúster. Para obtener más información, consulte Configuración.

En este tema se presupone que ya está familiarizado con Karpenter y NodePools. Para obtener más información, consulte la documentación de Karpenter.

Paso 1: Habilitación del modo automático de EKS en el clúster

Habilite el modo automático de EKS en el clúster existente mediante AWS CLI o la Consola de administración. Para obtener más información, consulte the section called “Habilitación en el clúster”.

> **Note**
>
> Al habilitar el modo automático de EKS, no habilite el nodepool de `general purpose` en esta etapa de la transición. Este grupo de nodos no es selectivo.

Para obtener más información, consulte the section called “Revisión de los grupos de nodos integrados”.

Paso 2: Creación de un NodePool del modo automático de EKS con taints aplicadas

Cree un nuevo NodePool para el modo automático de EKS con una taint. Con esto se garantiza que los pods existentes no se programarán automáticamente en los nuevos nodos del modo automático de EKS. Este grupo de nodos utiliza la `NodeClass default` integrada en el modo automático de EKS. Para obtener más información, consulte the section called “Cómo crear una clase de nodos”.

Ejemplo de grupo de nodos con taint aplicada:

```
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"
```

Actualice los requisitos del grupo de nodos de modo que coincidan con la configuración de Karpenter a partir de la cual se va a migrar. Necesita al menos un requisito.

Paso 3: Actualización de las cargas de trabajo para la migración

Identifique y actualice las cargas de trabajo que desea migrar al modo automático de EKS. Agregue tanto las tolerancias como los selectores de nodos a estas cargas de trabajo:

```
apiVersion: apps/v1
kind: Deployment
```

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

Este cambio permite programar la carga de trabajo en los nuevos nodos del modo automático de EKS.

El modo automático de EKS utiliza etiquetas diferentes a las de Karpenter. Las etiquetas relacionadas con las instancias administradas por EC2 comienzan con `eks.amazonaws.com`. Para obtener más información, consulte the section called “Creación de un grupo de nodos”.

Paso 4: Migración de las cargas de trabajo de forma gradual

Repita el paso 3 para cada carga de trabajo que desee migrar. Esto permite trasladar las cargas de trabajo de forma individual o en grupos, según los requisitos y la tolerancia al riesgo.

Paso 5: Eliminación del NodePool de Karpenter original

Una vez que se hayan migrado todas las cargas de trabajo, podrá eliminar el NodePool de Karpenter original:

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

Paso 6: Eliminación de la taint del NodePool del modo automático de EKS (opcional)

Si desea que el modo automático de EKS se convierta en el modo predeterminado para las nuevas cargas de trabajo, puede eliminar la taint del NodePool del modo automático de EKS:

```
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
```

Paso 7: Eliminación de los selectores de nodos de las cargas de trabajo (opcional)

Si ha eliminado la taint del NodePool del modo automático de EKS, puede optar por eliminar los selectores de nodos de las cargas de trabajo, ya que el modo automático de EKS es ahora la opción predeterminada:

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

Paso 8: Desinstalación de Karpenter del clúster

Los pasos que se deben seguir para eliminar Karpenter dependen de cómo se haya instalado. Para obtener más información, consulte las Instrucciones de instalación de Karpenter.

Migración de grupos de nodos administrados de EKS al modo automático de EKS

Al llevar a cabo la transición del clúster de Amazon EKS para usar el modo automático de EKS, las cargas de trabajo existentes se pueden migrar sin problemas desde los grupos de nodos administrados (MNG) mediante la herramienta de la CLI de eksctl. Este proceso garantiza la disponibilidad continua de las aplicaciones, mientras que el modo automático de EKS optimiza los recursos de computación. La migración se puede llevar a cabo con una interrupción mínima de las aplicaciones en ejecución.

En este tema, se explican los pasos que se deben seguir para vaciar de forma segura los pods de los grupos de nodos administrados existentes y permitir que el modo automático de EKS los reprograme en las instancias recién aprovisionadas. Si sigue este procedimiento, podrá aprovechar la consolidación inteligente de la carga de trabajo del modo automático de EKS y, al mismo tiempo, mantener la disponibilidad de la aplicación durante la migración.

Requisitos previos

- Clúster de Amazon EKS con el modo automático de EKS habilitado
- `eksctl` CLI instalada y conectada al clúster. Para obtener más información, consulte Configuración.
- Karpenter no debe estar instalado en el clúster.

Procedimiento

Utilice el siguiente comando de `eksctl` CLI para iniciar el vaciado de los pods de las instancias de grupos de nodos administrados existentes. El modo automático de EKS creará nuevos nodos para respaldar los pods desplazados.

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

Deberá ejecutar este comando para cada grupo de nodos administrado del clúster.

Para obtener información adicional sobre este comando, consulte la sección Eliminación y drenaje de grupos de nodos en la documentación de eksctl.

Migración de EKS Fargate a los nodos del modo automático de EKS

En esta sección, se explica el proceso para migrar cargas de trabajo de EKS Fargate al modo automático de Amazon EKS mediante `kubectl`. La migración se puede realizar de forma gradual, lo que le permite trasladar las cargas de trabajo a su propio ritmo y, al mismo tiempo, mantener la estabilidad del clúster y la disponibilidad de las aplicaciones durante la transición.

El enfoque paso a paso que se describe a continuación permite ejecutar EKS Fargate y el modo automático de EKS paralelamente durante el periodo de migración. Esta estrategia de doble operación ayuda a garantizar una transición fluida, ya que permite validar el comportamiento de la carga de trabajo en el modo automático de EKS antes de desmantelar completamente EKS Fargate. Puede migrar aplicaciones individualmente o en grupos, lo que proporciona flexibilidad para acomodar los requisitos operativos específicos y el nivel de tolerancia al riesgo.

Comparación entre el modo automático de Amazon EKS y EKS con AWS Fargate

Amazon EKS con AWS Fargate sigue siendo una opción para los clientes que desean utilizar EKS, pero el modo automático de Amazon EKS es el enfoque recomendado en el futuro. El

modo automático de EKS es totalmente compatible con Kubernetes y con todas las primitivas y herramientas de plataforma originales de Kubernetes, como Istio, que Fargate no admite. El modo automático de EKS también es totalmente compatible con todas las opciones de compra en tiempo de ejecución de EC2, incluidas las instancias de GPU y de spot, lo que permite a los clientes aprovechar los descuentos negociados de EC2 y otros mecanismos de ahorro. Estas funciones no están disponibles cuando se utiliza EKS con Fargate.

Además, el modo automático de EKS permite a los clientes lograr el mismo modelo de aislamiento que Fargate, utilizando las capacidades de programación estándar de Kubernetes para garantizar que cada instancia de EC2 ejecute un único contenedor de aplicaciones. Al adoptar el modo automático de Amazon EKS, los clientes pueden aprovechar todas las ventajas de ejecutar Kubernetes en AWS, una plataforma totalmente compatible con Kubernetes que ofrece la flexibilidad necesaria para aprovechar toda la gama de EC2 y las opciones de compra, al tiempo que conserva la facilidad de uso y la abstracción de la administración de infraestructuras que ofrece Fargate.

Requisitos previos

Antes de iniciar la migración, asegúrese de que ha hecho lo siguiente:

- Configurar un clúster con Fargate. Para obtener más información, consulte the section called “Introducción”.
- Instalar y conectar `kubectl` al clúster. Para obtener más información, consulte Configuración.

Paso 1: Revisión del clúster de Fargate

1. Revise si el clúster de EKS con Fargate está en ejecución:

```
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. Revise los pods en ejecución:

```
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. Cree una implementación en un archivo llamado `deployment_fargate.yaml`:

```
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. Aplique la implementación:

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

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

5. Revise los pods y las implementaciones:

```
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. Revise el nodo:

```
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
```

Paso 2: Habilitación del modo automático de EKS en el clúster

1. Habilite el modo automático de EKS en el clúster existente mediante AWS CLI o la Consola de administración. Para obtener más información, consulte the section called “Habilitación en el clúster”.
2. Revise el nodepool:

```
kubectl get nodepool
```

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

Paso 3: Actualización de las cargas de trabajo para la migración

Identifique y actualice las cargas de trabajo que desea migrar al modo automático de EKS.

Para migrar una carga de trabajo de Fargate al modo automático de EKS, aplique la anotación `eks.amazonaws.com/compute-type: ec2`. Esto garantiza que Fargate no programará la carga de trabajo, a pesar del perfil de Fargate, y que lo hará el NodePool del modo automático de EKS. Para obtener más información, consulte the section called “Creación de un grupo de nodos”.

1. Modifique las implementaciones (por ejemplo, el archivo `deployment_fargate.yaml`) para cambiar el tipo de procesamiento a `ec2`:

```
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. Aplique la implementación. Este cambio permite programar la carga de trabajo en los nuevos nodos del modo automático de EKS:

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

3. Verifique que la implementación se esté ejecutando en el clúster del modo automático de EKS:

```
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. Verifique que no haya ningún nodo Fargate en ejecución ni ninguna implementación en ejecución en los nodos administrados del modo automático de EKS:

```
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
```

Paso 4: Migración de las cargas de trabajo de forma gradual

Repita el paso 3 para cada carga de trabajo que desee migrar. Esto permite trasladar las cargas de trabajo de forma individual o en grupos, según los requisitos y la tolerancia al riesgo.

Paso 5: Eliminación del perfil original de Fargate

Una vez que se hayan migrado todas las cargas de trabajo, podrá eliminar el perfil original de `fargate`. Reemplace *<nombre del perfil de fargate>* por el nombre de su perfil de Fargate:

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

Paso 6: Reducción vertical de CoreDNS

Como el modo automático de EKS gestiona CoreDNS, puede reducir la implementación de `coredns` a 0:

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

Ejecución de cargas de trabajo de muestra en clústeres del modo automático de EKS

En este capítulo se ofrecen ejemplos de cómo implementar diferentes tipos de cargas de trabajo en clústeres de Amazon EKS que se ejecutan en modo automático. Los ejemplos muestran patrones clave de cargas de trabajo, como aplicaciones de muestra, aplicaciones web con equilibrio de carga, cargas de trabajo con estado que utilizan almacenamiento persistente y cargas de trabajo con requisitos específicos de ubicación de nodos. Cada ejemplo incluye manifiestos completos e instrucciones de implementación paso a paso que se pueden utilizar como plantillas para aplicaciones propias.

Antes de continuar con los ejemplos, asegúrese de que tiene un clúster de EKS en ejecución en modo automático y que ha instalado AWS CLI y kubectl. Para obtener más información, consulte Configuración. Los ejemplos presuponen cierta familiaridad con los conceptos de Kubernetes y los comandos kubectl.

Puede utilizar estas muestras basadas en casos de uso para ejecutar cargas de trabajo en clústeres del modo automático de EKS.

the section called “Implemente una carga de trabajo de expansión”

Muestra cómo implementar una carga de trabajo de muestra en un clúster de modo automático de EKS mediante comandos `kubectl`.

the section called “Implementación del equilibrador de carga”

Muestra cómo implementar una versión en contenedores del videojuego 2048 en Amazon EKS.

the section called “Implementación de una carga de trabajo con estado”

Muestra cómo implementar una aplicación con estado de muestra en un clúster del modo automático de EKS.

the section called “Implementación de carga de trabajo acelerada”

Muestra cómo implementar cargas de trabajo aceleradas por hardware en nodos administrados por el modo automático de EKS.

the section called “Control de la implementación”

Muestra cómo utilizar una anotación para controlar si una carga de trabajo se implementa en nodos administrados por el modo automático de EKS.

Implementación de una carga de trabajo de expansión de muestra en un clúster del modo automático de Amazon EKS

En este tutorial, aprenderá a implementar una carga de trabajo de muestra en un clúster del modo automático de EKS y observará cómo se aprovisionan automáticamente los recursos de computación necesarios. Utilizará comandos `kubectl` para observar el comportamiento del clúster y comprobar de primera mano la forma en que el modo automático simplifica las operaciones de Kubernetes en AWS. Al final de este tutorial, comprenderá la forma en que el modo automático de EKS responde a las implementaciones de cargas de trabajo mediante la administración automática de los recursos de computación subyacentes, sin necesidad de configurar manualmente los grupos de nodos.

Requisitos previos

- Un clúster del modo automático de Amazon EKS. Anote el nombre y la región de AWS del clúster.
- Una entidad principal de IAM, como un usuario o rol, con permisos suficientes para administrar recursos de red, computación y EKS.
  - Para obtener más información, consulte Cómo crear roles y asociar políticas en la Guía del usuario de IAM.
- `aws` CLI instalada y configurada con una identidad de IAM.
- `kubectl` CLI instalada y conectada al clúster.
  - Para obtener más información, consulte Configuración.

Paso 1: Revisión de los recursos de computación existentes (opcional)

En primer lugar, utilice `kubectl` para enumerar los grupos de nodos en el clúster.

```
kubectl get nodepools
```

Resultado de ejemplo:

```
general-purpose
```

En este tutorial, implementaremos una carga de trabajo configurada para utilizar el grupo de nodos `general-purpose`. Este grupo de nodos está integrado en el modo automático de EKS e incluye valores predeterminados razonables para cargas de trabajo generales, como microservicios y aplicaciones web. Puede crear un grupo de nodos propio. Para obtener más información, consulte the section called “Creación de un grupo de nodos”.

En segundo lugar, utilice `kubectl` para enumerar los nodos conectados al clúster.

```
kubectl get nodes
```

Si acaba de crear un clúster de modo automático de EKS, no tendrá nodos.

En este tutorial, implementará una carga de trabajo de ejemplo. Si no tiene nodos, o la carga de trabajo no cabe en los nodos existentes, el modo automático de EKS aprovisionará un nuevo nodo.

Paso 2: Implementación de una aplicación de ejemplo en el clúster

Revise la siguiente implementación de Kubernetes y guárdela como `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
```

Tenga en cuenta que el selector `eks.amazonaws.com/compute-type: auto` requiere que la carga de trabajo se implemente en un nodo de modo automático de Amazon EKS.

Aplique la implementación al clúster.

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

Paso 3: Visualización de los eventos de Kubernetes

Utilice el siguiente comando para ver los eventos de Kubernetes, incluida la creación de un nuevo nodo. Utilice `ctrl+c` para dejar de ver los eventos.

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

Utilice `kubectl` para enumerar los nodos conectados al clúster nuevamente. Tenga en cuenta el nodo recién creado.

```
kubectl get nodes
```

Paso 4: Visualización de nodos e instancias en la consola de AWS

Puede ver los nodos de modo automático de EKS en la consola de EKS, y las instancias de EC2 asociadas en la consola de EC2.

Las instancias de EC2 implementadas por el modo automático de EKS están restringidas. No se pueden ejecutar comandos arbitrarios en los nodos del modo automático de EKS.

Paso 5: Eliminación de la implementación

Utilice `kubectl` para eliminar la implementación de muestra

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

Si no tiene otras cargas de trabajo implementadas en el clúster, el nodo creado por el modo automático de EKS estará vacío.

En la configuración predeterminada, el modo automático de EKS detecta los nodos que han estado vacíos durante treinta segundos y los termina.

Utilice `kubectl` o la consola de EC2 para confirmar que la instancia asociada se ha eliminado.

Implementación de una carga de trabajo de equilibrador de carga de muestra en el modo automático de EKS

En esta guía se explica cómo implementar una versión en contenedores del videojuego 2048 en Amazon EKS, con equilibrio de carga y accesibilidad a Internet.

Requisitos previos

- Un clúster de modo automático de EKS
- `kubectl` configurado para interactuar con el clúster
- Permisos de IAM adecuados para crear recursos del equilibrador de carga de aplicaciones

Paso 1: Creación del espacio de nombres

En primer lugar, cree un espacio de nombres dedicado para la aplicación del videojuego 2048.

Cree un archivo denominado `01-namespace.yaml`:

```
apiVersion: v1
kind: Namespace
metadata:
  name: game-2048
```

Aplique la configuración del espacio de nombres:

```
kubectl apply -f 01-namespace.yaml
```

Paso 2: Implementación de la aplicación

La aplicación ejecuta varias réplicas del contenedor del videojuego 2048.

Cree un archivo denominado `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**
>
> Si obtiene un error al cargar la imagen `public.ecr.aws/l6m2t8p7/docker-2048:latest`, confirme que el rol de IAM del nodo tenga los permisos suficientes para extraer imágenes de ECR. Para obtener más información, consulte the section called "Rol de IAM de nodo". Además, la imagen `docker-2048` del ejemplo es una imagen `x86_64` y no se ejecutará en otras arquitecturas.

Componentes principales:

- Implementa 5 réplicas de la aplicación
- Utiliza una imagen de ECR pública
- Solicita 0,5 núcleos de CPU por pod

- Expone el puerto 80 para el tráfico HTTP

Aplique la implementación:

```
kubectl apply -f 02-deployment.yaml
```

Paso 3: Creación del servicio

El servicio expone la implementación a la red del clúster.

Cree un archivo denominado `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
```

Componentes principales:

- Crea un servicio NodePort
- Asigna el puerto 80 al puerto 80 del contenedor
- Utiliza el selector de etiquetas para encontrar pods

Aplique el servicio:

```
kubectl apply -f 03-service.yaml
```

Paso 4: Configuración del equilibrio de carga

Configurará un ingreso para exponer la aplicación a Internet.

Primero, cree la `IngressClass`. Cree un archivo denominado `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**
>
> El modo automático de EKS requiere etiquetas de subred para identificar las subredes públicas y privadas.
> Si creó el clúster con `eksctl`, ya dispone de estas etiquetas.
> Aprenda cómo the section called “Etiquetado de subredes”.

A continuación, cree el recurso de ingreso. Cree un archivo denominado `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
```

Componentes principales:

- Creación de un equilibrador de carga de aplicaciones expuesto a Internet
- Utiliza el tipo de destino de IP para el enrutamiento directo de pods
- Dirige todo el tráfico (/) al servicio del videojuego

Aplique las configuraciones de ingreso:

```
kubectl apply -f 04-ingressclass.yaml
kubectl apply -f 05-ingress.yaml
```

Paso 5: Verificación de la implementación

1. Compruebe que todos los pods estén en ejecución:

   ```
   kubectl get pods -n game-2048
   ```

2. Compruebe que el servicio se haya creado:

   ```
   kubectl get svc -n game-2048
   ```

3. Obtenga el punto de conexión del equilibrador de carga de aplicaciones:

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

El campo ADDRESS en la salida del ingreso mostrará el punto de conexión del equilibrador de carga de aplicaciones. Espere de 2 a 3 minutos para que el equilibrador de carga de aplicaciones aprovisione y registre todos los destinos.

Paso 6: Acceso al videojuego

Abra el navegador web y vaya a la URL del punto de conexión del equilibrador de carga de aplicaciones del paso anterior. Aparecerá la interfaz del videojuego 2048.

Paso 7: Efectúe una limpieza

Para eliminar todos los recursos creados en este tutorial:

```
kubectl delete namespace game-2048
```

Esto eliminará todos los recursos del espacio de nombres, incluidos los recursos de implementación, servicio e ingreso.

Lo que ocurre detrás de las cámaras

1. La implementación crea 5 pods que ejecutan el videojuego 2048
2. El servicio proporciona un acceso de red estable a estos pods
3. Modo automático de EKS:
   - Crea un equilibrador de carga de aplicación en AWS
   - Configura grupos de destino para los pods
   - Establece reglas de enrutamiento para dirigir el tráfico al servicio

Solución de problemas

Si el videojuego no se carga

- Asegúrese de que todos los pods se ejecutan: `kubectl get pods -n game-2048`
- Compruebe el estado del ingreso: `kubectl describe ingress -n game-2048`
- Verifique las comprobaciones de estado del equilibrador de carga de aplicación: compruebe el estado del grupo de destino en la Consola de AWS

Implementación de una carga de trabajo con estado de ejemplo en el modo automático de EKS

En este tutorial se explica cómo implementar una aplicación con estado de ejemplo en el clúster de modo automático de EKS. La aplicación escribe marcas de tiempo en un volumen persistente, lo que demuestra las capacidades automáticas de persistencia y aprovisionamiento de volúmenes de EBS del modo automático de EKS.

Requisitos previos

- Un clúster de modo automático de EKS
- AWS CLI configurada con los permisos adecuados

- `kubectl` instalado y configurado
  - Para obtener más información, consulte Configuración.

Paso 1: Configuración del entorno

1. Configure las variables de entorno:

```
export CLUSTER_NAME=my-auto-cluster
export AWS_REGION="us-west-2"
```

2. Actualice la kubeconfig:

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

Paso 2: Creación de la clase de almacenamiento

La `StorageClass` define cómo el modo automático de EKS aprovisionará los volúmenes de EBS.

El modo automático de EKS no crea una `StorageClass` en su nombre. Debe crear una `StorageClass` que haga referencia a `ebs.csi.eks.amazonaws.com` para utilizar la capacidad de almacenamiento del modo automático de EKS.

1. Cree un archivo denominado `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. Aplicación de `StorageClass`:

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

Componentes principales:

- `provisioner: ebs.csi.eks.amazonaws.com`: utiliza el modo automático de EKS
- `volumeBindingMode: WaitForFirstConsumer`: retrasa la creación del volumen hasta que un pod lo necesite
- `type: gp3`: especifica el tipo de volumen de EBS
- `encrypted: "true"`: EBS utilizará la clave de `aws/ebs` predeterminada para cifrar los volúmenes creados con esta clase. Esto es opcional, pero recomendable.
- `storageclass.kubernetes.io/is-default-class: "true"`: Kubernetes utilizará esta clase de almacenamiento de forma predeterminada, a menos que se especifique una clase de volumen diferente en una reclamación de volumen persistente. Tenga cuidado al establecer este valor si va a migrar desde otro controlador de almacenamiento. (opcional)

Paso 3: Creación de la reclamación de volumen persistente

La reclamación de volumen persistente solicita almacenamiento a la `StorageClass`.

1. Cree un archivo denominado `pvc.yaml`:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: auto-ebs-claim
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: auto-ebs-sc
  resources:
    requests:
      storage: 8Gi
```

2. Aplicación de la reclamación de volumen persistente:

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

Componentes principales:

- `accessModes: ReadWriteOnce`: el volumen se puede montar con un nodo a la vez

- `storage: 8Gi`: solicita un volumen de 8 GiB
- `storageClassName: auto-ebs-sc`: hace referencia a la `StorageClass` que creamos

Paso 4: Implementación de la aplicación

La implementación ejecuta un contenedor que escribe marcas de tiempo en el volumen persistente.

1. Cree un archivo denominado `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. Aplique la implementación:

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

Componentes principales:

- Contenedor bash simple que escribe marcas de tiempo en un archivo
- Monta la reclamación de volumen persistente en `/data`
- Solicita 1 núcleo de CPU
- Utiliza el selector de nodos para los nodos administrados por EKS

Paso 5: Verificación del modo en que quedó establecido

1. Compruebe que el pod se ejecuta:

```
kubectl get pods -l app=inflate-stateful
```

2. Compruebe que la reclamación de volumen persistente está vinculada:

```
kubectl get pvc auto-ebs-claim
```

3. Compruebe el volumen de EBS:

```
# 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. Verifique que los datos se escriben:

```
kubectl exec "$(kubectl get pods -l app=inflate-stateful \
  -o=jsonpath='{.items[0].metadata.name}')" -- \
  cat /data/out.txt
```

Paso 6: Efectúe una limpieza

Ejecute el siguiente comando para eliminar todos los recursos creados en este tutorial:

```
# Delete all resources in one command
kubectl delete deployment/inflate-stateful pvc/auto-ebs-claim storageclass/auto-ebs-sc
```

Lo que ocurre detrás de las cámaras

1. La reclamación de volumen persistente solicita almacenamiento a la `StorageClass`.
2. Cuando el pod se ha programado:
   a. El modo automático de EKS aprovisiona un volumen de EBS
   b. Crea un PersistentVolume
   c. Asocia el volumen al nodo
3. El pod monta el volumen y comienza a escribir las marcas de tiempo

Controlador de instantáneas

El modo automático de EKS es compatible con Kubernetes CSI Snapshotter, también conocido como controlador de instantáneas. Sin embargo, el modo automático de EKS no incluye el controlador de instantáneas. Usted es responsable de instalar y configurar el controlador de instantáneas. Para obtener más información, consulte the section called “Controlador de instantáneas CSI”.

Revise la siguiente `VolumeSnapshotClass` que hace referencia a la capacidad de almacenamiento del modo automático de EKS.

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: auto-ebs-vsclass
driver: ebs.csi.eks.amazonaws.com
deletionPolicy: Delete
```

Más información sobre Kubernetes CSI Snapshotter.

Implementación de una carga de trabajo acelerada

En este tutorial, se muestra cómo el modo automático de Amazon EKS simplifica el lanzamiento de cargas de trabajo aceleradas por hardware. Con el modo automático, Amazon EKS simplifica las operaciones más allá del propio clúster al automatizar componentes clave de infraestructura, lo que permite contar desde el primer momento con funciones de computación, redes, equilibrio de carga, almacenamiento y administración de identidades y accesos.

El modo automático de Amazon EKS incluye los controladores y los complementos de dispositivo necesarios para determinados tipos de instancias, como los controladores NVIDIA y AWS Neuron. No tiene que instalar ni actualizar estos componentes.

El modo automático de EKS administra automáticamente los controladores de estos aceleradores:

- AWS Trainium
- AWS Inferentia
- GPU NVIDIA en instancias aceleradas de Amazon EC2

> **Note**
>
> El modo automático de EKS incluye el complemento de dispositivo de NVIDIA para Kubernetes. Este complemento se ejecuta automáticamente y no aparece como un conjunto de daemon en el clúster.

Compatibilidad de red adicional:

- Elastic Fabric Adapter (EFA)

Con el modo automático, Amazon EKS elimina el trabajo manual y repetitivo de administrar el controlador del acelerador y el complemento del dispositivo.

También puede ahorrar costos al reducir la capacidad del clúster a cero. Puede habilitar el modo automático de EKS para que termine las instancias cuando no haya cargas de trabajo en ejecución. Esto es especialmente útil para cargas de trabajo de inferencia basadas en lotes.

A continuación, se muestra un ejemplo de cómo lanzar cargas de trabajo aceleradas con el modo automático de Amazon EKS.

Requisitos previos

- Un clúster de Kubernetes con el modo automático de Amazon EKS configurado.
- Una clase de nodo `default` de EKS, tal como se crea al habilitar los grupos de nodos administrados de `general-purpose` o del `system`.

Paso 1: Implementación de una carga de trabajo de GPU

Este ejemplo muestra cómo crear un NodePool para cargas de trabajo basadas en NVIDIA que requieren 45 GB de memoria de GPU. Con el modo automático de EKS, puede usar las restricciones de programación de Kubernetes para especificar los requisitos de instancia.

Para implementar el `NodePool` de Amazon EKS en modo automático junto con la `workload` de ejemplo, revise las siguientes definiciones de NodePool y Pod, y guárdelas como `nodepool-gpu.yaml` y `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
```

Tenga en cuenta que el selector `eks.amazonaws.com/compute-type: auto` requiere que la carga de trabajo se implemente en un nodo de modo automático de Amazon EKS. El NodePool también aplica un taint que solo permite la programación de pods con tolerancias para GPU de NVIDIA.

Aplique el NodePool y la carga de trabajo al clúster.

```
kubectl apply -f nodepool-gpu.yaml
kubectl apply -f pod.yaml
```

Debería ver los siguientes datos de salida:

```
nodepool.karpenter.sh/gpu configured created
pod/nvidia-smi created
```

Espere unos segundos y verifique los nodos del clúster. Ahora debería poder ver un nuevo nodo aprovisionado en el clúster del modo automático de Amazon EKS:

```
> kubectl get nodes

NAME        TYPE           CAPACITY    ZONE         NODE                   READY   AGE
gpu-dnknr   g6e.2xlarge    on-demand   us-west-2b   i-02315c7d7643cdee6   True    76s
```

Paso 2: Validación

Puede observar que el modo automático de Amazon EKS lanzó una instancia `g6e.2xlarge` en lugar de una `g6.2xlarge`, ya que la carga de trabajo requería una instancia con GPU l40s, conforme a las siguientes restricciones de programación de Kubernetes:

```
...
  nodeSelector:
    eks.amazonaws.com/instance-gpu-name: l40s
...
    requests:
        memory: "30Gi"
        cpu: "3500m"
        nvidia.com/gpu: 1
      limits:
        memory: "30Gi"
        nvidia.com/gpu: 1
```

A continuación, consulte los registros de los contenedores con el siguiente comando:

```
kubectl logs nvidia-smi
```

Código de salida de ejemplo:

```
+-----------------------------------------------------------------------------------------
+
| 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
 |
+---------------------------------------------------------------------------------------
+
```

Puede ver que el contenedor detectó que se ejecuta en una instancia con GPU `NVIDIA` y que no necesitó instalar ningún controlador, ya que el modo automático de Amazon EKS se encarga de ello.

Paso 3: Limpieza

Para deshacerse de todos los objetos creados, utilice `kubectl` para eliminar la implementación de muestra y NodePool para que se termine el nodo:

```
kubectl delete -f nodepool-gpu.yaml
kubectl delete -f pod.yaml
```

Ejemplo de referencia de NodePools

Creación de un NodePool de NVIDIA

El siguiente NodePool define:

- Lance únicamente instancias de las familias g6e y g6
- Consolide los nodos cuando estén vacíos durante 1 hora
  - El valor de 1 hora en `consolodateAfter` está diseñado para admitir cargas de trabajo con picos y reducir la rotación de nodos. Puede ajustar `consolidateAfter` en función de los requisitos de la carga de trabajo.

Ejemplo de NodePool con una familia de instancias con GPU y consolidación

```
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
```

En lugar de configurar el `eks.amazonaws.com/instance-gpu-name`, puede utilizar `eks.amazonaws.com/instance-family` para especificar la familia de instancias. Para ver otras etiquetas conocidas que influyen en la revisión de la programación, consulte the section called "Etiquetas compatibles con el modo automático de EKS".

Cuando existan requisitos específicos de almacenamiento, es posible ajustar los `iops`, el `size` y el `throughput` del almacenamiento efímero de los nodos mediante la creación de una NodeClass propia, la cual puede ser referenciada desde el NodePool. Más información sobre las opciones configurables de NodeClass.

Ejemplo de configuración de almacenamiento de NodeClass

```
apiVersion: eks.amazonaws.com/v1
kind: NodeClass
metadata:
  name: gpu
spec:
  ephemeralStorage:
    iops: 3000
    size: 80Gi
    throughput: 125
```

Definición de un NodePool de AWS Trainium y AWS Inferentia

El siguiente NodePool cuenta con un conjunto de `eks.amazonaws.com/instance-category` que indica: lanzar únicamente instancias de las familias Inferentia y Trainium.

```
        - key: "eks.amazonaws.com/instance-category"
          operator: In
          values:
            - inf
            - trn
```

Configuración de los ajustes del modo automático de EKS

En este capítulo se describe cómo configurar aspectos específicos de los clústeres del modo automático de Amazon Elastic Kubernetes Service (EKS). Aunque el modo automático de EKS

administra la mayoría de los componentes de la infraestructura de forma automática, puede personalizar ciertas características para satisfacer los requisitos de la carga de trabajo.

Con las opciones de configuración que se describen en este tema, se pueden modificar los ajustes de red, los recursos de computación y los comportamientos de equilibrio de carga, a la vez que se aprovechan las ventajas que ofrece la administración automatizada de la infraestructura. Antes de realizar cualquier cambio en la configuración, revise las opciones disponibles en las siguientes secciones para determinar qué enfoque se adapta mejor a sus necesidades.

¿Qué características desea configurar?	Opción de configuración
Redes y almacenamiento de nodos • Configuración de la ubicación de nodos en subredes públicas y privadas • Definición de grupos de seguridad personalizados para el control de acceso a los nodos • Personalización de las políticas de traducción de direcciones de red (SNAT) • Activación de las políticas de red de Kubernetes, el registro y la supervisión detallados de las políticas de red • Ajuste de los parámetros de almacenamiento efímero (tamaño, IOPS, rendimiento) • Configuración del almacenamiento efímero cifrado con claves de KMS personalizadas • Aislamiento del tráfico de pods en subredes separadas de los nodos	the section called “Cómo crear una clase de nodos”
Recursos de computación de nodos • Selección de tipos y familias de instancias de EC2 específicos • Definición de arquitecturas de CPU (x86_64, ARM64)	the section called “Creación de un grupo de nodos”

¿Qué características desea configurar?	Opción de configuración
• Configuración de los tipos de capacidad (bajo demanda, spot) • Especificación de las zonas de disponibilidad • Configuración de taints y etiquetas de nodos • Establecimiento de límites de recursos para el uso de la CPU y la memoria	
Grupos de nodos con capacidad estática • Mantenimiento de un número fijo de nodos independientemente de la demanda de los pods • Configuración de una capacidad predecible para las instancias reservadas • Establecimiento de recuentos de réplicas específicos para una huella de la infraestr uctura coherente • Escalado manual de grupos de nodos estáticos • Supervisión del estado de grupos de nodos con capacidad estática	the section called “Grupos de nodos con capacidad estática”

¿Qué características desea configurar?	Opción de configuración
Configuración del equilibrador de carga de aplicación • Implementación de equilibradores de carga internos o orientados a Internet • Configuración del equilibrio de carga entre zonas • Ajuste de los periodos de inactividad • Habilitación de la compatibilidad con HTTP/2 y WebSocket • Configuración de los parámetros de comprobación de estado • Especificación de la configuración del certificado TLS • Definición de los atributos del grupo de destino • Ajuste del tipo de dirección IP (IPv4, doble pila)	the section called “Creación de una clase de ingreso”

¿Qué características desea configurar?	Opción de configuración
Configuración del equilibrador de carga de red • Configuración del direccionamiento IP directo del pod • Habilitación del equilibrio de carga entre zonas • Ajuste del tiempo de inactividad de la conexión • Configuración de los parámetros de comprobación de estado • Especificación de la ubicación de la subred • Ajuste del tipo de dirección IP (IPv4, doble pila) • Configuración de la preservación de la IP de origen del cliente • Definición de los atributos del grupo de destino	the section called “Crear un servicio”
Configuración de la clase de almacenamiento • Definición de los tipos de volúmenes de EBS (gp3, io1, io2, etc.) • Configuración del cifrado de volúmenes y uso de claves de KMS • Ajuste de los parámetros de IOPS y rendimiento • Establecimiento como clase de almacenamiento predeterminada • Definición de etiquetas personalizadas para volúmenes aprovisionados	the section called “Creación de una StorageClass”

¿Qué características desea configurar?	Opción de configuración
Control del uso de ODCR • Ajuste de la implementación de la carga de trabajo en las reservas de capacidad bajo demanda de EC2 • Selección explícita de ODCR específicos por ID para el uso de capacidad dirigido • Uso de la selección basada en etiquetas en grupos de destino de ODCR • Filtro de ODCR por cuenta de propietario de AWS para escenarios entre cuentas • Control para verificar si las cargas de trabajo utilizan automáticamente los ODCR abiertos	the section called “Control de las reservas de capacidad”

Cómo crear una clase de nodos para Amazon EKS

Las clases de nodos de Amazon EKS son plantillas que ofrecen un control detallado de la configuración de los nodos administrados del modo automático de EKS. Una clase de nodos define los ajustes a nivel de infraestructura que se aplican a los grupos de nodos del clúster de EKS, incluida la configuración de la red, los ajustes de almacenamiento y el etiquetado de los recursos. En este tema se explica cómo crear y configurar una clase de nodos para cumplir con requisitos operativos específicos.

Si necesita personalizar la forma en que el modo automático de EKS aprovisiona y configura las instancias de EC2 más allá de los ajustes predeterminados, la creación de una clase de nodos permite controlar con precisión los parámetros críticos de la infraestructura. Por ejemplo, puede especificar la ubicación de la subred privada para mejorar la seguridad, configurar el almacenamiento efímero de las instancias para cargas de trabajo sensibles al rendimiento o aplicar un etiquetado personalizado para asignar los costos.

Cómo crear una clase de nodos

Para crear una `NodeClass`, siga estos pasos:

1. Cree un archivo YAML (por ejemplo, `nodeclass.yaml`) con la configuración de la clase de nodo

2. Aplique la configuración al clúster mediante `kubectl`
3. Haga referencia a la clase de nodos en la configuración del grupo de nodos. Para obtener más información, consulte the section called “Creación de un grupo de nodos”.

Debe tener `kubectl` instalado y configurado. Para obtener más información, consulte Configuración.

Ejemplo de clase de nodos básica

A continuación, se muestra un ejemplo de clase de nodos:

```
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"
```

Esta NodeClass aumenta la cantidad de almacenamiento efímero en el nodo.

Aplique esta configuración mediante:

```
kubectl apply -f nodeclass.yaml
```

A continuación, haga referencia a la clase de nodos en la configuración del grupo de nodos. Para obtener más información, consulte the section called “Creación de un grupo de nodos”.

Creación de una entrada de acceso a una clase de nodos

Si crea una clase de nodos personalizada, debe crear una entrada de acceso de EKS para permitir que los nodos se unan al clúster. EKS crea entradas de acceso automáticamente cuando se utilizan la clase de nodos y los grupos de nodos integrados.

Para obtener información sobre cómo funcionan las entradas de acceso, consulte the section called "Entradas de los registros".

Al crear entradas de acceso para las clases de nodos del modo automático de EKS, debe utilizar el tipo de entrada de acceso `EC2`.

Creación de una entrada de acceso con la CLI

Para crear una entrada de acceso para los nodos de EC2 y asociar la política de nodos automáticos de EKS:

Actualice los siguientes comandos de la CLI con el nombre del clúster y el ARN del rol del nodo. El ARN del rol del nodo se especifica en la clase de nodo YAML.

```
# 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
```

Creación de una entrada de acceso con CloudFormation

Para crear una entrada de acceso para los nodos de EC2 y asociar la política de nodos automáticos de EKS:

Actualice el siguiente ejemplo de CloudFormation con el nombre de su clúster y el ARN del rol del nodo. El ARN del rol del nodo se especifica en la clase de nodo YAML.

```
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
```

Para obtener información sobre la implementación de pilas de CloudFormation, consulte Introducción a CloudFormation.

Especificación de clase de nodos

```
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

  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
```

Consideraciones

- Si desea verificar cuánto almacenamiento local tiene una instancia, puede describir el nodo para ver el recurso de almacenamiento efímero.

- Cifrado de volumen: EKS utiliza la clave de KMS personalizada configurada para cifrar el volumen raíz de solo lectura de la instancia y el volumen de datos de lectura/escritura.
- Sustitución del rol de IAM del nodo: si cambia el rol de IAM del nodo asociado a una `NodeClass`, tendrá que crear una nueva entrada de acceso. EKS crea automáticamente una entrada de acceso para el rol de IAM del nodo durante la creación del clúster. El rol de IAM del nodo requiere la política de acceso de EKS `AmazonEKSAutoNodePolicy`. Para obtener más información, consulte the section called “Entradas de los registros”.
- Densidad máxima de pods: EKS limita la cantidad máxima de pods en un nodo a 110. Este límite se aplica después del cálculo de la cantidad máxima de pods existente. Para obtener más información, consulte the section called “Tipos de instancias de Amazon EC2”.
- Etiquetas: si desea propagar etiquetas de Kubernetes a EC2, debe configurar permisos de IAM adicionales. Para obtener más información, consulte the section called “Identidad y acceso”.
- Clase de nodo predeterminada: no asigne el nombre `default` a su clase de nodo personalizada. Esto se debe a que el modo automático de EKS incluye una `NodeClass` llamada `default` que se aprovisiona automáticamente cuando se habilita al menos una `NodePool` integrada. Para obtener más información sobre la activación de `NodePools` integrados, consulte the section called “Revisión de los grupos de nodos integrados”.
- Comportamiento de **`subnetSelectorTerms`** con varias subredes: si hay varias subredes que coinciden con las condiciones de `subnetSelectorTerms` o que proporciona por ID, el modo automático de EKS crea nodos distribuidos entre las subredes.
  - Si las subredes se encuentran en diferentes zonas de disponibilidad (AZ), puede usar las características de Kubernetes, como las restricciones de propagación de topología de pods y el enrutamiento consciente de la topología, para distribuir los pods y el tráfico entre las zonas, respectivamente.
  - Si hay varias subredes en la misma AZ que coincidan con `subnetSelectorTerms`, el modo automático de EKS crea pods en cada nodo distribuidos en las subredes de esa AZ. El modo automático de EKS crea interfaces de red secundarias en cada nodo de las demás subredes de la misma AZ. Elige en función del número de direcciones IP disponibles en cada subred, para utilizar las subredes de manera más eficiente. Sin embargo, no puede especificar qué subred usa el modo automático de EKS para cada pod; si necesita que los pods se ejecuten en subredes específicas, utilice the section called “Selección de subredes para los pods” en su lugar.

Selección de subredes para los pods

Los campos `podSubnetSelectorTerms` y `podSecurityGroupSelectorTerms` permiten efectuar configuraciones de red avanzadas, ya que permiten que los pods se ejecuten en subredes diferentes a las de sus nodos. Esta separación proporciona un mayor control sobre el enrutamiento del tráfico de la red y las políticas de seguridad. Tenga en cuenta que `podSecurityGroupSelectorTerms` son obligatorios con `podSubnetSelectorTerms`.

Casos de uso

Use `podSubnetSelectorTerms` cuando necesite:

- Separar el tráfico de infraestructura (comunicación de nodo a nodo) del tráfico de aplicaciones (comunicación de pod a pod).
- Aplicar configuraciones de red diferentes a las subredes de nodos que a las subredes de pods.
- Implementar diferentes políticas de seguridad o reglas de enrutamiento para los nodos y los pods.
- Configurar proxies inversos o filtrado de red específicamente para el tráfico de nodos sin afectar al tráfico de los pods. Utilizar `advancedNetworking` y `certificateBundles` para definir su proxy inverso y cualquier certificado autofirmado o privado para el proxy.

Configuración de ejemplo

```
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"
```

Consideraciones sobre los selectores de subred para los pods

- Densidad de pods reducida: se pueden ejecutar menos pods en cada nodo cuando se utiliza `podSubnetSelectorTerms`, ya que la interfaz de red principal del nodo se encuentra en la subred del nodo y no se puede usar para los pods de la subred de pods.
- Limitaciones del selector de subredes: las configuraciones `subnetSelectorTerms` y `securityGroupSelectorTerms` estándar no se aplican a la selección de subredes de los pods.
- Planificación de la red: garantice un espacio de direcciones IP adecuado en las subredes de nodos y pods para satisfacer sus requisitos de carga de trabajo.
- Configuración de enrutamiento: compruebe que la tabla de enrutamiento y la lista de control de acceso (ACL) de red de las subredes de pods estén configuradas correctamente para la comunicación entre las subredes de nodos y pods.
- Zonas de disponibilidad: verifique que haya creado subredes de pods en varias AZ. Si está utilizando una subred de pod específica, debe estar en la misma AZ que la subred del nodo.

Creación de un grupo de nodos para el modo automático de EKS

Los grupos de nodos de Amazon EKS ofrecen una forma flexible de administrar los recursos de computación del clúster de Kubernetes. En este tema se muestra cómo crear y configurar grupos de nodos mediante Karpenter, una herramienta de aprovisionamiento de nodos que ayuda a optimizar el escalado del clúster y el uso de los recursos. Con el recurso NodePool de Karpenter, puede definir requisitos específicos para los recursos de computación, como los tipos de instancias, las zonas de disponibilidad, las arquitecturas y los tipos de capacidad.

No puede modificar los grupos de nodos `system` y `general-purpose` integrados. No puede habilitarlos ni deshabilitarlos. Para obtener más información, consulte the section called “Revisión de los grupos de nodos integrados”.

La especificación NodePool permite un control detallado de los recursos de computación del clúster de EKS mediante diversas etiquetas y requisitos compatibles. Estas incluyen opciones para especificar las categorías de instancias de EC2, los ajustes de la CPU, las zonas de disponibilidad, las arquitecturas (ARM64 o AMD64) y los tipos de capacidad (spot o bajo demanda). También puede establecer límites de recursos para el uso de la CPU y la memoria, lo que garantiza que el clúster se mantenga dentro de los límites operativos necesarios.

El modo automático de EKS utiliza las conocidas etiquetas de Kubernetes para ofrecer formas uniformes y estandarizadas de identificar las características de los nodos. Estas etiquetas, como `topology.kubernetes.io/zone` de las zonas de disponibilidad y `kubernetes.io/arch` de la arquitectura de la CPU, siguen las convenciones establecidas de Kubernetes. Además, las etiquetas específicas de EKS (con el prefijo `eks.amazonaws.com/`) amplían esta funcionalidad con atributos específicos de AWS, como los tipos de instancias, los fabricantes de CPU, las capacidades de la GPU y las especificaciones de red. Este sistema de etiquetado estandarizado permite una integración perfecta con herramientas de Kubernetes existentes y, al mismo tiempo, proporciona una profunda integración de la infraestructura de AWS.

Cómo crear un NodePool

Siga estos pasos para crear un NodePool para el clúster de Amazon EKS:

1. Cree un archivo YAML llamado `nodepool.yaml` con la configuración de NodePool necesaria. Puede usar la siguiente configuración de muestra.
2. Aplique el NodePool al clúster:

```
kubectl apply -f nodepool.yaml
```

3. Compruebe que el NodePool se haya creado correctamente:

```
kubectl get nodepools
```

4. (Opcional) Supervise el estado del NodePool:

```
kubectl describe nodepool default
```

Asegúrese de que el NodePool haga referencia a una NodeClass válida que exista en el clúster. La NodeClass define configuraciones específicas de AWS para los recursos de computación. Para obtener más información, consulte the section called “Cómo crear una clase de nodos”.

NodePool de muestra

```
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
```

Etiquetas compatibles con el modo automático de EKS

El modo automático de EKS es compatible con las siguientes etiquetas conocidas.

> **Note**
>
> El modo automático de EKS utiliza etiquetas diferentes a las de Karpenter. Las etiquetas relacionadas con las instancias administradas por EC2 comienzan con `eks.amazonaws.com`.

Etiqueta	Ejemplo	Descripción
topology.kubernetes.io/zone	us-east-2a	Región AWS
node.kubernetes.io/instance-type	g4dn.8xlarge	Tipo de instancia AWS
kubernetes.io/arch	amd64	Las arquitecturas se definen mediante valores GOARCH de la instancia
karpenter.sh/capacity-type	spot	Los tipos de capacidad incluyen `spot` y `on-demand`
eks.amazonaws.com/instance-hypervisor	nitro	Tipos de instancias que usan un hipervisor específico
eks.amazonaws.com/compute-type	auto	Identifica los nodos administrados del modo automático de EKS
eks.amazonaws.com/instance-encryption-in-transit-supported	true	Tipos de instancias que admiten (o no) el cifrado en tránsito
eks.amazonaws.com/instance-category	g	Tipos de instancias de la misma categoría, generalmente la cadena antes del número de generación.
eks.amazonaws.com/instance-generation	4	Número de generación del tipo de instancia dentro de una categoría de instancias
eks.amazonaws.com/instance-family	g4dn	Tipos de instancias de propiedades similares, pero con diferentes cantidades de recursos

Etiqueta	Ejemplo	Descripción
eks.amazonaws.com/instance-size	8xlarge	Tipos de instancias con cantidades de recursos similares, pero con propiedades diferentes
eks.amazonaws.com/instance-cpu	32	Cantidad de CPU en la instancia
eks.amazonaws.com/instance-cpu-manufacturer	aws	Nombre del fabricante de la CPU
eks.amazonaws.com/instance-memory	131072	Cantidad de mebibytes de memoria de la instancia
eks.amazonaws.com/instance-ebs-bandwidth	9500	Cantidad máxima de megabits de EBS disponibles en la instancia
eks.amazonaws.com/instance-network-bandwidth	131072	Cantidad de megabits de referencia disponibles en la instancia
eks.amazonaws.com/instance-gpu-name	t4	Nombre de la GPU de la instancia, si está disponible
eks.amazonaws.com/instance-gpu-manufacturer	nvidia	Nombre del fabricante de la GPU
eks.amazonaws.com/instance-gpu-count	1	Cantidad de GPU en la instancia
eks.amazonaws.com/instance-gpu-memory	16384	Cantidad de mebibytes de memoria de la GPU
eks.amazonaws.com/instance-local-nvme	900	Cantidad de gibibytes de almacenamiento nvme local en la instancia

> **Note**
>
> El modo automático de EKS solo es compatible con determinadas instancias y tiene requisitos de tamaño mínimo. Para obtener más información, consulte the section called “Referencia de instancias compatibles con el modo automático de EKS”.

Etiquetas no compatibles con el modo automático de EKS

El modo automático de EKS no es compatible con las siguientes marcas.

- El modo automático de EKS solo es compatible con Linux
  - `node.kubernetes.io/windows-build`
  - `kubernetes.io/os`

Desactivación de los grupos de nodos integrados

Si crea grupos de nodos personalizados, puede deshabilitar los grupos de nodos integrados. Para obtener más información, consulte the section called “Revisión de los grupos de nodos integrados”.

Clúster sin grupos de nodos integrados

Puede crear un clúster sin los grupos de nodos integrados. Esta opción resulta útil cuando la organización ha creado grupos de nodos personalizados.

> **Note**
>
> Al crear un clúster sin grupos de nodos integrados, la NodeClass `default` no se aprovisiona automáticamente. Deberá crear una NodeClass personalizada. Para obtener más información, consulte the section called “Cómo crear una clase de nodos”.

Información general:

1. Cree un clúster de EKS con los valores `nodePools` y `nodeRoleArn` vacíos.
   - Ejemplo de `autoModeConfig` de eksctl:

```
autoModeConfig:
```

```
  enabled: true
  nodePools: []
  # Do not set a nodeRoleARN
```

Para obtener más información, consulte the section called “CLI de eksctl”

2. Creación de una clase de nodos personalizada con un ARN de rol del nodo
   - Para obtener más información, consulte the section called “Cómo crear una clase de nodos”
3. Creación de una entrada de acceso para la clase de nodos personalizada
   - Para obtener más información, consulte the section called “Creación de una entrada de acceso a una clase de nodos”
4. Creación de un grupo de nodos personalizado, como se ha descrito anteriormente.

Interrupción

Puede configurar el modo automático de EKS para interrumpir los nodos que atraviesan su NodePool de varias formas. Puede utilizar `spec.disruption.consolidationPolicy`, `spec.disruption.consolidateAfter` o `spec.template.spec.expireAfter`. También puede limitar la frecuencia de las interrupciones del modo automático de EKS a través de `spec.disruption.budgets` del NodePool. También puede controlar los intervalos de tiempo y el número de nodos simultáneos interrumpidos. Para obtener instrucciones sobre cómo configurar este comportamiento, consulte Disruption en la documentación de Karpenter.

Puede configurar la interrupción de los grupos de nodos para:

- Identificar cuándo están infrautilizadas las instancias y consolidar las cargas de trabajo.
- Crear un presupuesto de interrupciones del grupo de nodos para limitar las terminaciones de nodos a causa de desviaciones, vacíos y consolidaciones.

De forma predeterminada, el modo automático de EKS:

- Consolida las instancias infrautilizadas.
- Termina las instancias después de 336 horas.
- Establece un presupuesto único de interrupciones del 10 % de los nodos.
- Permite reemplazar los nodos debido a la desviación cuando se lanza una nueva AMI en modo automático, lo que ocurre aproximadamente una vez por semana.

Periodo de gracia por terminación

Cuando no se define explícitamente un `terminationGracePeriod` en el NodePool del modo automático de EKS, el sistema aplica automáticamente un periodo de gracia por terminación predeterminado de 24 horas al NodeClaim asociado. Si bien los clientes del modo automático de EKS no verán un `terminationGracePeriod` predeterminado en sus configuraciones personalizadas de NodePool, sí lo podrán ver en el NodeClaim. La funcionalidad permanece constante tanto si el periodo de gracia se establece explícitamente en el NodePool como si se establece de forma predeterminada en el NodeClaim, lo que garantiza un comportamiento predecible de terminación de nodos en todo el clúster.

Grupos de nodos con capacidad estática en el modo automático de EKS

El modo automático de Amazon EKS admite grupos de nodos con capacidad estática que mantienen un número fijo de nodos independientemente de la demanda de los pods. Los grupos de nodos con capacidad estática son útiles para las cargas de trabajo que requieren una capacidad predecible, instancias reservadas o requisitos de cumplimiento específicos cuando es necesario mantener una huella de la infraestructura coherente.

A diferencia de los grupos de nodos dinámicos que escalan en función de las exigencias de programación de los pods, los grupos de nodos con capacidad estática mantienen la cantidad de nodos que ha configurado.

Ejemplo básico

Este es un grupo de nodos simple con capacidad estática que mantiene 5 nodos:

```
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
```

Configuración de un grupo de nodos con capacidad estática

Para crear un grupo de nodos con capacidad estática, defina el campo `replicas` en la especificación de NodePool. El campo `replicas` define el número exacto de nodos que conservará el grupo de nodos.

Restricciones de los grupos de nodos con capacidad estática

Los grupos de nodos con capacidad estática tienen varias restricciones y comportamientos importantes:

Restricciones de configuración:

- No se puede cambiar de modo: una vez que haya configurado `replicas` en un grupo de nodos, no puede eliminarlo. El grupo de nodos no puede cambiar entre los modos estático y dinámico.
- Límites de recursos limitados: solo se admite el campo `limits.nodes` en la sección de límites. Los límites de CPU y memoria no son aplicables.
- Campo sin peso: el campo `weight` no se puede configurar en grupos de nodos con capacidad estática, ya que la selección de nodos no se basa en la prioridad.

Comportamiento operativo:

- Sin consolidación: los nodos de los grupos con capacidad estática no se tienen en cuenta para la consolidación en función del uso.
- Operaciones de escalado: las operaciones de escalado eluden los presupuestos de interrupción de nodos, pero siguen respetando los PodDisruptionBudgets.
- Sustitución de nodos: los nodos se siguen sustituyendo por desviación (como las actualizaciones de la AMI) y vencimiento en función de la configuración.

Escalado de un grupo de nodos con capacidad estática

Puede cambiar el número de réplicas de un grupo de nodos con capacidad estática mediante el comando `kubectl scale`:

```
# Scale down to 5 nodes
kubectl scale nodepool static-nodepool --replicas=5
```

Al reducir verticalmente, el modo automático de EKS terminará los nodos correctamente: respetará los PodDisruptionBudgets y permitirá reprogramar los pods en ejecución para los nodos restantes.

Supervisión de grupos de nodos con capacidad estática

Utilice los siguientes comandos para supervisar los grupos de nodos con capacidad estática:

```
# 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}'
```

El campo `status.nodes` muestra el número actual de nodos administrados por el grupo de nodos, que debería coincidir con el recuento de `replicas` deseado en condiciones normales.

Configuraciones de ejemplo

Grupo de nodos básico con capacidad estática

```
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
```

Capacidad estática con tipos de instancias específicos

```
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%
```

Grupo de nodos con capacidad estática multizona

```
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%
```

Prácticas recomendadas

Planificación de la capacidad:

- Establezca un valor de `limits.nodes` superior a `replicas` para permitir el escalado temporal durante las operaciones de sustitución de nodos.
- Tenga en cuenta la capacidad máxima necesaria durante la desviación de nodos o las actualizaciones de la AMI al establecer los límites.

Selección de instancias:

- Utilice tipos de instancias específicos cuando tenga instancias reservadas o requisitos de hardware específicos.
- Evite los requisitos demasiado restrictivos que puedan limitar la disponibilidad de las instancias durante el escalado.

Administración de interrupciones:

- Configure los presupuestos de interrupciones adecuados para equilibrar la disponibilidad con las operaciones de mantenimiento.
- Tenga en cuenta la tolerancia de la aplicación con respecto a la sustitución de nodos al establecer los porcentajes del presupuesto.

Supervisión de:

- Supervise el campo `status.nodes` periódicamente para garantizar que se mantenga la capacidad deseada.
- Configure alertas para cuando el recuento real de nodos se desvíe de las réplicas deseadas.

Distribución en zonas:

- Distribuya la capacidad estática en varias zonas de disponibilidad para una mayor disponibilidad.
- Al crear un grupo de nodos con capacidad estática que abarca varias zonas de disponibilidad, el modo automático de EKS distribuye los nodos entre las zonas especificadas, pero no se garantiza que la distribución sea uniforme.
- Para lograr una distribución predecible y uniforme entre las zonas de disponibilidad, cree grupos de nodos con capacidad estática independientes, cada uno fijado a una zona de disponibilidad específica según el requisito `topology.kubernetes.io/zone`.
- Si necesita 12 nodos distribuidos uniformemente en 3 zonas, cree 3 grupos de nodos con 4 réplicas cada uno, en lugar de un grupo de nodos con 12 réplicas en 3 zonas.

Solución de problemas

Nodos que no llegan a las réplicas deseadas:

- Compruebe si el valor de `limits.nodes` es suficiente.
- Compruebe que los requisitos no restrinjan demasiado la selección de instancias.
- Revise las AWS Service Quotas para los tipos de instancias y las regiones que utiliza.

Sustitución del nodo que tarda demasiado tiempo:

- Ajuste los presupuestos de interrupción para permitir más sustituciones simultáneas.
- Compruebe si los PodDisruptionBudgets impiden la terminación del nodo.

Terminación inesperada del nodo:

- Revise las configuraciones `expireAfter` y `terminationGracePeriod`.
- Compruebe si hay terminaciones manuales del nodo o eventos de mantenimiento de AWS.

Creación de una IngressClass para configurar el equilibrador de carga de aplicación

El modo automático de EKS automatiza las tareas rutinarias de equilibrio de carga, incluida la exposición de las aplicaciones del clúster a Internet.

AWS sugiere usar equilibradores de carga de aplicación (ALB) para atender el tráfico HTTP y HTTPS. Los equilibradores de carga de aplicación pueden dirigir las solicitudes en función del contenido de la solicitud. Para obtener más información sobre los equilibradores de carga de aplicación, consulte ¿Qué es un equilibrador de carga de aplicación?

El modo automático de EKS crea y configura equilibradores de carga de aplicación (ALB). Por ejemplo, el modo automático de EKS crea un equilibrador de carga cuando se crea un objeto de Kubernetes de `Ingress` y lo configura de modo que dirija el tráfico a la carga de trabajo del clúster.

Información general

1. Cree una carga de trabajo que desea exponer a Internet.
2. Cree un recurso de `IngressClassParams` y especifique los valores de configuración específicos de AWS, como el certificado que se utilizará para SSL/TLS y las subredes de la VPC.
3. Cree un recurso de `IngressClass` y especifique que el modo automático de EKS será el controlador del recurso.
4. Cree un recurso de `Ingress` que asocie una ruta HTTP y un puerto a una carga de trabajo de clúster.

El modo automático de EKS creará un equilibrador de carga de aplicación que apunte a la carga de trabajo especificada en el recurso de `Ingress`, con la configuración del equilibrador de carga especificada en el recurso de `IngressClassParams`.

Requisitos previos

- El modo automático de EKS habilitado en un clúster de Amazon EKS
- Kubectl configurado para establecer conexión con el clúster
  - Puede utilizar `kubectl apply -f <filename>` para aplicar al clúster los archivos YAML de configuración de ejemplo que aparecen a continuación.

> **Note**
>
> El modo automático de EKS requiere etiquetas de subred para identificar las subredes públicas y privadas.
> Si creó el clúster con `eksctl`, ya dispone de estas etiquetas.
> Información sobre cómo the section called “Etiquetado de subredes”.

Paso 1: creación de una carga de trabajo

Para comenzar, cree una carga de trabajo que desee exponer a Internet. Puede ser cualquier recurso de Kubernetes que sirva tráfico HTTP, como una implementación o un servicio.

En este ejemplo, se utiliza un servicio HTTP simple llamado `service-2048` que escucha en el puerto `80`. Para crear este servicio y su implementación, aplique el siguiente manifiesto, `2048-deployment-service.yaml`:

```
---
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
```

Aplique la configuración al clúster:

```
kubectl apply -f 2048-deployment-service.yaml
```

Los recursos enumerados anteriormente se crearán en el espacio de nombres predeterminado. Para verificarlo, ejecute el siguiente comando:

```
kubectl get all -n default
```

Paso 2: creación de IngressClassParams

Cree un objeto de `IngressClassParams` para especificar opciones de configuración específicas de AWS para el equilibrador de carga de aplicación. En este ejemplo, creamos un recurso `IngressClassParams` denominado `alb` (que utilizará en el siguiente paso) que especifica el esquema del equilibrador de carga, tal como se muestra `internet-facing` en un archivo llamado `alb-ingressclassparams.yaml`.

```
apiVersion: eks.amazonaws.com/v1
kind: IngressClassParams
metadata:
  name: alb
spec:
  scheme: internet-facing
```

Aplique la configuración al clúster:

```
kubectl apply -f alb-ingressclassparams.yaml
```

Paso 3: creación de IngressClass

Cree una `IngressClass` que haga referencia a los valores de configuración específicos de AWS establecidos en el recurso `IngressClassParams` en un archivo llamado `alb-ingressclass.yaml`. Anote el nombre de la `IngressClass`. En este ejemplo, se asigna el nombre de `alb` tanto a la `IngressClass` como a los `IngressClassParams`.

Utilice la anotación `is-default-class` para controlar si los recursos de `Ingress` deben usar esta clase de forma predeterminada.

```
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
```

Para obtener más información acerca de las opciones de configuración, consulte the section called “Referencia de IngressClassParams”.

Aplique la configuración al clúster:

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

Paso 4: creación del ingreso

Cree un recurso `Ingress` en un archivo llamado `alb-ingress.yaml`. El propósito de este recurso es asociar las rutas y los puertos del equilibrador de carga de aplicación a las cargas de trabajo del clúster. Para este ejemplo, creamos un recurso `Ingress` denominado `2048-ingress` que enruta el tráfico a un servicio denominado `service-2048` en el puerto 80.

Para obtener más información sobre la configuración de este recurso, consulte Ingreso en la documentación de Kubernetes.

```
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
```

Aplique la configuración al clúster:

```
kubectl apply -f alb-ingress.yaml
```

Paso 5: comprobación del estado

Utilice `kubectl` para encontrar el estado de `Ingress`. El equilibrador de carga puede tardar unos minutos en estar disponible.

Utilice el nombre del recurso de `Ingress` que definió en el paso anterior. Por ejemplo:

```
kubectl get ingress 2048-ingress
```

Una vez que el recurso esté listo, recupere el nombre de dominio del equilibrador de carga.

```
kubectl get ingress 2048-ingress -o
 jsonpath='{.status.loadBalancer.ingress[0].hostname}'
```

Para ver el servicio en un navegador web, revise el puerto y la ruta especificados en el recurso de `Ingress`.

Paso 6: Efectúe una limpieza

Para limpiar el equilibrador de carga, utilice el siguiente comando:

```
kubectl delete ingress 2048-ingress
kubectl delete ingressclass alb
kubectl delete ingressclassparams alb
```

El modo automático de EKS eliminará automáticamente el equilibrador de carga asociado en la cuenta de AWS.

Referencia de IngressClassParams

La siguiente tabla es una referencia rápida de las opciones de configuración que se utilizan con mayor frecuencia.

Campo	Descripción	Ejemplo de valor
`scheme`	Define si el equilibrador de carga de aplicación es interno o está orientado a Internet	`internet-facing`
`namespaceSelector`	Restringe los espacios de nombres que pueden usar esta IngressClass	`environment: prod`
`group.name`	Agrupa varios ingresos para compartir un único equilibrador de carga de aplicación	`retail-apps`
`ipAddressType`	Establece el tipo de dirección IP para el equilibrador de carga de aplicación	`dualstack`
`subnets.ids`	Lista de identificadores de subred para la implementación del equilibrador de carga de aplicación	`subnet-xxxx, subnet-yyyy`
`subnets.tags`	Etiquete los filtros para seleccionar las subredes del equilibrador de carga de aplicación	`Environment: prod`
`certificateARNs`	Los ARN de los certificados SSL que se van a utilizar	`arn:aws:acm:region:account:certificate/id`
`tags`	Etiquetas personalizadas para los recursos de AWS	`Environment: prod, Team: platform`

Campo	Descripción	Ejemplo de valor
`loadBalancerAttributes`	Atributos específicos del equilibrador de carga	`idle_timeout.timeout_seconds: 60`

Consideraciones

- No puede usar las anotaciones en una IngressClass para configurar los equilibradores de carga con el modo automático de EKS.
- No puede configurar ListenerAttribute con el Modo automático de EKS.
- Debe actualizar el rol de IAM del clúster para permitir la propagación de etiquetas desde Kubernetes a los recursos del equilibrador de carga de AWS. Para obtener más información, consulte the section called “Etiquetas personalizadas de AWS para los recursos del modo automático de EKS”.
- Para obtener información sobre la asociación de recursos con el modo automático de EKS o con el controlador del equilibrador de carga de AWS autoadministrado, consulte the section called “Referencia para las migraciones”.
- Para obtener información sobre cómo solucionar problemas con los equilibradores de carga, consulte the section called “Solución de problemas”.
- Para conocer más aspectos a tener en cuenta a la hora de utilizar la capacidad de equilibrio de carga del modo automático de EKS, consulte the section called “Equilibrio de carga”.

Las siguientes tablas proporcionan una comparación detallada de los cambios en las configuraciones de IngressClassParams, Ingress Annotations y TargetGroupBinding para el modo automático de EKS. Estas tablas destacan las principales diferencias entre la capacidad de equilibrio de carga del modo automático de EKS y el controlador del equilibrador de carga de código abierto, incluidos los cambios en la versión de la API, las características obsoletas y la actualización de los nombres de los parámetros.

IngressClassParams

Anteriores	New	Descripción
`elbv2.k8s.aws/v1beta1`	`eks.amazonaws.com/v1`	Cambio de versión de la API

Anteriores	New	Descripción
`spec.certificateArn`	`spec.certificateARNs`	Compatibilidad con múltiples ARN de certificados
`spec.subnets.tags`	`spec.subnets.matchTags`	Se modificó el esquema de coincidencia de subredes
`spec.listeners.listenerAttributes`	No admitido	Etiquetas que aún no son compatibles con el Modo automático de EKS

Anotaciones de ingreso

Anteriores	New	Descripción
`kubernetes.io/ingress.class`	No admitido	Utilice `spec.ingressClassName` en objetos de ingreso
`alb.ingress.kubernetes.io/group.name`	No admitido	Especifique grupos en IngressClass únicamente
`alb.ingress.kubernetes.io/waf-acl-id`	No admitido	En su lugar, utilice WAF v2
`alb.ingress.kubernetes.io/web-acl-id`	No admitido	En su lugar, utilice WAF v2
`alb.ingress.kubernetes.io/shield-advanced-protection`	No admitido	Integración de Shield desactivada
`alb.ingress.kubernetes.io/auth-type: oidc`	No admitido	Actualmente, no se admite el tipo de autenticación OIDC

TargetGroupBinding

Anteriores	New	Descripción
`elbv2.k8s.aws/v1beta1`	`eks.amazonaws.com/v1`	Cambio de versión de la API
`spec.targetType` opcional	`spec.targetType` obligatorio	Especificación explícita del tipo de destino
`spec.networking.ingress.from`	No admitido	Ya no es compatible con equilibradores de carga de red sin grupos de seguridad

Para utilizar la característica TargetGroupBinding personalizada, debe etiquetar el grupo de destino con la etiqueta eks:eks-cluster-name con el nombre del clúster para conceder al controlador los permisos de IAM necesarios. Tenga en cuenta que el controlador eliminará el grupo de destino cuando se elimine el recurso TargetGroupBinding o el clúster.

Cómo utilizar las anotaciones de servicio para configurar los equilibradores de carga de red

Aprenda a configurar los equilibradores de carga de red (NLB) en Amazon EKS mediante anotaciones de servicio de Kubernetes. En este tema se explican las anotaciones que admite el modo automático de EKS para personalizar el comportamiento de los NLB, incluidos los modos de accesibilidad a Internet, las comprobaciones de estado, la terminación de SSL/TLS y la segmentación por IP.

Al crear un servicio de Kubernetes de tipo `LoadBalancer` en el modo automático de EKS, EKS aprovisiona y configura automáticamente un equilibrador de carga de red de AWS en función de las anotaciones que especifique. Este enfoque declarativo permite administrar las configuraciones del equilibrador de carga directamente a través de los manifiestos de Kubernetes, manteniendo las prácticas de infraestructura como código.

El modo automático de EKS administra el aprovisionamiento del equilibrador de carga de red de forma predeterminada para todos los servicios del tipo LoadBalancer, sin necesidad de instalar ni configurar ningún controlador adicional. La especificación `loadBalancerClass:`

`eks.amazonaws.com/nlb` se establece automáticamente como la predeterminada del clúster, lo que agiliza el proceso de implementación y garantiza la compatibilidad con las cargas de trabajo de Kubernetes existentes.

> **Note**
>
> El modo automático de EKS requiere etiquetas de subred para identificar las subredes públicas y privadas.
> Si creó el clúster con `eksctl`, ya dispone de estas etiquetas.
> Información sobre cómo the section called “Etiquetado de subredes”.

Ejemplo de servicio

Para obtener más información sobre el recurso `Service` de Kubernetes, consulte la documentación de Kubernetes.

Revise el recurso de `Service` de muestra que aparece a continuación:

```
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
```

Anotaciones de uso frecuente

En la siguiente tabla se enumeran las anotaciones más utilizadas compatibles con el modo automático de EKS. Tenga en cuenta que es posible que el modo automático de EKS no admita todas las anotaciones.

> **Tip**
>
> Todas las anotaciones que aparecen a continuación deben ir precedidas de `service.beta.kubernetes.io/`

Campo	Descripción	Ejemplo
`aws-load-balancer-type`	Especifica el tipo de equilibrador de carga. Utilice `external` para nuevas implementaciones.	`external`
`aws-load-balancer-nlb-target-type`	Especifica si se debe dirigir el tráfico a las instancias de los nodos o directamente a las IP de los pods. Utilice `instance` para implementaciones estándar o `ip` para el enrutamiento directo a los pods.	`instance`
`aws-load-balancer-scheme`	Controla si el equilibrador de carga es interno o está expuesto a Internet.	`internet-facing`
`aws-load-balancer-healthcheck-protocol`	Protocolo de comprobación de estado correspondiente al grupo de destino. Las opciones más comunes son `TCP` (predeterminada) o `HTTP`.	`HTTP`

Campo	Descripción	Ejemplo
`aws-load-balancer-healthcheck-path`	La ruta HTTP para las comprobaciones de estado cuando se utiliza el protocolo HTTP/HTTPS.	`/healthz`
`aws-load-balancer-healthcheck-port`	El puerto que se utiliza para las comprobaciones de estado. Puede ser un número de puerto específico o `traffic-port` .	`traffic-port`
`aws-load-balancer-subnets`	Especifica en qué subredes se va a crear el equilibrador de carga. Puede usar nombres o ID de subred.	`subnet-xxxx, subnet-yyyy`
`aws-load-balancer-ssl-cert`	ARN del certificado SSL de AWS Certificate Manager para HTTPS/TLS.	`arn:aws:acm:region:account:certificate/cert-id`
`aws-load-balancer-ssl-ports`	Especifica qué puertos deben usar SSL/TLS.	`443, 8443`
`load-balancer-source-ranges`	Los rangos de CIDR permiten acceder al equilibrador de carga.	`10.0.0.0/24, 192.168.1.0/24`
`aws-load-balancer-additional-resource-tags`	Etiquetas de AWS adicionales para aplicarlas al equilibrador de carga y a los recursos relacionados.	`Environment=prod,Team=platform`
`aws-load-balancer-ip-address-type`	Especifica si el equilibrador de carga usa IPv4 o pila dual (IPv4 + IPv6).	`ipv4` o `dualstack`

Consideraciones

- Debe actualizar el rol de IAM del clúster para permitir la propagación de etiquetas desde Kubernetes a los recursos del equilibrador de carga de AWS. Para obtener más información, consulte the section called “Etiquetas personalizadas de AWS para los recursos del modo automático de EKS”.
- Para obtener información sobre la asociación de recursos con el modo automático de EKS o con el controlador del equilibrador de carga de AWS autoadministrado, consulte the section called “Referencia para las migraciones”.
- Para obtener información sobre cómo solucionar problemas con los equilibradores de carga, consulte the section called “Solución de problemas”.
- Para conocer más aspectos a tener en cuenta a la hora de utilizar la capacidad de equilibrio de carga del modo automático de EKS, consulte the section called “Equilibrio de carga”.

Al migrar al modo automático de EKS para el equilibrio de carga, es necesario realizar varios cambios en las anotaciones de servicio y las configuraciones de recursos. En las tablas que aparecen a continuación se describen las principales diferencias entre las implementaciones anteriores y las nuevas, incluidas las opciones no compatibles y las alternativas recomendadas.

Anotaciones del servicio

Anteriores	New	Descripción
`service.beta.kubernetes.io/load-balancer-source-ranges`	No admitido	Utilice `spec.loadBalancerSourceRanges` en el servicio
`service.beta.kubernetes.io/aws-load-balancer-type`	No admitido	Utilice `spec.loadBalancerClass` en el servicio
`service.beta.kubernetes.io/aws-load-balancer-internal`	No admitido	Uso de `service.beta.kubernetes.io/aws-load-balancer-scheme`

Anteriores	New	Descripción
`service.beta.kubernetes.io/aws-load-balancer-proxy-protocol`	No admitido	En su lugar, use `service.beta.kubernetes.io/aws-load-balancer-target-group-attributes`
Varios atributos del equilibrador de carga	No admitido	Uso de `service.beta.kubernetes.io/aws-load-balancer-attributes`
`service.beta.kubernetes.io/aws-load-balancer-access-log-enabled`	No admitido	En su lugar, use `service.beta.kubernetes.io/aws-load-balancer-attributes`
`service.beta.kubernetes.io/aws-load-balancer-access-log-s3-bucket-name`	No admitido	En su lugar, use `service.beta.kubernetes.io/aws-load-balancer-attributes`
`service.beta.kubernetes.io/aws-load-balancer-access-log-s3-bucket-prefix`	No admitido	En su lugar, use `service.beta.kubernetes.io/aws-load-balancer-attributes`
`service.beta.kubernetes.io/aws-load-balancer-cross-zone-load-balancing-enabled`	No admitido	En su lugar, use `service.beta.kubernetes.io/aws-load-balancer-attributes`

Para migrar desde anotaciones de atributos del equilibrador de cargas obsoletas, consolide estos ajustes en la anotación `service.beta.kubernetes.io/aws-load-balancer-attributes`. Esta anotación acepta una lista separada por comas de pares de clave y valor para varios atributos

del equilibrador de carga. Por ejemplo, para especificar el registro de acceso y el equilibrio de carga entre zonas, utilice el siguiente formato:

```
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
```

Este formato consolidado proporciona una forma más coherente y flexible de configurar los atributos del equilibrador de carga y, al mismo tiempo, reduce la cantidad de anotaciones individuales necesarias. Revise las configuraciones de servicio existentes y actualícelas para usar este formato consolidado.

TargetGroupBinding

Anteriores	New	Descripción
`elbv2.k8s.aws/v1beta1`	`eks.amazonaws.com/v1`	Cambio de versión de la API
`spec.targetType` opcional	`spec.targetType` obligatorio	Especificación explícita del tipo de destino
`spec.networking.ingress.from`	No admitido	Ya no es compatible con equilibradores de carga de red sin grupos de seguridad

Nota: Para utilizar la característica TargetGroupBinding personalizada, debe etiquetar el grupo de destino con la etiqueta `eks:eks-cluster-name` con el nombre del clúster para conceder al controlador los permisos de IAM necesarios. Tenga en cuenta que el controlador eliminará el grupo de destino cuando se elimine el recurso TargetGroupBinding o el clúster.

Creación de una clase de almacenamiento

Una `StorageClass` en el modo automático de EKS define cómo se aprovisionan automáticamente los volúmenes de Amazon EBS cuando las aplicaciones solicitan almacenamiento persistente. En esta página se explica cómo crear y configurar una `StorageClass` que funcione con el modo automático de Amazon EKS para aprovisionar volúmenes de EBS.

Al configurar una `StorageClass`, puede especificar la configuración predeterminada para los volúmenes de EBS, incluidos el tipo de volumen, el cifrado, las IOPS y otros parámetros de almacenamiento. También puede configurar `StorageClass` para que utilice claves de AWS KMS para la administración del cifrado.

El modo automático de EKS no crea una `StorageClass` en su nombre. Debe crear una `StorageClass` que haga referencia a `ebs.csi.eks.amazonaws.com` para utilizar la capacidad de almacenamiento del modo automático de EKS.

Primero, cree un archivo denominado `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"
```

En segundo lugar, aplique la clase de almacenamiento al clúster.

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

Componentes principales:

- `provisioner: ebs.csi.eks.amazonaws.com`: utiliza el modo automático de EKS
- `allowedTopologies`: al especificar `matchLabelExpressions` para que coincida con `eks.amazonaws.com/compute-type:auto`, se asegurará de que los pods no se programarán en nodos no automáticos si sus pods necesitan que un volumen se aprovisione automáticamente con el Modo automático.

- `volumeBindingMode: WaitForFirstConsumer`: retrasa la creación del volumen hasta que un pod lo necesite
- `type: gp3`: especifica el tipo de volumen de EBS
- `encrypted: "true"`: EBS cifrará todos los volúmenes creados con la `StorageClass`. EBS utilizará el alias de clave de `aws/ebs` predeterminado. Para obtener más información, consulte Cómo funciona el cifrado de Amazon EBS en la Guía del usuario de Amazon EBS. Este valor es opcional, pero recomendado.
- `storageclass.kubernetes.io/is-default-class: "true"`: Kubernetes utilizará esta clase de almacenamiento de forma predeterminada, a menos que se especifique una clase de volumen diferente en una reclamación de volumen persistente. Este valor es opcional. Tenga cuidado al establecer este valor si va a migrar desde un controlador de almacenamiento diferente.

Utilice una clave de KMS autoadministrada para cifrar los volúmenes de EBS

Para utilizar una clave de KMS autoadministrada para cifrar los volúmenes de EBS automatizados mediante el modo automático de EKS, deberá:

1. Crear una clave de KMS autoadministrada.
   - Para obtener más información, consulte Cómo crear una clave de KMS de cifrado simétrico o Cómo Amazon Elastic Block Store (Amazon EBS) usa KMS en la Guía del usuario de KMS.
2. Cree una nueva política que permita el acceso a la clave de KMS.
   - Utilice la política de IAM de muestra que aparece a continuación para crear la política. Inserte el ARN de la nueva clave de KMS autoadministrada. Para obtener más información, consulte Cómo crear roles y asociar políticas (consola) en la Guía de usuario de IAM AWS.
3. Asocie la política al rol de clúster de EKS.
   - Utilice la consola de AWS para buscar el ARN del rol de clúster de EKS. La información del rol aparece en la sección Descripción general. Para obtener más información, consulte the section called “Rol de IAM de clúster”.
4. Actualice la `StorageClass` de modo que haga referencia al ID de la clave de KMS en el campo `parameters.kmsKeyId`.

Ejemplo de política de IAM de KMS autoadministrada

Actualice los siguientes valores de la política que aparece a continuación:

- <account-id>: el ID de la cuenta de AWS, como 111122223333
- <aws-region>: la región de AWS del clúster, como 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"
            }
        }
    }
  ]
}
```

Ejemplo de **`StorageClass`** de KMS autoadministrada

```
parameters:
  type: gp3
  encrypted: "true"
  kmsKeyId: <custom-key-arn>
```

Referencia de los parámetros de **`StorageClass`**

Para obtener información general sobre los recursos de `StorageClass` de Kubernetes, consulte Clases de almacenamiento en la documentación de Kubernetes.

La sección `parameters` del recurso `StorageClass` es específica de AWS. Utilice la siguiente tabla para revisar las opciones disponibles.

Parameters	Valores	Predeterminado	Descripción
“csi.storage.k8s.io/fstype”	xfs, ext2, ext3, ext4	ext4	Tipo de sistema de archivos que se formateará durante la creación del volumen. Este parámetro distingue entre mayúsculas y minúsculas.
“tipo”	io1, io2, gp2, gp3, sc1, st1, standard, sbp1, sbg1	gp3	Tipo de volumen de EBS.
“iopsPerGB”			Operaciones de E/S por segundo por GiB. Se puede especificar para los volúmenes IO1, IO2 y GP3.
“allowAutoIOPSPerGBIncrease”	true, false	false	Si es `"true"`, el controlador CSI aumenta las IOPS

Parameters	Valores	Predeterminado	Descripción
			de un volumen si `iopsPerGB * <volume size>` es demasiado bajo para ajustarse en el rango de IOPS admitido por AWS. Esto permite que el aprovisionamiento dinámico siempre se realice correctam ente, incluso si el usuario especific a una capacidad de PVC o un valor de `iopsPerGB` demasiado bajo. Por otro lado, esto puede generar costos adicionales, ya que dichos volúmenes tienen IOPS más altas que las solicitadas en `iopsPerGB` .
“iops”			Operaciones de E/S por segundo. Se puede especificar para los volúmenes IO1, IO2 y GP3.

Parameters	Valores	Predeterminado	Descripción
“rendimiento”		125	El rendimiento se indica en MB/s. Solo es efectivo cuando se especifica el tipo de volumen gp3.
“cifrado”	true, false	false	Especifica si el volumen debe estar cifrado o no. Los valores válidos son “verdadero” o “falso”.
“blockExpress”	true, false	false	Habilita la creación de volúmenes io2 Block Express.
“kmsKeyId”			El ARN completo de la clave que se va a utilizar al cifrar el volumen. Si no se especifica, AWS utilizará la clave de KMS predeterminada para la región en la que se encuentra el volumen. Será una clave generada automáticamente y se llamará `/aws/ebs`, si no se modifica.

Parameters	Valores	Predeterminado	Descripción
“blockSize”			El tamaño del bloque que se utilizará al formatear el sistema de archivos subyacente. Solo se admite en nodos de Linux y con fstype `ext2`, `ext3`, `ext4` o `xfs`.
“inodeSize”			El tamaño del nodo de índice que se utilizará al formatear el sistema de archivos subyacente. Solo se admite en nodos de Linux y con fstype `ext2`, `ext3`, `ext4` o `xfs`.
“bytesPerInode”			Los `bytes-per-inode` que se utilizarán al formatear el sistema de archivos subyacente. Solo se admite en nodos de Linux y con fstype `ext2`, `ext3` y `ext4`.

Parameters	Valores	Predeterminado	Descripción
“numberOfInodes”			Los `number-of-inodes` que se utilizarán al formatear el sistema de archivos subyacente. Solo se admite en nodos de Linux y con fstype `ext2`, `ext3` y `ext4`.
“ext4BigAlloc”	true, false	false	Cambia el sistema de archivos `ext4` de modo que utilice la asignación de bloques en clústeres, para lo que habilita la opción de formateo `bigalloc`. Advertencia: es posible que `bigalloc` no sea completamente compatible con el núcleo de Linux del nodo.

Parameters	Valores	Predeterminado	Descripción
“ext4ClusterSize”			El tamaño del clúster que se utilizará al formatear un sistema de archivos `ext4`cuando la característica `bigalloc` esté habilitada. Nota: El parámetro `ext4BigAlloc` debe estar estableci do en “verdadero”.

Para obtener más información, consulte Controlador de CSI de AWS EBS en GitHub.

Consideraciones

> **Note**
>
> Solo puede implementar cargas de trabajo que dependan de las clases de almacenamiento del modo automático de EKS en los nodos del modo automático de EKS. Si tiene un clúster con varios tipos de nodos, debe configurar sus cargas de trabajo para que se ejecuten únicamente en los nodos del modo automático de EKS. Para obtener más información, consulte the section called “Control de la implementación”.

La capacidad de almacenamiento en bloques del modo automático de EKS es diferente a la del controlador de CSI de EBS.

- Aprovisionamiento estático
  - Si desea utilizar volúmenes de EBS creados externamente con el modo automático de EKS, tendrá que agregar manualmente una etiqueta `eks:eks-cluster-name` con la clave AWS y el valor del nombre del clúster.
- Taint de inicio de nodos

  - No puede usar la característica de taint de inicio de nodos para evitar la programación de pods antes de que la capacidad de almacenamiento esté lista.
- Etiquetas personalizadas en volúmenes aprovisionados dinámicamente
  - No puede utilizar la marca de la CLI de etiquetas adicionales para configurar etiquetas personalizadas en volúmenes de EBS aprovisionados dinámicamente
  - Puede utilizar el etiquetado de `StorageClass` para agregar etiquetas personalizadas. El modo automático de EKS agregará etiquetas a los recursos de AWS asociados. Deberá actualizar el rol de IAM del clúster para las etiquetas personalizadas. Para obtener más información, consulte the section called “Etiquetas personalizadas de AWS para los recursos del modo automático de EKS”.
- Métricas de rendimiento detalladas de EBS
  - No puede acceder a las métricas de Prometheus para obtener el rendimiento detallado de EBS

Instale el complemento de controlador de instantáneas de CSI

El modo automático de EKS es compatible con el complemento de Amazon EKS de controlador de instantáneas de CSI.

AWS sugiere configurar este complemento para que se ejecute en el grupo de nodos del `system` integrado.

Para obtener más información, consulte:

- the section called “Ejecución de complementos esenciales”
- the section called “Revisión de los grupos de nodos integrados”
- the section called “Controlador de instantáneas CSI”

Para instalar el controlador de instantáneas en el grupo de nodos del sistema

1. Abra el clúster de EKS en la consola de AWS
2. En la pestaña Complementos, seleccione Obtener más complementos
3. Seleccione el Controlador de instantáneas de CSI y luego Siguiente
4. En la página Configuración de los ajustes de los complementos seleccionados, seleccione Ajustes de configuración opcionales para ver el Esquema de configuración del complemento

a. Inserte el siguiente yaml para asociar el controlador de instantáneas al grupo de nodos del `system`. El controlador de instantáneas incluye una tolerancia a la taint `CriticalAddonsOnly`.

```
{
        "nodeSelector": {
            "karpenter.sh/nodepool": "system"
        }
}
```

b. Seleccione Siguiente.

5. Revise la configuración del complemento y, a continuación, seleccione Crear

Cómo desactivar el modo automático de EKS

Puede desactivar el modo automático de EKS en un clúster de EKS existente. Se trata de una operación destructiva.

- EKS terminará todas las instancias de EC2 operadas por el modo automático de EKS.
- EKS eliminará todos los equilibradores de carga operados por el modo automático de EKS.
- EKS no eliminará los volúmenes de EBS aprovisionados por el modo automático de EKS.

El modo automático de EKS está diseñado para administrar completamente los recursos que crea. Las intervenciones manuales podrían provocar que el modo automático de EKS no limpie completamente esos recursos cuando está desactivado. Por ejemplo, si ha hecho referencia a un grupo de seguridad administrado desde las reglas de un grupo de seguridad externo y se olvida de eliminar esa referencia antes de desactivar el modo automático de EKS para un clúster, el grupo de seguridad administrado no se eliminará y quedará residual. Los siguientes pasos describen cómo eliminar un grupo de seguridad residual en caso de que eso ocurra.

Desactivación del modo automático de EKS (Consola de AWS)

1. Abra la página de información general del clúster en la Consola de administración de AWS.
2. En Modo automático de EKS, seleccione Administrar
3. Desactivar el Modo automático de EKS con la opción `off`.

Si un grupo de seguridad administrado no se elimina al final de este proceso, podrá eliminarlo manualmente con las descripciones de Eliminar un grupo de seguridad.

Desactivación del modo automático de EKS (AWS CLI)

Utilice el siguiente comando para desactivar el modo automático de EKS en un clúster existente.

Debe tener `aws` CLI instalada y haber iniciado sesión con los permisos suficientes para administrar los clústeres de EKS. Para obtener más información, consulte Configuración.

> **Note**
>
> Las capacidades de computación, almacenamiento en bloque y equilibrio de carga se deben habilitar o desactivar en la misma solicitud.

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

Puede verificar si un grupo de seguridad del modo automático de EKS quedó residual y no se eliminó después de desactivar el modo automático de EKS de la siguiente manera:

```
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]"
```

Para después eliminar el grupo de seguridad:

```
aws ec2 delete-security-group --group-name=<sg-name>
```

Actualización de la versión de Kubernetes de un clúster del modo automático de EKS

En este tema se explica cómo actualizar la versión de Kubernetes del clúster del modo automático. El modo automático simplifica el proceso de actualización de versiones al gestionar la coordinación

de las actualizaciones del plano de control con las sustituciones de nodos, mientras mantiene la disponibilidad de la carga de trabajo gracias a los presupuestos de interrupción de pods.

Al actualizar un clúster del modo automático, muchos componentes que tradicionalmente requerían actualizaciones manuales ahora se administran como parte del servicio. Entender los aspectos automatizados del proceso de actualización y las responsabilidades asociadas contribuye a asegurar una transición de versión fluida para el clúster.

Más información sobre las actualizaciones con el modo automático de EKS

Una vez iniciada la actualización del plano de control, el modo automático de EKS comenzará a actualizar los nodos del clúster. A medida que los nodos expiren, el modo automático de EKS los reemplazará con nodos nuevos. Los nuevos nodos cuentan con la nueva versión de Kubernetes correspondiente. El modo automático de EKS respeta los presupuestos de interrupción de pods al actualizar nodos.

Además, ya no tendrá que actualizar componentes como:

- CNI de Amazon VPC
- Controlador del equilibrador de carga de AWS
- CoreDNS
- `kube-proxy`
- Karpenter
- Controlador CSI de AWS EBS

El modo automático de EKS sustituye estos componentes por funciones de servicio.

Aún es responsable de la actualización de:

- Aplicaciones y cargas de trabajo implementadas en el clúster
- Complementos y controladores autoadministrados
- Complementos de Amazon EKS
  - Información sobre cómo the section called “Cómo actualizar un complemento”

Más información sobre las prácticas recomendadas para la actualización de clústeres

Cómo iniciar la actualización del clúster

Para iniciar una actualización del clúster, consulte the section called “Actualización de una versión de Kubernetes”.

Cómo habilitar o desactivar los NodePools integrados

El modo automático de EKS tiene dos NodePools integrados. Puede habilitar o desactivar estos NodePools mediante la consola de AWS, la CLI o la API.

Referencia de NodePool integrado

- `system`
  - Este NodePool tiene una taint `CriticalAddonsOnly`. Muchos complementos de EKS, como CoreDNS, toleran esta taint. Utilice este grupo de nodos del sistema para separar las aplicaciones críticas del clúster.
  - Admite tanto arquitecturas `amd64` como `arm64`.
- `general-purpose`
  - Este NodePool admite el lanzamiento de nodos para cargas de trabajo de uso general en el clúster.
  - Utiliza solo la arquitectura `amd64`.

Ambos NodePools integrados:

- Utilizan la NodeClass de EKS predeterminada
- Utilizan únicamente la capacidad de EC2 bajo demanda
- Utilizan las familias de instancias C, M y R de EC2
- Exigen instancias de EC2 de generación 5 o posterior

> **Note**
>
> Es necesario habilitar al menos un NodePool integrado para que EKS aprovisione la NodeClass “predeterminada”. Si desactiva todos los NodePools integrados, tendrá que crear una NodeClass personalizada y configurar un NodePool para usarla. Para obtener más

información acerca de NodeClasses, consulte the section called “Cómo crear una clase de nodos”.

Procedimiento

Requisitos previos

- La versión más reciente de la Interfaz de Línea de Comandos de AWS (AWS CLI) instalada y configurada en el dispositivo. Para comprobar su versión actual, utilice `aws --version`. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS.
  - Inicie sesión en la CLI con permisos de IAM suficientes para crear recursos de AWS, como políticas de IAM, roles de IAM y clústeres de EKS.

Habilitar con la CLI de AWS

Utilice el siguiente comando para habilitar ambos NodePools integrados:

```
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}
  }'
```

Puede modificar el comando para habilitar los NodePools de forma selectiva.

Desactivar con la CLI de AWS

Utilice el siguiente comando para desactivar ambos NodePools integrados:

```
aws eks update-cluster-config \
```

```
  --name <cluster-name> \
  --compute-config '{
  "enabled": true,
  "nodePools": []
  }' \
  --kubernetes-network-config '{
  "elasticLoadBalancing":{"enabled": true}}' \
  --storage-config '{
  "blockStorage":{"enabled": true}
  }'
```

Cómo controlar si una carga de trabajo se implementa en nodos del modo automático de EKS

Al ejecutar cargas de trabajo en un clúster de EKS con el modo automático de EKS, es posible que deba controlar si determinadas cargas de trabajo se ejecutan en nodos del modo automático de EKS o en otros tipos de computación. En este tema se describe cómo usar los selectores de nodos y las reglas de afinidad para garantizar que las cargas de trabajo estén programadas en la infraestructura de computación prevista.

Los ejemplos de este tema muestran cómo utilizar la etiqueta `eks.amazonaws.com/compute-type` para exigir o impedir la implementación de la carga de trabajo en los nodos del modo automático de EKS. Esto resulta especialmente útil en clústeres de modo mixto en los que se ejecutan tanto el modo automático de EKS como otros tipos de computación, como los aprovisionadores Karpenter autoadministrados o los grupos de nodos administrados por EKS.

Los nodos del modo automático de EKS han establecido el valor de la etiqueta `eks.amazonaws.com/compute-type` en `auto`. Muestra cómo utilizar esta marca para controlar si una carga de trabajo se implementa en nodos administrados por el modo automático de EKS.

Es necesario implementar una carga de trabajo en los nodos del modo automático de EKS

> **Note**
>
> Este valor de `nodeSelector` no es obligatorio para el modo automático de EKS. Este valor de `nodeSelector` es relevante únicamente si ejecuta un clúster en un modo mixto, con tipos de nodos no administrados por el modo automático de EKS. Por ejemplo, es posible que haya capacidad de computación estática implementada en el clúster con grupos

de nodos administrados por EKS, así como una capacidad de computación dinámica administrada por el modo automático de EKS.

Puede agregar este `nodeSelector` a las implementaciones u otras cargas de trabajo para exigir a Kubernetes que las programe en los nodos del modo automático de EKS.

```
apiVersion: apps/v1
kind: Deployment
spec:
  template:
    nodeSelector:
      eks.amazonaws.com/compute-type: auto
```

Requiere que una carga de trabajo no se implemente en nodos del modo automático de EKS.

Puede agregar esta `nodeAffinity` a las implementaciones u otras cargas de trabajo para exigir a Kubernetes que no las programe en los nodos del modo automático de EKS.

```
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: eks.amazonaws.com/compute-type
            operator: NotIn
            values:
            - auto
```

Ejecución de complementos esenciales en instancias dedicadas

En este tema, aprenderá a implementar una carga de trabajo con una tolerancia `CriticalAddonsOnly` para que el modo automático de EKS la programe en el grupo de nodos de `system`.

El grupo de nodos del `system` integrado del modo automático de EKS está diseñado para ejecutar complementos esenciales en instancias dedicadas. Esta división asegura que los componentes esenciales dispongan de recursos dedicados y se mantengan aislados de las cargas de trabajo generales, mejorando así la estabilidad y el rendimiento global del clúster.

Esta guía muestra cómo implementar complementos en el grupo de nodos del `system` mediante la tolerancia `CriticalAddonsOnly` y los selectores de nodos adecuados. Si sigue estos pasos, podrá asegurarse de que las aplicaciones críticas se programen en los nodos dedicados del `system`, aprovechando las ventajas de aislamiento y asignación de recursos que ofrece la estructura especializada de grupos de nodos del modo automático de EKS.

El modo automático de EKS tiene dos grupos de nodos integrados: `general-purpose` y `system`. Para obtener más información, consulte the section called “Revisión de los grupos de nodos integrados”.

El propósito del grupo de nodos del `system` es segregar los complementos críticos en diferentes nodos. Los nodos aprovisionados por el grupo de nodos del `system` tienen una taint de Kubernetes `CriticalAddonsOnly`. Kubernetes solo programará los pods en estos nodos si tienen la tolerancia correspondiente. Para obtener más información, consulte Taints y toleraciones en la documentación de Kubernetes.

Requisitos previos

- Clúster de modo automático de EKS con el grupo de nodos de `system` integrado habilitado. Para obtener más información, consulte the section called “Revisión de los grupos de nodos integrados”
- `kubectl` instalado y configurado Para obtener más información, consulte Configuración.

Procedimiento

Revise el ejemplo de yaml que aparece a continuación. Tenga en cuenta las siguientes condiciones:

- `nodeSelector`: esto asocia la carga de trabajo con el grupo de nodos integrado del `system`. Este grupo de nodos debe estar habilitado con la API AWS. Para obtener más información, consulte the section called “Revisión de los grupos de nodos integrados”.
- `tolerations`: esta tolerancia supera la taint de `CriticalAddonsOnly` en los nodos del grupo de nodos del `system`.

```
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"
```

Para actualizar una carga de trabajo de modo que se ejecute en el grupo de nodos del `system`, deberá:

1. Actualizar la carga de trabajo existente para agregar las siguientes configuraciones descritas anteriormente:
   - `nodeSelector`
   - `tolerations`
2. Implementar la carga de trabajo actualizada en el clúster con `kubectl apply`

Tras actualizar la carga de trabajo, se ejecutará en nodos dedicados.

Cómo utilizar las políticas de red con el modo automático de EKS

Descripción general

A medida que los clientes escalan sus entornos de aplicaciones mediante EKS, el aislamiento del tráfico de la red se vuelve cada vez más fundamental para evitar el acceso no autorizado a los recursos tanto dentro como fuera del clúster. Esto es especialmente importante en un entorno de varios inquilinos con varias cargas de trabajo no relacionadas que se ejecutan en paralelo en el

clúster. Las políticas de red de Kubernetes le permiten mejorar la posición de seguridad de la red para sus cargas de trabajo de Kubernetes y sus integraciones con los puntos de conexión externos al clúster. El modo automático de EKS admite distintos tipos de políticas de red.

Aislamiento de las capas 3 y 4

Las políticas de red estándar de Kubernetes operan en las capas 3 y 4 del modelo de red OSI y le permiten controlar el flujo de tráfico en la dirección IP o el puerto dentro del clúster de Amazon EKS.

Casos de uso

- Segmente el tráfico de red entre las cargas de trabajo para garantizar que solo las aplicaciones relacionadas puedan comunicarse entre sí.
- Aísle los inquilinos del espacio de nombres mediante políticas que impongan la separación de la red.

Aplicación basada en DNS

Los clientes suelen implementar cargas de trabajo en EKS que forman parte de un entorno distribuido más amplio, algunas de las cuales tienen que comunicarse con sistemas y servicios externos al clúster (tráfico en dirección norte). Estos sistemas y servicios pueden estar en la nube de AWS o completamente fuera de AWS. Las políticas basadas en el sistema de nombres de dominio (DNS) le permiten reforzar su posición de seguridad mediante la adopción de un enfoque más estable y predecible para evitar el acceso no autorizado desde los pods a los recursos o puntos de conexión externos al clúster. Este mecanismo elimina la necesidad de hacer un seguimiento manual de direcciones IP específicas y permitir incluirlas en una lista. Al proteger los recursos con un enfoque basado en DNS, también tiene más flexibilidad para actualizar la infraestructura externa sin tener que flexibilizar la posición de seguridad ni modificar las políticas de red en caso de cambios en los servidores y hosts ascendentes. Puede filtrar el tráfico de salida hacia puntos de conexión externos mediante un nombre de dominio completo (FQDN) o un patrón coincidente para un nombre de dominio de DNS. Esto le ofrece la flexibilidad adicional de ampliar el acceso a varios subdominios asociados a un punto de conexión externo al clúster en particular.

Casos de uso

- Estandarice según un enfoque basado en DNS para filtrar el acceso desde un entorno de Kubernetes a los puntos de conexión externos al clúster.
- Proteja el acceso a los servicios de AWS en un entorno de varios inquilinos.

- Administre el acceso a la red desde los pods hasta las cargas de trabajo en las instalaciones en sus entornos de nube híbrida.

Reglas de administración (o según el ámbito del clúster)

En algunos casos, como en los escenarios de varios inquilinos, es posible que los clientes tengan que aplicar un estándar de seguridad de red a todo el clúster. En lugar de definir y mantener de forma repetitiva una política distinta para cada espacio de nombres, puede usar una política única para administrar de forma centralizada los controles de acceso a la red para las diferentes cargas de trabajo del clúster, independientemente de su espacio de nombres. Estos tipos de políticas le permiten ampliar el ámbito de aplicación de las reglas de filtrado de red que se aplican en las capas 3 y 4 y cuando se utilizan las reglas de DNS.

Casos de uso

- Administre de forma centralizada los controles de acceso a la red para todas las cargas de trabajo (o un subconjunto de ellas) del clúster de EKS.
- Defina una posición de seguridad de red predeterminada en todo el clúster.
- Amplíe los estándares de seguridad de la organización al ámbito del clúster de una manera más eficiente desde el punto de vista operativo.

Introducción

Requisitos previos

- Un clúster de Amazon EKS con el modo automático de EKS habilitado
- kubectl configurado para conectarse al clúster

Paso 1: Habilitación del controlador de políticas de red

Para utilizar las políticas de red con el modo automático de EKS, primero debe habilitar el controlador de políticas de red, para lo cual se aplica un ConfigMap al clúster.

1. Cree un archivo denominado `enable-network-policy.yaml` con el siguiente contenido:

```
apiVersion: v1
kind: ConfigMap
metadata:
```

```
  name: amazon-vpc-cni
  namespace: kube-system
data:
  enable-network-policy-controller: "true"
```

2. Aplique el ConfigMap al clúster

```
kubectl apply -f enable-network-policy.yaml
```

Paso 2: Habilitación de políticas de red en Clase de nodo

Antes de poder utilizar las políticas de red, debe asegurarse de que la Clase de nodo está configurada para admitirlas. Siga estos pasos:

1. Cree o edite un archivo YAML de Clase de nodo (por ejemplo, `nodeclass-network-policy.yaml`) con el siguiente contenido:

```
apiVersion: eks.amazonaws.com/v1
kind: NodeClass
metadata:
  name: network-policy-enabled
spec:
  # Enables network policy support
  networkPolicy: DefaultAllow
  # Optional: Enables logging for network policy events
  networkPolicyEventLogs: Enabled
  # Include other Node Class configurations as needed
```

2. Aplique la configuración de Clase de nodo al clúster:

```
kubectl apply -f nodeclass-network-policy.yaml
```

3. Verifique que se haya creado la Clase de nodo:

```
kubectl get nodeclass network-policy-enabled
```

4. Actualice el Grupo de nodos para utilizar esta Clase de nodo. Para obtener más información, consulte the section called “Creación de un grupo de nodos”.

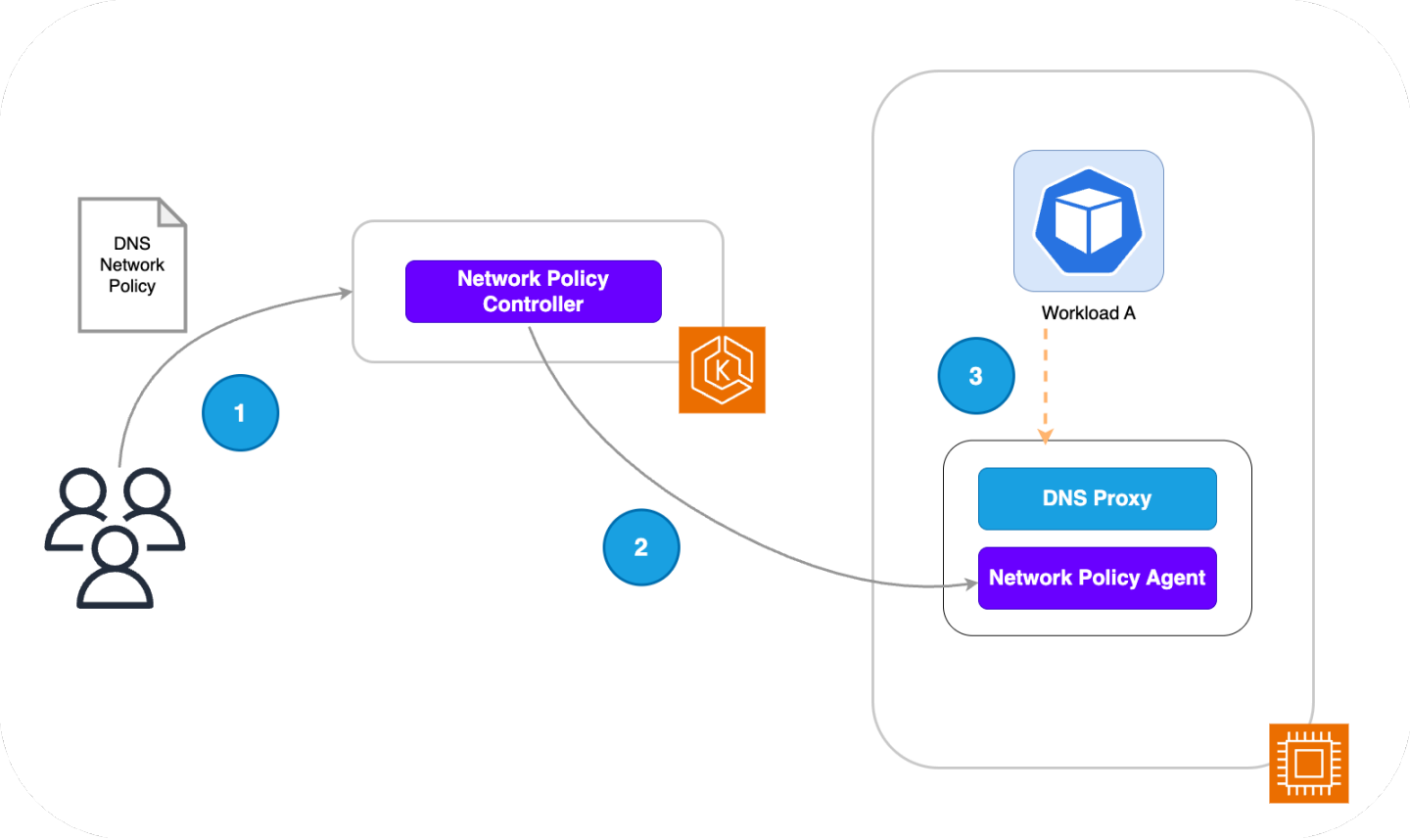
Una vez que los nodos utilicen esta Clase de nodo, podrán aplicar políticas de red. Ahora puede proceder a crear y aplicar políticas de red para controlar el tráfico dentro del clúster. Para conocer todas las opciones de configuración de las clases de nodos, consulte the section called “Cómo crear una clase de nodos”.

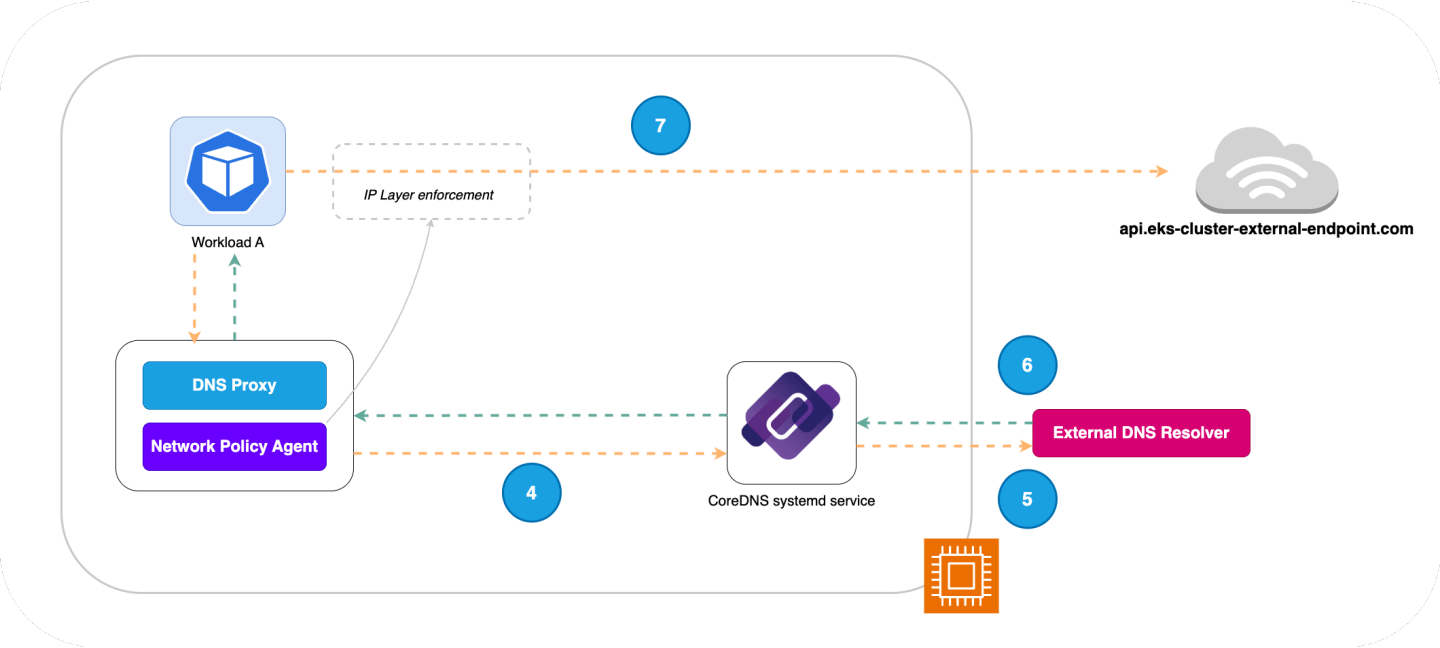
Paso 3: Cómo crear y probar políticas de red

El clúster del modo automático de EKS ya está configurado para admitir las políticas de red de Kubernetes. Puede probar esto con la the section called “Demostración de política Stars”.

¿Cómo funciona?

Política de red basada en DNS





1. El equipo de la plataforma aplica una política basada en DNS al clúster de EKS.
2. El controlador de políticas de red es responsable de supervisar la creación de políticas dentro del clúster y, a continuación, conciliar los puntos de conexión de las políticas. En este caso de uso, el controlador de políticas de red indica al agente de nodos que filtre las solicitudes de DNS en función de los dominios permitidos en la política creada. Los nombres de dominio se enumeran en la lista de dominios permitidos mediante el FQDN o un nombre de dominio que coincida con un patrón definido en la configuración de recursos de Kubernetes.
3. La carga de trabajo A intenta resolver la IP de un punto de conexión externo al clúster. La solicitud de DNS pasa primero por un proxy que filtra dichas solicitudes en función de la lista de solicitudes permitidas aplicada a través de la política de red.
4. Una vez que la solicitud de DNS pasa por la lista de filtros de DNS permitidos, se envía por proxy a CoreDNS.
5. A su vez, CoreDNS envía la solicitud al Solucionador de DNS externo (Amazon Route 53 Resolver) para obtener la lista de direcciones IP detrás del nombre de dominio.
6. Las IP resueltas con TTL se devuelven en respuesta a la solicitud de DNS. A continuación, estas direcciones IP se escriben en un mapa de eBPF que se utiliza en el siguiente paso para la aplicación de la capa de IP.
7. A continuación, las sondas de eBPF conectadas a la interfaz veth del pod filtrarán el tráfico de salida de la carga de trabajo A al punto de conexión externo del clúster en función de las reglas vigentes. Esto garantiza que los pods solo puedan enviar tráfico externo al clúster a las IP de los

dominios de la lista de dominios permitidos. La validez de estas IP se basa en el TTL obtenido del Solucionador de DNS externo (Amazon Route 53 Resolver).

Uso de la política de red de la aplicación

La `ApplicationNetworkPolicy` combina las capacidades de las políticas de red estándar de Kubernetes con el filtrado basado en DNS del espacio de nombres mediante una única definición de recursos personalizados (CRD). Por lo tanto, `ApplicationNetworkPolicy` se puede utilizar para:

1. Definir las restricciones en las capas 3 y 4 de la pila de red mediante bloques de IP y números de puerto.
2. Definir reglas que operen en la capa 7 de la pila de red y permitir filtrar el tráfico en función de los FQDN.

Nota importante: Las reglas basadas en DNS definidas mediante la `ApplicationNetworkPolicy` solo se aplican a las cargas de trabajo que se ejecutan en instancias de EC2 lanzadas por el modo automático de EKS.

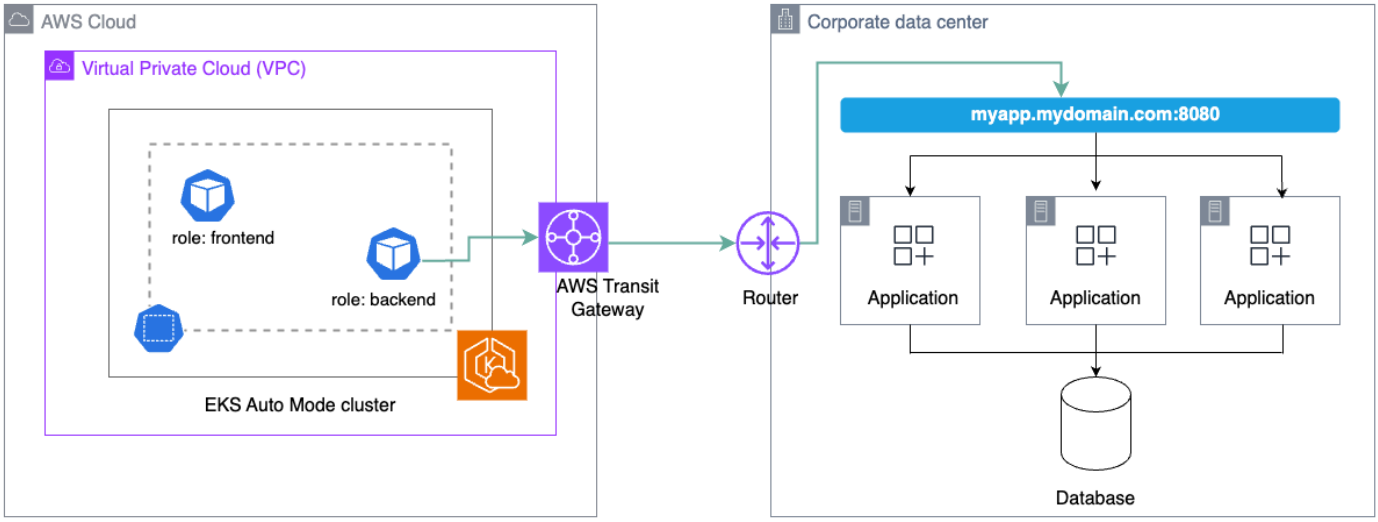
Ejemplo

Tiene una carga de trabajo en el clúster del modo automático de EKS que necesita comunicarse con una aplicación en las instalaciones que se encuentra detrás de un equilibrador de carga con un nombre de DNS. Para ello, puede utilizar la siguiente política de red:

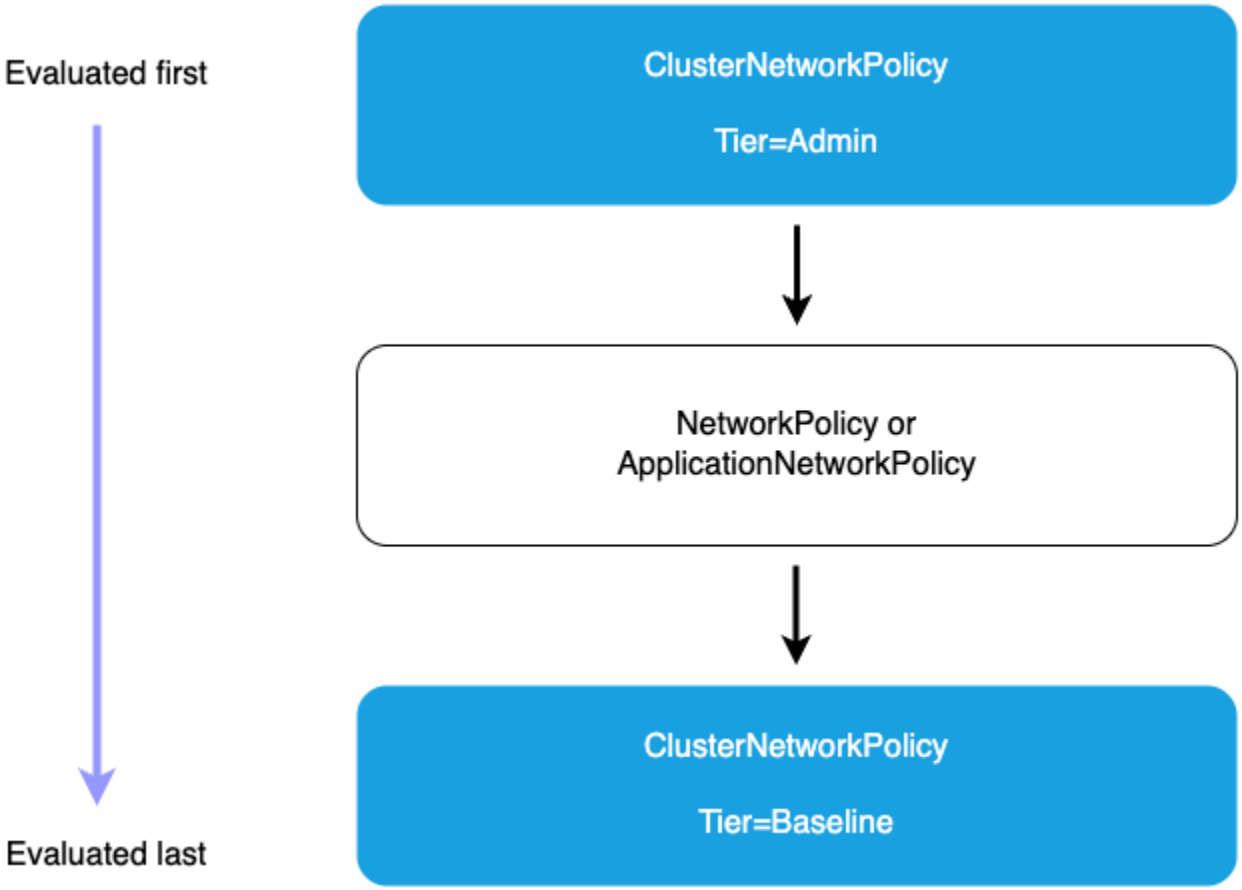
```
apiVersion: networking.k8s.aws/v1alpha1
kind: ApplicationNetworkPolicy
metadata:
  name: my-onprem-app-egress
  namespace: galaxy
spec:
  podSelector:
    matchLabels:
      role: backend
  policyTypes:
  - Egress
  egress:
  - to:
    - domainNames:
      - "myapp.mydomain.com"
    ports:
```

```
    - protocol: TCP
      port: 8080
```

En la red de Kubernetes, esto permitiría salir de cualquier pod del espacio de nombres “galaxy” etiquetado con `role: backend` para conectarse al nombre de dominio myapp.mydomain.com en el puerto TCP 8080. Además, tendría que configurar la conectividad de red para el tráfico de salida de la VPC al centro de datos corporativo.



Política de red de administración (o del clúster)



Uso de la política de red del clúster

Cuando se usa una `ClusterNetworkPolicy`, las políticas del nivel de administración se evalúan primero y no se pueden anular. Una vez evaluadas las políticas del nivel de administración, se utilizan las políticas estándar del ámbito del espacio de nombres para ejecutar las reglas de segmentación de red aplicadas. Esto se puede lograr mediante el uso de `ApplicationNetworkPolicy` o `NetworkPolicy`. Por último, se aplicarán las reglas del nivel de línea de base que definen las restricciones de red predeterminadas para las cargas de trabajo de los clústeres. Si es necesario, las políticas del ámbito del espacio de nombres pueden anular estas reglas del nivel de línea de base.

Ejemplo

Tiene una aplicación en el clúster que desea aislar de las cargas de trabajo de otros inquilinos. Puede bloquear de forma explícita el tráfico del clúster desde otros espacios de nombres para impedir el acceso de la red al espacio de nombres confidencial de la carga de trabajo.

```
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
```

Consideraciones

Descripción del orden de evaluación de políticas

Las capacidades de políticas de red compatibles con EKS se evalúan en un orden específico para garantizar una administración del tráfico predecible y segura. Por lo tanto, es importante comprender el flujo de evaluación para diseñar una posición de seguridad de red efectiva para el entorno.

1. Políticas del nivel de administración (se evalúan primero): todas las ClusterNetworkPolicies del nivel de administración se evalúan antes que cualquier otra política. En el nivel de administración, las políticas se procesan por orden de prioridad (se comienza por el número de prioridad más baja). El tipo de acción determina lo que sucede a continuación.
   - Acción de denegación (máxima prioridad): cuando una política de administración con una acción de denegación coincide con el tráfico, ese tráfico se bloquea inmediatamente, independientemente de cualquier otra política. No se procesan más reglas de ClusterNetworkPolicy ni NetworkPolicy. Esto garantiza que las políticas del espacio de nombres no puedan anular los controles de seguridad de toda la organización.
   - Acción de permiso: una vez evaluadas las reglas de denegación, las políticas de administración que incluyen acciones de permiso se procesan por orden de prioridad (se comienza por el número de prioridad más baja). Cuando una acción de permiso coincide, se acepta el tráfico y no se lleva a cabo ninguna otra evaluación de la política. Estas políticas pueden conceder el acceso a varios espacios de nombres en función de los selectores de etiquetas, lo que

proporciona un control centralizado sobre qué cargas de trabajo pueden acceder a recursos específicos.

- Acción de transferencia: las acciones de transferencia en las políticas del nivel de administración delegan la toma de decisiones a los niveles inferiores. Cuando el tráfico coincide con una regla de transferencia, la evaluación omite todas las reglas del nivel de administración restantes para ese tráfico y continúa directamente con el nivel de la NetworkPolicy. Esto permite a los administradores delegar explícitamente el control de determinados patrones de tráfico a los equipos de aplicaciones. Por ejemplo, puede usar reglas de transferencia para delegar la administración del tráfico dentro del espacio de nombres a los administradores del espacio de nombres y, al mismo tiempo, mantener controles estrictos sobre el acceso externo.

2. Nivel de política de red: si ninguna política del nivel de administración coincide con ninguna acción de denegación o permiso, o si coincide con una acción de transferencia, se evalúan los recursos de la ApplicationNetworkPolicy y de la NetworkPolicy tradicional del ámbito del espacio de nombres. Estas políticas proporcionan un control detallado dentro de los espacios de nombres individuales y las administran los equipos de aplicaciones. Las políticas relacionadas con el espacio de nombres solo pueden ser más restrictivas que las políticas de administración. No pueden anular ninguna decisión de denegación de una política de administración, pero pueden restringir aún más el tráfico permitido o transferido por las políticas de administración.
3. Políticas de administración del nivel de línea de base: si ninguna política de administración o del ámbito del espacio de nombres coincide con el tráfico, se evalúan las ClusterNetworkPolicies del nivel de línea de base. Proporcionan posiciones de seguridad predeterminadas que pueden ser anuladas por políticas del ámbito del espacio de nombres, lo que permite a los administradores establecer valores predeterminados para toda la organización y, al mismo tiempo, ofrecer a los equipos la flexibilidad necesaria para personalizarlas según sea necesario. Las políticas de referencia se evalúan por orden de prioridad (se comienza por el número de prioridad más baja).
4. Denegación de forma predeterminada (si ninguna política coincide): este comportamiento de denegación de forma predeterminada garantiza que solo se permitan las conexiones explícitamente permitidas, lo que mantiene una posición de seguridad sólida.

Aplicación del principio de privilegio mínimo

- Inicio con políticas restrictivas y adición de permisos gradualmente según sea necesario: comience por implementar políticas de denegación de forma predeterminada en el clúster y, a continuación, agregue reglas de autorización de forma gradual según valide los requisitos de conectividad legítimos. Este enfoque obliga a los equipos a justificar de forma explícita cada conexión externa, lo que crea un entorno más seguro y auditable.

- Auditoría y eliminación periódicas de las reglas de políticas no utilizadas: las políticas de red pueden acumularse con el tiempo a medida que las aplicaciones evolucionan y dejar atrás reglas obsoletas que amplían innecesariamente la superficie expuesta a ataques. Implemente un proceso de revisión periódico para identificar y eliminar las reglas de políticas que ya no sean necesarias, lo que garantiza que su posición de seguridad siga siendo estricta y fácil de mantener.
- Uso de nombres de dominio específico en lugar de patrones amplios cuando sea posible: si bien los patrones de comodines como `*.amazonaws.com` proporcionan conveniencia, también otorgan acceso a una amplia gama de servicios. Siempre que sea posible, especifique nombres de dominio exactos como `s3.us-west-2.amazonaws.com` para limitar el acceso solo a los servicios específicos que requieren las aplicaciones, lo que reduce el riesgo de movimientos laterales si la carga de trabajo se ve comprometida.

Uso de políticas basadas en DNS en EKS

- Las reglas basadas en DNS definidas mediante la `ApplicationNetworkPolicy` solo se aplican a las cargas de trabajo que se ejecutan en instancias de EC2 lanzadas por el modo automático de EKS. Si ejecuta un clúster de modo mixto (compuesto por nodos de trabajo del modo automático de EKS y nodos que no son del modo automático de EKS), las reglas basadas en DNS solo son efectivas en los nodos de trabajo del modo automático de EKS (instancias administradas de EC2).

Validación de las políticas de DNS

- Uso de clústeres de almacenamiento provisional que reflejen la topología de la red de producción para llevar a cabo pruebas: el entorno de pruebas debe replicar la arquitectura de la red, las dependencias externas y los patrones de conectividad de la producción para garantizar la precisión de las pruebas de políticas. Esto incluye configuraciones de VPC coincidentes, comportamiento de resolución de DNS y acceso a los mismos servicios externos que requieren las cargas de trabajo de producción.
- Implementación de pruebas automatizadas para rutas de red críticas: cree pruebas automatizadas que validen la conectividad con los servicios externos esenciales como parte del proceso de CI/CD. Estas pruebas deben verificar que se permiten los flujos de tráfico legítimos mientras se bloquean las conexiones no autorizadas, lo que permite validar continuamente que las políticas de red mantengan la posición de seguridad correcta a medida que la infraestructura evoluciona.
- Supervisión del comportamiento de las aplicaciones tras los cambios en las políticas: tras implementar políticas de red nuevas o modificadas en el entorno de producción, supervise de cerca los registros de las aplicaciones, las tasas de errores y las métricas de rendimiento para

identificar rápidamente cualquier problema de conectividad. Establezca procedimientos de reversión claros para poder revertir rápidamente los cambios en las políticas si provocan un comportamiento inesperado de las aplicaciones o interrupciones en el servicio.

Interacción con el firewall de DNS de Amazon Route 53

Las políticas de administración y de red de EKS se evalúan primero en el pod cuando se inicia el tráfico. Si una política de red de EKS permite la salida a un dominio específico, el pod lleva a cabo una consulta de DNS que llega a Route 53 Resolver. En este punto, se evalúan las reglas de firewall de DNS de Route 53. Si el firewall de DNS bloquea la consulta del dominio, se produce un error en la resolución de DNS y no se puede establecer la conexión, aunque la política de red de EKS lo permita. Esto crea capas de seguridad complementarias: las políticas de red basadas en DNS de EKS proporcionan un control de salida de pods para los requisitos de acceso específicos de la aplicación y los límites de seguridad de varios inquilinos, mientras que el firewall de DNS proporciona protección en toda la VPC contra los dominios maliciosos conocidos y aplica las listas de bloqueo en toda la organización.

Etiquetado de subredes para el modo automático de EKS

Si utiliza la capacidad de equilibrio de carga del modo automático de EKS, debe agregar etiquetas de AWS a las subredes de la VPC.

Introducción

Estas etiquetas identifican las subredes como asociadas al clúster y, lo que es más importante, si la subred es pública o privada.

Las subredes públicas tienen acceso directo a Internet a través de una puerta de enlace de Internet. Se utilizan para recursos que deben ser de acceso público, como los equilibradores de carga.

Las subredes privadas no tienen acceso directo a Internet y utilizan puertas de enlace de NAT para el tráfico saliente. Se utilizan para recursos internos, como los nodos de EKS, que no necesitan direcciones IP públicas.

Para obtener más información sobre las puertas de enlace de NAT y las puertas de enlace de Internet, consulte Conectar la VPC a otras redes en la Guía del usuario de Amazon Virtual Private Cloud (VPC).

Requisito

En este momento, las subredes que utiliza el modo automático de EKS para equilibrar la carga deben tener una de las siguientes etiquetas.

Subredes públicas

Las subredes públicas se utilizan para los equilibradores de carga expuestos a Internet. Estas subredes deben tener las siguientes etiquetas:

Clave	Valor
`kubernetes.io/role/elb`	`1` o ``

Subredes privadas

Las subredes privadas se utilizan para los equilibradores de carga internos. Estas subredes deben tener las siguientes etiquetas:

Clave	Valor
`kubernetes.io/role/internal-elb`	`1` o ``

Procedimiento

Antes de empezar, identifique qué subredes son públicas (con acceso a través de puertas de enlace de Internet) y cuáles son privadas (a través de puertas de enlace de NAT). Necesitará permisos para modificar los recursos de la VPC.

Consola de administración de AWS

1. Abra la consola de Amazon VPC y, a continuación, Subredes.
2. Seleccione la subred que se etiquetará.
3. Elija la pestaña Etiquetas y, a continuación, Agregar etiqueta.
4. Agregue la etiqueta adecuada:
   - Para subredes públicas: Clave=`kubernetes.io/role/elb`

- Para subredes privadas: Clave=`kubernetes.io/role/internal-elb`

5. Defina el valor en 1 o déjelo en blanco.
6. Guarde y repita el procedimiento para el resto de las subredes.

AWS CLI

Para subredes públicas:

```
aws ec2 create-tags \
    --resources subnet-ID \
    --tags Key=kubernetes.io/role/elb,Value=1
```

Para subredes privadas:

```
aws ec2 create-tags \
    --resources subnet-ID \
    --tags Key=kubernetes.io/role/internal-elb,Value=1
```

Sustituya `subnet-ID` por su ID de subred real.

Generación de informes de cumplimiento con el CIS desde los nodos de Kubernetes mediante la depuración de kubectl

En esta sección, se describe cómo generar informes de cumplimientos con el CIS (Center for Internet Security) para los nodos de Amazon EKS mediante el comando `kubectl debug`. El comando le permite crear temporalmente un contenedor de depuración en un nodo de Kubernetes y ejecutar comprobaciones de cumplimiento con el CIS mediante la herramienta `apiclient`. La herramienta `apiclient` forma parte del sistema operativo Bottlerocket, utilizado por los nodos del modo automático de EKS.

Requisitos previos

Antes de comenzar, asegúrese de que dispone de lo siguiente:

- Acceso a un clúster de Amazon EKS con `kubectl` configurado (la versión debe ser al menos la v1.32.0; escriba `kubectl version` para comprobarlo).
- Los permisos de IAM adecuados para depurar los nodos.

- Un perfil válido que permita realizar operaciones de depuración (por ejemplo, `sysadmin`).

Para obtener más información sobre el uso de perfiles de depuración con `kubectl`, consulte Depuración de un pod o nodo durante la aplicación de un perfil en la documentación de Kubernetes.

Procedimiento

1. Determine el ID de instancia de AWS del nodo en el que quiere ejecutar el informe. Use el siguiente comando para enumerar todos los nodos en el clúster. El ID de instancia se encuentra en la columna de nombre y comienza por `i-`:

```
kubectl get nodes
```

```
NAME                  STATUS   ROLES    AGE   VERSION
i-0ea0ba0f8ef9ad609   Ready    <none>   62s   v1.30.10-eks-1a9dacd
```

2. Ejecute el siguiente comando y reemplace `<instance-id>` con el ID de instancia del nodo que desea consultar:

```
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"
```

   Los componentes de este comando incluyen los siguientes:

   - `kubectl debug node/<instance-id>`: crea una sesión de depuración en el ID de instancia EC2 especificado.
   - `-it`: asigna un TTY (intérprete de línea de comandos) y mantiene stdin abierto para su uso interactivo.
   - `--profile=sysadmin`: utiliza el perfil `kubectl` especificado con los permisos adecuados.
   - `--image=public.ecr.aws/amazonlinux/amazonlinux:2023`: utiliza `amazonlinux:2023` como imagen contenedora para la depuración.
   - `bash -c "…"` : ejecuta los siguientes comandos en un intérprete de comandos bash:
     - `yum install -q -y util-linux-core`: instala silenciosamente el paquete de utilidades necesario.
     - `nsenter -t 1 -m`: ejecuta `nsenter` para introducir el espacio de nombres del proceso del host (PID 1).

- `apiclient report cis --level 1 --format text`: ejecuta el informe de conformidad con el CIS en el nivel 1 con salida de texto.

3. Revise el resultado de texto del informe.

Interpretación del resultado

El comando genera un informe basado en texto que muestra el estado de cumplimiento de varios controles del CIS. El resultado incluye lo siguiente:

- ID de control del CIS individual
- Descripción de cada control
- Estado de aprobación, rechazo u omisión de cada comprobación
- Detalles que explican cualquier problema de cumplimiento

Este es un ejemplo del resultado del informe ejecutado en una instancia de Bottlerocket:

```
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
```

Para obtener más información sobre los valores de referencia, consulte Kubernetes Benchmark del Center for Internet Security (CIS).

Recursos relacionados

- Bottlerocket CIS Benchmark en la documentación del sistema operativo Bottlerocket.
- Depurar pods en ejecución en la documentación de Kubernetes.
- Kubernetes Benchmark del Center for Internet Security (CIS)

Habilitación del cifrado de volúmenes de EBS con claves de KMS administradas por el cliente para el modo automático de EKS

Puede cifrar el volumen raíz efímero de las instancias en el modo automático de EKS con una clave de KMS administrada por el cliente.

El modo automático de Amazon EKS utiliza roles vinculados a servicios para delegar permisos a otros servicios de AWS al administrar volúmenes de EBS cifrados para los clústeres de Kubernetes. Este tema describe cómo configurar la política de claves que necesita al especificar una clave administrada por el cliente para el cifrado de Amazon EBS con el modo automático de EKS.

Consideraciones:

- El modo automático de EKS no necesita autorización adicional para usar la clave administrada por AWS predeterminada para proteger los volúmenes cifrados en la cuenta.
- Este tema aborda el cifrado de volúmenes efímeros, es decir, los volúmenes raíz de las instancias de EC2. Para obtener más información sobre el cifrado de volúmenes de datos utilizados para cargas de trabajo, consulte the section called “Creación de una StorageClass”.

Descripción general

Las siguientes claves de AWS KMS se pueden utilizar para el cifrado del volumen raíz de Amazon EBS cuando el modo automático de EKS lanza instancias:

- Clave administrada por AWS: una clave de cifrado en la cuenta que Amazon EBS crea, posee y administra. Esta es la clave de cifrado predeterminada en las cuentas nuevas.
- Clave administrada por el cliente: una clave de cifrado personalizada que usted crea, posee y administra.

> **Note**
>
> La clave debe ser simétrica. Amazon EBS no es compatible con claves asimétricas administradas por el cliente.

Paso 1: configuración de la política de claves

Las claves de KMS deben tener una política de claves que permita al modo automático de EKS lanzar instancias con volúmenes de Amazon EBS cifrados mediante una clave administrada por el cliente.

Configure la política de claves con la siguiente estructura:

> **Note**
>
> Esta política solo incluye permisos para el modo automático de EKS. Es posible que la política de claves necesite permisos adicionales si otras identidades deben usar la clave o administrar concesiones.

```
{
    "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"
                }
            }
        }
    ]
}
```

Asegúrese de reemplazar <account-id> con el ID real de cuenta de AWS.

Al configurar la política de claves:

- El ClusterServiceRole debe tener los permisos necesarios de IAM para usar la clave de KMS en operaciones de cifrado.

- La condición `kms:GrantIsForAWSResource` garantiza que las concesiones solo se puedan crear para servicios de AWS.

Paso 2: configuración de NodeClass con la clave administrada por el cliente

Después de configurar la política de claves, haga referencia a la clave de KMS en la configuración de NodeClass del modo automático de EKS:

```
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>"
```

Sustituya los valores de marcador de posición por los valores reales.

- `<region>` por la región de AWS
- `<account-id>` por el ID de cuenta de AWS
- `<key-id>` por el ID de la clave de KMS

Puede especificar la clave de KMS con cualquiera de los siguientes formatos:

- ID de clave de KMS: `1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d`
- ARN de clave de KMS: `arn:aws:kms:us-west-2:111122223333:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d`
- Nombre del alias de la clave: `alias/eks-auto-mode-key`
- ARN del alias de la clave: `arn:aws:kms:us-west-2:111122223333:alias/eks-auto-mode-key`

Use kubectl para aplicar la configuración de NodeClass:

```
kubectl apply -f nodeclass.yaml
```

Activos relacionados

- Creación de una clase de nodo para Amazon EKS
- Consulte más información en la Guía para desarrolladores de AWS Key Management Service
  - Permisos para servicios de AWS en políticas de claves
  - Cómo cambiar una política de claves
  - Concesiones en AWS KMS

Actualización de los controles de organización para el modo automático de EKS

Algunos controles de organización pueden impedir que el modo automático de EKS funcione correctamente. Si es así, debe actualizar estos controles para permitir que el modo automático de EKS disponga de los permisos necesarios para administarr las instancias de EC2 en su nombre.

El modo automático de EKS utiliza un rol de servicio para ejecutar las instancias de EC2 que respaldan los nodos del modo automático de EKS. Un rol de servicio es un rol de IAM que se crea en su cuenta y que un servicio asume para realizar acciones en su nombre. Las políticas de control de servicio (SCP) siempre se aplican a las acciones realizadas con los roles de servicio. Esto permite que una SCP inhiba las operaciones del modo automático. Lo más común es cuando se utiliza una SCP para restringir las imágenes de máquina de Amazon (AMI) que se pueden ejecutar. Para permitir el funcionamiento del modo automático de EKS, modifique la SCP para permitir la ejecución de las AMI desde las cuentas del modo automático de EKS.

También puede utilizar la característica de AMI permitidas por EC2 para limitar la visibilidad de las AMI en otras cuentas. Si utiliza esta característica, debe ampliar los criterios de imagen para incluir también las cuentas de AMI del modo automático de EKS en las regiones de interés.

Ejemplo de SCP para bloquear todas las AMI excepto las AMI del modo automático de EKS

La siguiente SCP impide realizar llamadas a `ec2:RunInstances` a menos que la AMI pertenezca a la cuenta de AMI del modo automático de EKS para us-west-2 o us-east-1.

> **Note**
>
> Es importante no utilizar la clave de contexto `ec2:Owner`. Amazon es propietario de las cuentas AMI del modo automático de EKS y el valor de esta clave siempre será `amazon`. Crear una SCP que permita ejecutar AMI si `ec2:Owner` es `amazon` permitirá ejecutar cualquier AMI propiedad de Amazon, no solo las del modo automático de EKS.*

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAMI",
      "Effect": "Deny",
      "Action": "ec2:RunInstances",
      "Resource": "arn:*:ec2:*::image/ami-*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "767397842682",
            "992382739861"
          ]
        }
      }
    }
  ]
}
```

Cuentas de AMI de modo automático de EKS

Cuentas de AWS que varían según la región alojan las AMI públicas del modo automático de EKS.

Región de AWS	Cuenta
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

Asociación de direcciones IP públicas

Cuando se llama a `ec2:RunInstances`, el campo `AssociatePublicIpAddress` para el lanzamiento de una instancia se determina automáticamente en función del tipo de subred en la que se lanza la instancia. Se puede usar una SCP para exigir que este valor se establezca explícitamente en falso, independientemente del tipo de subred en la que se lance. En este caso, el campo `spec.advancedNetworking.associatePublicIPAddress` de NodeClass también se puede establecer en falso para cumplir con los requisitos de la SCP.

```
{
      "Sid": "DenyPublicEC2IPAddesses",
      "Effect": "Deny",
      "Action": "ec2:RunInstances",
      "Resource": "arn:aws:ec2:*:*:network-interface/*",
      "Condition": {
          "BoolIfExists": {
              "ec2:AssociatePublicIpAddress": "true"
          }
      }
  }
```

Control de la implementación de las cargas de trabajo en las reservas de capacidad con el modo automático de EKS

Puede controlar la implementación de las cargas de trabajo en las reservas de capacidad. El modo automático de EKS admite reservas de capacidad bajo demanda (ODCR) de EC2 y bloques de capacidad de EC2 para ML.

> **Tip**
>
> De forma predeterminada, el modo automático de EKS se lanza automáticamente en las ODCR abiertas y los bloques de capacidad de ML. Cuando se usa `capacityReservationSelectorTerms` en la definición de NodeClass, el modo automático de EKS dejará de utilizar automáticamente las reservas de capacidad abiertas.

Reservas de capacidad bajo demanda (ODCR) de EC2

Las reservas de capacidad bajo demanda (ODCR) de EC2 le permiten reservar capacidad de computación para sus instancias de Amazon EC2 en una zona de disponibilidad específica por cualquier período. Cuando utilice el modo automático de EKS, le recomendamos controlar si sus cargas de trabajo de Kubernetes se implementan en estas instancias reservadas para maximizar la utilización de la capacidad previamente adquirida o para garantizar que las cargas de trabajo críticas tengan acceso a los recursos garantizados.

De forma predeterminada, el modo automático de EKS se inicia automáticamente en las ODCR abiertas. Sin embargo, cuando configura `capacityReservationSelectorTerms` en una NodeClass, puede controlar de forma explícita cuáles ODCR utilizan sus cargas de trabajo. Los nodos aprovisionados mediante ODCR configuradas tendrán `karpenter.sh/capacity-type: reserved` y se priorizarán frente a los nodos bajo demanda y nodos de spot. Una vez habilitada esta característica, el modo automático de EKS ya no utilizará automáticamente las ODCR abiertas; deben seleccionarse explícitamente mediante una NodeClass, lo que le permite controlar con precisión el uso de las reservas de capacidad en todo el clúster.

> **Warning**
>
> Si configura `capacityReservationSelectorTerms` en una NodeClass de un clúster, el modo automático de EKS ya no utilizará automáticamente las ODCR abiertas para ninguna NodeClass del clúster.

Ejemplo de 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"
```

Este ejemplo de NodeClass muestra dos enfoques para seleccionar las ODCR. El primer método hace referencia directamente a una ODCR específica mediante su ID (`cr-56fac701cc1951b03`). El segundo método utiliza la selección basada en etiquetas y se dirige a las ODCR con la etiqueta `Name: "targeted-odcr"`. También puede filtrar opcionalmente por la cuenta de AWS propietaria de la reserva, lo cual es particularmente útil en escenarios de varias cuentas o cuando se trabaja con reservas de capacidad compartida.

Bloques de capacidad de EC2 para ML

Los bloques de capacidad para ML reservan instancias de computación acelerada basadas en GPU en una fecha futura para admitir cargas de trabajo de machine learning (ML) de corta duración. Las instancias que se ejecutan en un bloque de capacidad se colocan automáticamente cerca dentro de Amazon EC2 UltraClusters para conseguir redes que no generen bloqueos, de escala de petabits y de baja latencia.

Para obtener más información acerca de las plataformas y los tipos de instancias compatibles, consulte Bloques de capacidad para ML en la Guía del usuario de EC2.

Puede crear una NodeClass del modo automático EKS que utilice un bloque de capacidad para ML, similar a una ODCR (descrita anteriormente).

Los siguientes ejemplos de definiciones crean tres recursos:

1. Una NodeClass que hace referencia a la reserva del bloque de capacidad
2. Un NodePool que usa la NodeClass y aplica una taint
3. Una especificación de pod que tolera la taint y solicita recursos de la GPU

Ejemplo de NodeClass

Esta NodeClass hace referencia a un bloque de capacidad específico para ML mediante su ID de reserva. Puede obtener este ID en la consola de EC2.

```
apiVersion: eks.amazonaws.com/v1
kind: NodeClass
metadata:
  name: gpu
spec:
  # Specify your Capacity Block reservation ID
  capacityReservationSelectorTerms:
    - id: cr-56fac701cc1951b03
```

Para obtener más información, consulte the section called “Cómo crear una clase de nodos”.

NodePool de muestra

Este NodePool hace referencia a la NodeClass `gpu` y especifica una configuración importante:

- Solo usa la capacidad reservada mediante la configuración `karpenter.sh/capacity-type: reserved`.
- Solicita familias de instancias de GPU específicas adecuadas para las cargas de trabajo de ML.
- Aplica una taint `nvidia.com/gpu` para garantizar que solo las cargas de trabajo de GPU estén programadas en estos nodos

```
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
```

Para obtener más información, consulte the section called “Creación de un grupo de nodos”.

Pod de ejemplo

Este pod de ejemplo muestra cómo configurar una carga de trabajo para que se ejecute en los nodos del bloque de capacidad:

- Utiliza un nodeSelector para usar como destino tipos de GPU específicos (en este caso, GPU H200).
- Incluye una tolerancia a la taint `nvidia.com/gpu` aplicada por el NodePool.
- Solicita explícitamente los recursos de la GPU mediante el tipo de recurso `nvidia.com/gpu`.

```
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
```

Para obtener más información, consulte Pods en la documentación de Kubernetes.

Activos relacionados

- Bloques de capacidad para ML en la Guía del usuario de Amazon EC2
- Búsqueda y compra de bloques de capacidad en la Guía del usuario de Amazon EC2
- Administración de los recursos computacionales para las cargas de trabajo de IA/ML en Amazon EKS

- Optimización de recursos de GPU y administración de costos en la Guía de prácticas recomendadas de EKS

Implementación de nodos del modo automático de EKS en Zonas locales

El modo automático de EKS proporciona una administración de clústeres simplificada con aprovisionamiento automático de nodos. Las Zonas locales de AWS amplían la infraestructura de AWS a ubicaciones geográficas más cercanas a los usuarios finales, lo que reduce la latencia de las aplicaciones sensibles a la latencia. En esta guía, obtendrá los pasos del proceso de implementación de nodos del modo automático de EKS en Zonas locales de AWS, lo que le permite ejecutar aplicaciones en contenedores con una latencia más baja para los usuarios de áreas geográficas específicas.

Además, también se muestra cómo utilizar taints y tolerancias de Kubernetes para garantizar que solo se ejecuten cargas de trabajo específicas en los nodos de Zonas locales, lo que lo ayuda a controlar los costos y optimizar el uso de recursos.

Requisitos previos

Antes de comenzar a implementar los nodos del modo automático de EKS en Zonas Locales, asegúrese de cumplir los siguientes requisitos previos.

- Un clúster del modo automático de EKS existente
- Uso de la Zona local en su cuenta de AWS

Paso 1: creación de la subred de la Zona local

El primer paso para implementar los nodos del modo automático de EKS en una Zona local es crear una subred en esa Zona local. Esta subred proporciona la infraestructura de red para los nodos y les permite comunicarse con el resto de la VPC. Siga las instrucciones de Creación de una subred de Zona local (en la Guía del usuario de Zonas locales de AWS) para crear una subred en la Zona local que elija.

> **Tip**
>
> Anote el nombre de la subred de la Zona local.

Paso 2: creación de NodeClass para la subred de la Zona local

Después de crear la subred de la zona local, debe definir una NodeClass que haga referencia a esta subred. NodeClass es un recurso personalizado de Kubernetes que especifica los atributos de infraestructura de los nodos, lo que incluye las subredes, los grupos de seguridad y las configuraciones de almacenamiento que se deben utilizar. En el siguiente ejemplo, creamos una NodeClass llamada “local-zone” que tiene como destino una subred de la Zona local basada en su nombre. También puede usar el ID de la subred. Deberá adaptar esta configuración para que su destino sea la subred de la Zona local.

Para obtener más información, consulte the section called “Cómo crear una clase de nodos”.

```
apiVersion: eks.amazonaws.com/v1
kind: NodeClass
metadata:
  name: local-zone
spec:
  subnetSelectorTerms:
    - id: <local-subnet-id>
```

Paso 3: creación de NodePool con NodeClass y Taint

Con la NodeClass configurada, ahora debe crear un NodePool que use esta NodeClass. Un NodePool define las características de computación de los nodos, lo que incluye los tipos de instancias. El NodePool usa la NodeClass como referencia para determinar dónde se lanzarán las instancias.

En el siguiente ejemplo, creamos un NodePool que hace referencia a nuestra NodeClass “local-zone”. También agregamos una taint a los nodos para asegurarnos de que solo se puedan programar los pods con una tolerancia coincidente en estos nodos de la Zona local. Esto es especialmente importante para los nodos de la Zona local, que suelen tener costos más elevados y solo deberían utilizarlos las cargas de trabajo que se beneficien específicamente de la reducción de la latencia.

Para obtener más información, consulte the section called “Creación de un grupo de nodos”.

```
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"]
```

La taint con la clave `aws.amazon.com/local-zone` y el efecto `NoSchedule` garantiza que los pods que no tengan una tolerancia igual no se programen en estos nodos. De este modo, se evita que las cargas de trabajo normales se ejecuten accidentalmente en la Zona local, lo que podría generar costos inesperados.

Paso 4: implementación de cargas de trabajo con tolerancia y afinidad de nodos

Para tener un control óptimo de la ubicación de la carga de trabajo en los nodos de la Zona local, utilice simultáneamente taints y tolerancias con la afinidad de nodos. A continuación se enumeran las ventajas de este enfoque combinado:

1. Control de costos: esta taint garantiza que solo los grupos con tolerancias explícitas puedan utilizar recursos de la Zona local que podrían resultar costosos.
2. Ubicación garantizada: la afinidad de nodos garantiza que las aplicaciones sensibles a la latencia se ejecuten exclusivamente en la Zona local, no en los nodos normales de un clúster.

A continuación se muestra un ejemplo de una implementación configurada para ejecutarse específicamente en los nodos de la Zona local:

```
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"
```

Esta implementación tiene dos configuraciones de programación clave:

1. La tolerancia permite programar los pods en los nodos con la taint `aws.amazon.com/local-zone`.

2. El requisito de afinidad de nodos garantiza que estos pods solo se ejecuten en los nodos con la etiqueta `node-type: local-zone`.

En conjunto, garantizan que la aplicación sensible a la latencia se ejecute solo en los nodos de la Zona local y que las aplicaciones normales no consuman los recursos de la Zona local a menos que estén configuradas explícitamente para hacerlo.

Paso 5: verificar con la consola de AWS

Tras configurar la NodeClass, el NodePool y las implementaciones, debe comprobar que los nodos se aprovisionen en la Zona local según lo previsto y que las cargas de trabajo se ejecuten en ellos. Puede usar la Consola de administración de AWS para comprobar que las instancias de EC2 se estén lanzando en la subred de la Zona local correcta.

Además, puede consultar la lista de nodos de Kubernetes que utilizan `kubectl get nodes -o wide` para confirmar que los nodos se unen al clúster con las etiquetas y los caracteres correctos:

```
kubectl get nodes -o wide
kubectl describe node <node-name> | grep -A 5 Taints
```

También puede verificar que los pods de las cargas de trabajo estén programados en los nodos de la Zona local:

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

Este enfoque garantiza que solo las cargas de trabajo que toleren específicamente la taint de la Zona local se programen en estos nodos, lo que lo ayuda a controlar los costos y a hacer un uso más eficiente de los recursos de la Zona local.

Más información sobre cómo funciona el modo automático de EKS

Utilice este capítulo para aprender cómo funcionan los componentes de los clústeres del modo automático de Amazon EKS.

Temas

- Más información sobre las instancias administradas del modo automático de Amazon EKS
- Más información sobre las identidades y el acceso en el modo automático de EKS

- Más información sobre las redes de VPC y el equilibrio de carga en el modo automático de EKS

Más información sobre las instancias administradas del modo automático de Amazon EKS

En este tema se explica la forma en que el modo automático de Amazon EKS administra las instancias de Amazon EC2 en el clúster de EKS. Al habilitar el modo automático de EKS, EKS aprovisiona y administra automáticamente los recursos de computación del clúster, lo que cambia la forma en que se interactúa con las instancias de EC2 que se desempeñan como nodos en el clúster.

Comprender la forma en que el modo automático de Amazon EKS administra las instancias es esencial para planificar la estrategia de implementación de cargas de trabajo y los procedimientos operativos. A diferencia de las instancias de EC2 tradicionales o los grupos de nodos administrados, estas instancias siguen un modelo de ciclo de vida diferente en el que EKS asume la responsabilidad de varios aspectos operativos, a la vez que restringe ciertos tipos de acceso y personalización.

El modo automático de Amazon EKS automatiza las tareas rutinarias de creación de nuevas instancias de EC2, a las que asocia como nodos al clúster de EKS. El modo automático de EKS detecta cuándo una carga de trabajo no cabe en los nodos existentes y crea una nueva instancia de EC2.

El modo automático de Amazon EKS se encarga de crear y eliminar instancias de EC2, así como de aplicarles revisiones. Usted es responsable de los contenedores y pods implementados en la instancia.

Las Instancias de EC2 creadas por el modo automático de EKS son diferentes de otras instancias de EC2, ya que se trata de instancias administradas. Estas instancias administradas son propiedad de EKS y tienen más restricciones. No puede acceder directamente ni instalar software en instancias administradas por el modo automático de EKS.

AWS sugiere ejecutar el modo automático de EKS o el Karpenter autoadministrado. Puede instalar ambos durante una migración o en una configuración avanzada. Si ambos están instalados, configure los grupos de nodos de modo que las cargas de trabajo se asocien a Karpenter o al modo automático de EKS.

Para obtener más información, consulte Instancias administradas de Amazon EC2 en la Guía del usuario de Amazon EC2.

Tabla de comparación

Instancia de EC2 estándar	Instancia administrada del modo automático de EKS
Usted es responsable de aplicar revisiones a la instancia y de actualizarla.	AWS aplica revisiones y actualizaciones a la instancia de forma automática.
EKS no se hace responsable del software presente en la instancia.	EKS es responsable de cierto software en la instancia, como `kubelet`, el tiempo de ejecución del contenedor y el sistema operativo .
Puede eliminar la Instancia de EC2 mediante la API de EC2.	EKS determina la cantidad de instancias implementadas en la cuenta. Si elimina una carga de trabajo, EKS reducirá la cantidad de instancias en la cuenta.
Puede utilizar SSH para acceder a la Instancia de EC2.	Puede implementar pods y contenedores en la instancia administrada.
Usted determina el sistema operativo y la imagen (AMI).	AWS determina el sistema operativo y la imagen
Puede implementar cargas de trabajo que se basen en la funcionalidad de Windows o Ubuntu.	Puede implementar contenedores basados en Linux, pero sin dependencias específicas del sistema operativo.
Usted determina qué tipo de instancia y familia lanzar.	AWS determina qué tipo de instancia y familia lanzar. Puede utilizar un grupo de nodos para limitar los tipos de instancias de los que el modo automático de EKS puede seleccionar.

La siguiente funcionalidad funciona tanto para instancias administradas como para instancias de EC2 estándar:

- Puede ver la instancia en la consola de AWS.

- Puede usar el almacenamiento de instancias como almacenamiento efímero para las cargas de trabajo.

Soporte de AMI

Con el modo automático de EKS, AWS determina la imagen (AMI) utilizada para los nodos de computación. AWS supervisa el despliegue de las nuevas versiones de la AMI del modo automático de EKS. Si tiene problemas de carga de trabajo relacionados con una versión de la AMI, cree un caso de soporte. Para obtener más información, consulte Creating support cases and case management en la Guía del usuario de AWS Support.

Por lo general, EKS lanza una nueva AMI cada semana que contiene correcciones de seguridad y CVE.

Referencia de instancias compatibles con el modo automático de EKS

El modo automático de EKS solo crea instancias de los tipos compatibles y que cumplen con un requisito de tamaño mínimo.

El modo automático de EKS admite los siguientes tipos de instancias:

Familia	Tipos de instancias
Optimizadas para la computación (C)	c8g, c7a, c7g, c7gn, c7gd, c7i, c7i-flex, c6a, c6g, c6i, c6gn, c6id, c6in, c6gd, c5, c5a, c5d, c5ad, c5n, c4
Uso general (M)	m8g, m7i, m7a, m7g, m7gd, m7i-flex, m6a, m6i, m6in, m6g, m6idn, m6id, m6gd, m5, m5a, m5ad, m5n, m5dn, m5d, m5zn, m4
Optimizada para memoria (R)	r8g, r7a, r7iz, r7gd, r7i, r7g, r6a, r6i, r6id, r6in, r6idn, r6g, r6gd, r5, r5n, r5a, r5dn, r5b, r5ad, r5d, r4
Ampliable (T)	t4g, t3, t3a, t2
Memoria elevada (Z/X)	z1d, x8g, x2gd
Optimizadas para el	i8g, i7ie, i4g, i4i, i3, i3en, is4gen, d3, d3en, im4gn

Familia	Tipos de instancias
almacenamiento (I/D)	
Computación acelerada (P/G/Inf/Trn)	p5, p4d, p4de, p3, p3dn, gr6, g6, g6e, g5g, g5, g4dn, inf2, inf1, trn1, trn1n
Computación de alto rendimiento (X2)	x2iezn, x2iedn, x2idn

Además, el modo automático de EKS solo creará instancias de EC2 que cumplan los siguientes requisitos:

- Más de 1 CPU
- El tamaño de la instancia no es nano, micro ni pequeño

Para obtener más información, consulte Convenciones de nomenclatura de tipos de instancias de Amazon EC2.

Servicio de metadatos de instancias

- El modo automático de EKS aplica IMDSv2 con un límite de saltos de 1 de forma predeterminada, siguiendo las prácticas recomendadas de seguridad de AWS.
- Esta configuración predeterminada no se puede modificar en el modo automático.
- Para los complementos que normalmente requieren acceso al IMDS, proporcione parámetros (como la región de AWS) durante la instalación para evitar búsquedas en el IMDS. Para obtener más información, consulte the section called “Campos que puede personalizar”.
- Si un pod requiere acceso al IMDS cuando se ejecuta en modo automático, debe configurar el pod para que se ejecute con `hostNetwork: true`. Esto permite que el pod acceda directamente al servicio de metadatos de la instancia.
- Tenga en cuenta las implicaciones de seguridad a la hora de conceder a los pods acceso a los metadatos de la instancia.

Para obtener más información acerca del servicio de metadatos de instancias (IMDS) de Amazon EC2, consulte Configuración de las opciones del servicio de metadatos de instancias en la Guía del usuario de Amazon EC2.

Consideraciones

- Si el almacenamiento efímero configurado en NodeClass es más pequeño que el almacenamiento local NVMe de la instancia, el modo automático de EKS elimina la necesidad de realizar una configuración manual, ya que toma automáticamente las siguientes medidas:
  - Utiliza un volumen de datos de Amazon EBS más pequeño (20 GiB) para reducir los costos.
  - Formatea y configura el almacenamiento local NVMe para el uso efímero de datos. Esto incluye configurar una matriz RAID 0 si hay varias unidades NVMe.
- Cuando `ephemeralStorage.size` iguala o supera la capacidad de NVMe local, se producen las siguientes acciones:
  - El Modo automático omite el volumen pequeño de EBS.
  - Las unidades NVMe se exponen directamente a su carga de trabajo.
- El modo automático de Amazon EKS no admite las siguientes acciones del servicio de inyección de errores de AWS:
  - `ec2:RebootInstances`
  - `ec2:SendSpotInstanceInterruptions`
  - `ec2:StartInstances`
  - `ec2:StopInstances`
  - `ec2:TerminateInstances`
  - `ec2:PauseVolumeIO`
- El modo automático de Amazon EKS admite las acciones de pods de EKS del servicio de inyección de errores de AWS. Para obtener más información, consulte Administración de experimentos del servicio de inyección de errores y Uso de las acciones aws:eks:pod del FID de AWS en la Guía del usuario de AWS Resilience Hub.
- No es necesario instalar el `Neuron Device Plugin` en los nodos del modo automático de EKS.

  Si tiene otros tipos de nodos en el clúster, deberá configurar el complemento Neuron Device de modo que no se ejecute en nodos de modo automático. Para obtener más información, consulte the section called “Control de la implementación”.

Más información sobre las identidades y el acceso en el modo automático de EKS

En este tema se describen los roles y permisos de Identity and Access Management (IAM) que se necesitan para utilizar el modo automático de EKS. El modo automático de EKS utiliza dos roles de IAM principales: un rol de IAM de clúster y un rol de IAM del nodo. Estos roles funcionan conjuntamente con EKS Pod Identity y las entradas de acceso de EKS para proporcionar una administración de acceso completa para los clústeres de EKS.

Al configurar el modo automático de EKS, deberá establecer estos roles de IAM con permisos específicos que permitan a los servicios de AWS interactuar con los recursos del clúster. Esto incluye permisos para administrar recursos de computación, volúmenes de almacenamiento, equilibradores de carga y componentes de red. Comprender estas configuraciones de roles resulta esencial para el correcto funcionamiento y la seguridad del clúster.

En el modo automático de EKS, los roles de AWS IAM se asignan automáticamente a los permisos de Kubernetes a través de las entradas de acceso de EKS, por lo que no es necesario configurar manualmente ConfigMaps `aws-auth` ni vínculos personalizados. Al crear un nuevo clúster en modo automático, EKS crea automáticamente los permisos de Kubernetes correspondientes mediante entradas de acceso, lo que garantiza que los servicios de AWS y los componentes del clúster cuenten con los niveles de acceso adecuados en los sistemas de autorización de AWS y Kubernetes. Esta integración automatizada reduce la complejidad de la configuración y ayuda a evitar los problemas relacionados con los permisos que se producen por lo general al administrar clústeres de EKS.

Rol de IAM de clúster

El rol de IAM del clúster es un rol de AWS Identity and Access Management (IAM) que Amazon EKS utiliza para administrar los permisos de los clústeres de Kubernetes. Este rol concede a Amazon EKS los permisos necesarios para interactuar con otros servicios de AWS en nombre del clúster y se configura automáticamente con los permisos de Kubernetes mediante las entradas de acceso de EKS.

- Debe asociar las políticas de AWS IAM a este rol.
- El modo automático de EKS asigna permisos de Kubernetes a este rol automáticamente mediante entradas de acceso de EKS.
- Con el modo automático de EKS, AWS sugiere crear un único rol de IAM de clúster por cuenta de AWS.

- AWS sugiere asignar el nombre `AmazonEKSAutoClusterRole` a este rol.
- Este rol requiere permisos para varios servicios de AWS para administrar recursos, incluidos volúmenes de EBS, equilibradores de carga elásticos e instancias de EC2.
- La configuración sugerida para este rol incluye múltiples políticas de IAM administradas de AWS, relacionadas con las diferentes capacidades del modo automático de EKS.
  - `AmazonEKSComputePolicy`
  - `AmazonEKSBlockStoragePolicy`
  - `AmazonEKSLoadBalancingPolicy`
  - `AmazonEKSNetworkingPolicy`
  - `AmazonEKSClusterPolicy`

Para obtener más información sobre el rol de IAM de clúster y las políticas de IAM administradas por AWS, consulte:

- the section called " Políticas administradas de AWS"
- the section called "Rol de IAM de clúster"

Para obtener más información sobre el acceso a Kubernetes, consulte:

- the section called "Revisión de las políticas de acceso"

Rol de IAM de nodo

El rol de IAM del nodo es un rol de AWS Identity and Access Management (IAM) utilizado por Amazon EKS para administrar los permisos de los nodos de trabajo en los clústeres de Kubernetes. Este rol concede a las instancias de EC2 que se ejecutan como nodos de Kubernetes los permisos necesarios para interactuar con servicios y recursos de AWS, y se configura automáticamente con permisos RBAC de Kubernetes mediante entradas de acceso de EKS.

- Debe asociar las políticas de AWS IAM a este rol.
- El modo automático de EKS asigna permisos de RBAC de Kubernetes a este rol automáticamente mediante entradas de acceso de EKS.
- AWS sugiere asignar el nombre `AmazonEKSAutoNodeRole` a este rol.
- Con el modo automático de EKS, AWS sugiere crear un único rol de IAM del nodo por cuenta de AWS.

- Este rol tiene permisos limitados. Los permisos clave incluyen asumir un rol de Pod Identity y extraer imágenes de ECR.
- AWS sugiere las siguientes políticas de IAM administradas por AWS:
  - `AmazonEKSWorkerNodeMinimalPolicy`
  - `AmazonEC2ContainerRegistryPullOnly`

Para obtener más información sobre el rol de IAM de clúster y las políticas de IAM administradas por AWS, consulte:

- the section called “ Políticas administradas de AWS”
- the section called “Rol de IAM de nodo”

Para obtener más información sobre el acceso a Kubernetes, consulte:

- the section called “Revisión de las políticas de acceso”

Rol vinculado a servicios

Amazon EKS utiliza un rol vinculado al servicio (SLR) para determinadas operaciones. Un rol vinculado a servicios es un tipo único de rol de IAM que se encuentra vinculado directamente a Amazon EKS. Los roles vinculados a servicios se encuentran predefinidos por Amazon EKS e incluyen todos los permisos que el servicio requiere para llamar a otros servicios de AWS en su nombre.

AWS crea y configura automáticamente el SLR. Solo puede eliminar un SLR después de haber eliminado primero sus recursos relacionados. De esta forma, se protegen los recursos de Amazon EKS, ya que se evita que se puedan eliminar accidentalmente permisos de acceso a los recursos.

La política del SLR concede a Amazon EKS permisos para observar y eliminar componentes básicos de la infraestructura: recursos de EC2 (instancias, interfaces de red, grupos de seguridad), recursos de ELB (equilibradores de carga, grupos de destino), capacidades de CloudWatch (registro y métricas) y roles de IAM con prefijo “eks”. También permite la conexión en red de puntos de conexión privados mediante la asociación de VPC/zonas alojadas e incluye permisos para la supervisión de EventBridge y la limpieza de recursos etiquetados con EKS.

Para obtener más información, consulte:

- the section called “Política administrada de AWS: AmazonEKSServiceRolePolicy”
- the section called “Permisos de roles vinculados a servicios para Amazon EKS”

Etiquetas personalizadas de AWS para los recursos del modo automático de EKS

De forma predeterminada, las políticas administradas relacionadas con el modo automático de EKS no permiten aplicar etiquetas definidas por el usuario a los recursos de AWS aprovisionados en modo automático. Si desea aplicar etiquetas definidas por el usuario a los recursos de AWS, debe asociar permisos adicionales al rol de IAM del clúster con permisos suficientes para crear y modificar etiquetas en los recursos de AWS. A continuación, aparece un ejemplo de política que permitirá el acceso sin restricciones al etiquetado:

Ver ejemplo de política de etiquetas personalizadas

```
{
    "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}"
                }
            }
        }
    ]
}
```

Referencia de política de acceso

Para obtener más información sobre los permisos de Kubernetes utilizados por el modo automático de EKS, consulte the section called “Revisión de las políticas de acceso”.

Más información sobre las redes de VPC y el equilibrio de carga en el modo automático de EKS

En este tema se explica cómo configurar las características de red y equilibrio de carga de la nube privada virtual (VPC) en el modo automático de EKS. Aunque el modo automático de EKS administra la mayoría de los componentes de red automáticamente, aún puede personalizar ciertos aspectos de la configuración de red del clúster a través de los recursos `NodeClass` y las anotaciones del equilibrador de carga.

Al utilizar el modo automático de EKS, AWS administra la configuración de la interfaz de red de contenedores (CNI) de la VPC y el aprovisionamiento del equilibrador de carga de su clúster. Para influir en los comportamientos de red, puede definir objetos `NodeClass` y aplicar anotaciones específicas a los recursos de servicio y de ingreso, al tiempo que mantiene el modelo operativo automatizado que proporciona el modo automático de EKS.

Capacidad de redes

El modo automático de EKS tiene una nueva capacidad de red que gestiona las redes de nodos y pods. Para configurarla, puede crear un objeto `NodeClass` de Kubernetes.

Las opciones de configuración de la CNI de AWS VPC anterior no se aplicarán al modo automático de EKS.

Configuración de las redes con una **`NodeClass`**

El recurso `NodeClass` del modo automático de EKS permite personalizar ciertos aspectos de la capacidad de la red. A través de `NodeClass`, puede especificar selecciones de grupos de seguridad, controlar la colocación de nodos a través de subredes de la VPC, establecer políticas SNAT, configurar políticas de red y habilitar el registro de eventos de red. Este enfoque mantiene el modelo operativo automatizado del modo automático de EKS a la vez que ofrece flexibilidad a la hora de personalizar la red.

Puede utilizar `NodeClass` para:

- Seleccionar un grupo de seguridad para los nodos
- Controlar cómo se colocan los nodos en las subredes de la VPC
- Establecer la política SNAT del nodo en `random` o `disabled`
- Habilite las políticas de red de Kubernetes, que incluyen:

  - Establecer la política de red en Denegar de forma predeterminada o Permitir de forma predeterminada
  - Habilitar el registro de eventos de red en un archivo.
- Conectar los pods a diferentes subredes para aislar el tráfico de los pods del tráfico de los nodos.

Más información sobre la Creación de una NodeClass de Amazon EKS.

Consideraciones

El modo automático de EKS admite:

- Políticas de red de EKS.
- Las opciones `HostPort` y `HostNetwork` para los pods de Kubernetes.
- Nodos y pods en subredes públicas o privadas.
- Almacenamiento en caché de las consultas de DNS en el nodo.

El modo automático de EKS no admite:

- Grupos de seguridad por pod (SGPP).
- Redes personalizadas en `ENIConfig`. Puede colocar los pods en varias subredes o aislarlos exclusivamente del tráfico del nodo con the section called “Selección de subredes para los pods”.
- Configuraciones de IP en calentamiento, prefijos en calentamiento y ENI en calentamiento.
- Configuración mínima de destinos de la IP.
- Otras configuraciones admitidas por la CNI de VPC de AWS de código abierto.
- Configuraciones de políticas de red, como la personalización del temporizador conntrack (el valor predeterminado es 300s).
- Cómo exportar registros de eventos de red a CloudWatch.

Administración de recursos de red

El modo automático de EKS gestiona la administración de prefijos, direcciones IP e interfaces de red mediante la supervisión de los recursos de NodeClass para las configuraciones de red. El servicio realiza varias operaciones clave de forma automática:

Delegación de prefijos

El modo automático de EKS utiliza de forma predeterminada la delegación de prefijos (prefijos /28) para las redes de pods y mantiene un grupo de calentamiento predefinido de recursos de IP que se escala en función del número de pods programados. Cuando se detecta la fragmentación de la subred de los pods, el modo automático aprovisiona direcciones IP secundarias (/32). Gracias a este algoritmo de red de pods predeterminado, el modo automático calcula el número máximo de pods por nodo en función del número de ENI e IP que se admiten por tipo de instancia (con la suposición del peor de los casos de fragmentación). Para obtener más información sobre la cantidad máxima de ENI e IP que se admiten por tipo de instancia, consulte Máximo de direcciones IP por interfaz de red en la Guía del usuario de Amazon EC2. Las familias de instancias de nueva generación (Nitro v6 y versiones posteriores) suelen tener más ENI e IP por tipo de instancia, y el modo automático ajusta el cálculo del número máximo de pods en consecuencia.

En el caso de los clústeres IPv6, solo se utiliza la delegación de prefijos y el modo automático siempre utiliza un límite máximo de 110 pods por nodo.

Gestión de la recuperación

El servicio implementa un conjunto de recuperación para los prefijos o las direcciones IPv4 secundarias que ya no se utilizan. Una vez transcurrido el periodo de recuperación, estos recursos se devuelven a la VPC. Sin embargo, si los pods reutilizan estos recursos durante el periodo de recuperación, se restauran del conjunto de recuperación.

Compatibilidad con IPv6

Para los clústeres de IPv6, el modo automático de EKS proporciona un prefijo de IPv6 `/80` por nodo en la interfaz de red principal.

El servicio también garantiza una administración adecuada y la recopilación de elementos no utilizados de todas las interfaces de red.

Equilibrio de carga

Se configuran los equilibradores de carga elásticos de AWS aprovisionados por el modo automático de EKS mediante anotaciones en los recursos de servicio y de ingreso.

Para obtener más información, consulte the section called “Creación de una clase de ingreso” o the section called “Crear un servicio”.

Consideraciones para el equilibrio de carga con el modo automático de EKS

- El modo de asignación de destino predeterminado es el modo de IP, no el modo de instancia.

- El modo automático de EKS solo admite el modo de grupo de seguridad para los equilibradores de carga de red.
- AWS no admite la migración de los equilibradores de carga del controlador del equilibrador de carga de AWS autoadministrado a la administración mediante el modo automático de EKS.
- No se admite el campo `networking.ingress.ipBlock` en la especificación `TargetGroupBinding`.
- Si los nodos de trabajo utilizan grupos de seguridad personalizados (no un patrón de nomenclatura `eks-cluster-sg-`), el rol del clúster necesitará permisos de IAM adicionales. La política administrada por EKS predeterminada solo permite a EKS modificar los grupos de seguridad denominados `eks-cluster-sg-`. Sin permiso para modificar los grupos de seguridad personalizados, EKS no podrá agregar las reglas de ingreso necesarias que permitan que el tráfico del equilibrado de carga de aplicaciones y del equilibrador de carga de red (ALB/NLB) llegue a los pods.

Consideraciones de CoreDNS

El modo automático de EKS no utiliza la implementación tradicional de CoreDNS para proporcionar resolución de DNS dentro del clúster. En cambio, los nodos del modo automático utilizan CoreDNS que se ejecuta como un servicio del sistema directamente en cada nodo. Si está llevando a cabo la transición de un clúster tradicional al modo automático, puede eliminar la implementación de CoreDNS del clúster una vez que las cargas de trabajo se hayan trasladado a los nodos del modo automático.

> **Important**
>
> Si tiene previsto mantener un clúster con nodos del modo automático y nodos que no lo sean, debe retener la implementación de CoreDNS. Los nodos que no son del modo automático dependen de los pods de CoreDNS tradicionales para la resolución de DNS, ya que no pueden acceder al servicio de DNS del nodo que proporciona el modo automático.

Solución de problemas del modo automático de EKS

Con el modo automático de EKS, AWS asume más responsabilidad por las instancias de EC2 de la cuenta de AWS. EKS asume la responsabilidad del tiempo de ejecución del contenedor en los nodos, el sistema operativo en los nodos y determinados controladores. Incluye un controlador de almacenamiento en bloque, un controlador de equilibrio de carga y un controlador de computación.

Debe utilizar las API de Kubernetes y AWS para solucionar problemas en los nodos. Puede hacer lo siguiente:

- Utilice un recurso `NodeDiagnostic` de Kubernetes para recuperar registros de nodos mediante el the section called “Agente de supervisión de nodos”. Para ver más pasos, consulte the section called “Cómo obtener los registros de nodos”.
- Utilice el comando AWS de la CLI de `get-console-output` EC2 para recuperar la salida de la consola de los nodos. Para ver más pasos, consulte the section called “Obtenga la salida de la consola de una instancia administrada por EC2 mediante la CLI de AWS EC2”.
- Utilice contenedores de depuración de Kubernetes para recuperar registros de nodos. Para ver más pasos, consulte the section called “Obtenga los registros de los nodos mediante contenedores de depuración y la `kubectl` CLI”.

> **Note**
>
> El modo automático de EKS utiliza instancias administradas por EC2. No puede acceder directamente a las instancias administradas por EC2, ni siquiera mediante SSH.

Es posible que tenga los siguientes problemas con soluciones específicas para los componentes del modo automático de EKS:

- Los pods están atascados en el estado `Pending` y no están programados en los nodos del modo automático. Para obtener soluciones, consulte the section called “Solución de problemas de pods que no se pueden programar en el nodo del modo automático”.
- Instancias administradas de EC2 que no se unen al clúster como nodos de Kubernetes. Para obtener soluciones, consulte the section called “Solución de problemas de un nodo que no se une al clúster”.
- Errores y problemas con los `NodePools`, `PersistentVolumes` y `Services` que utilizan los controladores incluidos en el modo automático de EKS. Para obtener soluciones, consulte the section called “Solución de problemas de los controladores incluidos en el modo automático”.
- La seguridad mejorada de los pods impide compartir volúmenes entre pods. Para obtener soluciones, consulte the section called “Cómo compartir volúmenes entre pods”.

Puede utilizar los siguientes métodos para solucionar problemas de los componentes del modo automático de EKS:

- the section called “Obtenga la salida de la consola de una instancia administrada por EC2 mediante la CLI de AWS EC2”
- the section called “Obtenga los registros de los nodos mediante contenedores de depuración y la `kubectl` CLI”
- the section called “Consulte los recursos asociados al modo automático de EKS en la consola de AWS”
- the section called “Consulte los errores de IAM en la cuenta de AWS”
- the section called “Detección de problemas de conectividad de los nodos con el `VPC Reachability Analyzer`”

Agente de supervisión de nodos

El modo automático de EKS incluye el agente de supervisión de nodos de Amazon EKS. Puede utilizar este agente para ver información de solución de problemas y depuración relativa a los nodos. El agente de supervisión de nodos publica `events` de Kubernetes y `conditions` de nodos. Para obtener más información, consulte the section called “Estado de los nodos”.

Obtenga la salida de la consola de una instancia administrada por EC2 mediante la CLI de AWS EC2

Este procedimiento ayuda a solucionar problemas en el arranque o a nivel del kernel.

En primer lugar, debe determinar el ID de la instancia de EC2 asociada a la carga de trabajo. En segundo lugar, utilice AWS CLI para recuperar la salida de la consola.

1. Confirme que `kubectl` se encuentra instalado y conectado al clúster
2. (Opcional) Utilice el nombre de una implementación de Kubernetes para enumerar los pods asociados.

```
kubectl get pods -l app=<deployment-name>
```

3. Utilice el nombre del pod de Kubernetes para determinar el ID de instancia de EC2 del nodo asociado.

```
kubectl get pod <pod-name> -o wide
```

4. Utilice el ID de instancia de EC2 para recuperar la salida de la consola.

```
aws ec2 get-console-output --instance-id <instance id> --latest --output text
```

Obtenga los registros de los nodos mediante contenedores de depuración y la **kubectl** CLI

La forma recomendada de recuperar los registros de un nodo del modo automático de EKS es utilizar el recurso `NodeDiagnostic`. Para ver los pasos, consulte the section called “Cómo obtener los registros de nodos”.

Sin embargo, puede transmitir los registros activos desde una instancia mediante el comando `kubectl debug node`. Este comando lanza un nuevo pod en el nodo que desea depurar y que puede usar de forma interactiva.

1. Lance un contenedor de depuración. El siguiente comando usa `i-0123456789012345`6 para el ID de instancia del nodo, `-it` asigna `tty` y adjunta `stdin` para el uso interactivo y utiliza el perfil `sysadmin` del archivo kubeconfig.

   ```
   kubectl debug node/i-0123456789012345`6 -it --profile=sysadmin --image=public.ecr.aws/
   amazonlinux/amazonlinux:2023
   ```

   Un ejemplo de salida sería el siguiente.

   ```
   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. Desde el intérprete de comandos, ahora puede instalar `util-linux-core`, que proporciona el comando `nsenter`. Use `nsenter` para ingresar el espacio de nombres de montaje del PID 1 (`init`) en el host y ejecutar el comando `journalctl` para transmitir los registros desde el `kubelet`:

   ```
   yum install -y util-linux-core
   nsenter -t 1 -m journalctl -f -u kubelet
   ```

Por motivos de seguridad, la imagen del contenedor de Amazon Linux no instala muchos binarios de forma predeterminada. Puede usar el comando `yum whatprovides` a fin de identificar el paquete que debe instalarse para proporcionar un binario determinado.

```
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
```

Consulte los recursos asociados al modo automático de EKS en la consola de AWS

Puede utilizar la consola de AWS para ver el estado de los recursos asociados al clúster del modo automático de EKS.

- Volúmenes de EBS
  - Para ver los volúmenes del modo automático de EKS, busque la clave de etiqueta `eks:eks-cluster-name`
- Balanceadores de carga
  - Para ver los equilibradores de carga del modo automático de EKS, busque la clave de etiqueta `eks:eks-cluster-name`
- Instancias de EC2
  - Para ver las instancias del modo automático de EKS, busque la clave de etiqueta `eks:eks-cluster-name`

Consulte los errores de IAM en la cuenta de AWS

1. Vaya a la consola de CloudTrail
2. Seleccione “Historial de eventos” en el panel de navegación izquierdo
3. Aplique filtros de códigos de error:
   - AccessDenied
   - UnauthorizedOperation
   - InvalidClientTokenId

Busque errores relacionados con el clúster de EKS. Utilice los mensajes de error para actualizar las entradas de acceso de EKS, el rol de IAM del clúster o el rol de IAM del nodo. Es posible que tenga que asociar una nueva política a estos roles con permisos para el modo automático de EKS.

Solución de problemas de pods que no se pueden programar en el nodo del modo automático

Si el estado de los pods es `Pending` y no se programan en un nodo del modo automático, verifique si el manifiesto de implementación o el pod tiene `nodeSelector`. Si `nodeSelector` está presente, asegúrese de que use `eks.amazonaws.com/compute-type: auto` para programar en los nodos creados con el modo automático de EKS. Para obtener más información sobre las etiquetas de nodos utilizadas por el modo automático de EKS, consulte the section called “Control de la implementación”.

Solución de problemas de un nodo que no se une al clúster

El modo automático de EKS configura automáticamente las nuevas instancias de EC2 con la información correcta para que se unan al clúster, lo que incluye el punto de conexión del clúster y la autoridad de certificación (CA) del clúster. Sin embargo, es posible que estas instancias sigan sin unirse al clúster de EKS como nodo. Ejecute los siguientes comandos para identificar las instancias que no se unieron al clúster:

1. Ejecute `kubectl get nodeclaim` para buscar `NodeClaims` con `Ready = False`.

   ```
   kubectl get nodeclaim
   ```

2. Ejecute `kubectl describe nodeclaim <node_claim>` y busque en Estado cualquier problema que impida que el nodo se una al clúster.

```
kubectl describe nodeclaim <node_claim>
```

Mensajes de error comunes:

`Error getting launch template configs`

Es posible que reciba este error si configura etiquetas personalizadas en `NodeClass` con los permisos predeterminados del rol de IAM del clúster. Consulte the section called “Identidad y acceso”.

`Error creating fleet`

Es posible que exista algún problema de autorización al hacer la llamada `RunInstances` desde la API de EC2. Compruebe si hay errores en AWS CloudTrail y consulte en the section called “Rol de IAM del clúster en modo automático” los permisos de IAM requeridos.

Detección de problemas de conectividad de los nodos con el **`VPC Reachability Analyzer`**

> **Note**
>
> Se le cobrará por cada análisis que ejecute el Analizador de accesibilidad de VPC. Para obtener más información sobre precios, consulte Precios de Amazon VPC.

Una de las razones por las que una instancia no se unió al clúster es un problema de conectividad de red que le impide acceder al servidor de API. Para diagnosticar este problema, puede usar el Analizador de accesibilidad de VPC para efectuar un análisis de la conectividad entre un nodo que no puede unirse al clúster y el servidor de API. Necesitará dos datos:

- ID de instancia de un nodo que no puede unirse al clúster
- Dirección IP del punto de conexión del servidor de API de Kubernetes

Para obtener el ID de instancia, tendrá que crear una carga de trabajo en el clúster para que el modo automático de EKS lance una instancia de EC2. Esto también crea un objeto `NodeClaim` en el clúster que tendrá el ID de la instancia. Ejecute `kubectl get nodeclaim -o yaml` para imprimir

todas las `NodeClaims` del clúster. Cada `NodeClaim` contiene el ID de instancia como un campo y, de nuevo, en ProviderID:

```
kubectl get nodeclaim -o yaml
```

Un ejemplo de salida sería el siguiente.

```
    nodeName: i-01234567890123456
    providerID: aws:///us-west-2a/i-01234567890123456
```

Puede determinar el punto de conexión del servidor de API de Kubernetes mediante la ejecución de `kubectl get endpoint kubernetes -o yaml`. Las direcciones se encuentran en el campo de direcciones:

```
kubectl get endpoints kubernetes -o yaml
```

Un ejemplo de salida sería el siguiente.

```
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
```

Con estos dos datos, puede efectuar el análisis s. Primero, vaya al Analizador de accesibilidad de VPC en la Consola de administración de AWS.

1. Haga clic en “Crear y analizar ruta”.
2. Proporcione un nombre para el análisis (por ejemplo, “Error en la unión de nodos”).
3. En “Tipo de origen” seleccione “Instancias”.

4. Ingrese el ID de instancia del nodo que falla como “Origen”.
5. En “Destino de ruta”, seleccione “Dirección IP”.
6. Ingrese una de las direcciones IP del servidor de API como “Dirección de destino”.
7. Amplíe la sección “Configuración de encabezados de paquetes adicionales”.
8. Ingrese 443 como “Puerto de destino”.
9. Seleccione “Protocolo” como TCP si aún no está seleccionado.
10. Haga clic en “Crear y analizar ruta”.
11. El análisis puede tardar unos minutos en completarse. Si los resultados del análisis indican un error de accesibilidad, indicarán dónde se produjo el error en la ruta de la red para que pueda resolver el problema.

Cómo compartir volúmenes entre pods

Los nodos del modo automático de EKS se configuran con SELinux en modo de aplicación, lo que proporciona un mayor aislamiento entre los pods que se ejecutan en el mismo nodo. Cuando SELinux está habilitado, a la mayoría de los pods no privilegiados se les aplicará automáticamente su propia etiqueta de seguridad multicategoría (MCS). Esta etiqueta MCS es única para cada pod y está diseñada para garantizar que el proceso de un pod no pueda manipular el proceso de ningún otro pod o del host. Incluso si un pod etiquetado funciona como raíz y tiene acceso al sistema de archivos del host, no podrá manipular los archivos, hacer llamadas confidenciales del sistema desde el host, acceder al tiempo de ejecución del contenedor ni obtener la clave secreta del kubelet.

Por este motivo, es posible que tenga problemas al intentar compartir datos entre pods. Por ejemplo, una `PersistentVolumeClaim` con el modo de acceso `ReadWriteOnce` no permitirá que varios pods accedan al volumen simultáneamente.

Para habilitar este uso compartido entre pods, puede usar `seLinuxOptions` de los pods para configurar la misma etiqueta MCS en esos pods. En este ejemplo, asignamos las tres categorías `c123,c456,c789` al pod. Esto no entrará en conflicto con ninguna de las categorías que se asignen automáticamente a los pods en el nodo, ya que solo se les asignarán dos categorías.

```
securityContext:
  seLinuxOptions:
    level: "s0:c123,c456,c789"
```

Visualización de los eventos de Karpenter en los registros del plano de control

En el caso de los clústeres de EKS con los registros del plano de control habilitados, puede obtener información sobre las acciones y el proceso de toma de decisiones de Karpenter consultándolos. Esto puede resultar especialmente útil para solucionar problemas del modo automático de EKS relacionados con el aprovisionamiento, el escalado y la terminación de los nodos. Para ver los eventos relacionados con Karpenter, utilice la siguiente consulta de CloudWatch Logs Insights:

```
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
```

Esta consulta filtra eventos específicos relacionados con Karpenter en los registros de auditoría de kube-apiserver. Los eventos incluyen varios estados de interrupción, errores de programación, problemas de capacidad y problemas relacionados con los nodos. Al analizar estos registros, puede comprender mejor lo siguiente:

- Por qué Karpenter está tomando ciertas medidas.
- Cualquier problema que impida el aprovisionamiento, el escalado o la terminación adecuados de los nodos.
- Posibles problemas de capacidad o compatibilidad con los tipos de instancias.

- Eventos del ciclo de vida de los nodos, como interrupciones, expulsiones o terminaciones.

Para usar esta consulta, siga estos pasos:

1. Diríjase a la consola de CloudWatch.
2. Seleccione “Logs Insights” en el panel de navegación izquierdo.
3. Elija el grupo de registros para los registros del plano de control de su clúster de EKS.
4. Pegue la consulta en el editor de consultas.
5. Ajuste el intervalo de tiempo según sea necesario.
6. Ejecute la consulta

Los resultados le mostrarán una cronología de los eventos relacionados con Karpenter, lo que le ayudará a solucionar problemas y a comprender el comportamiento del modo automático EKS en su clúster. Para revisar las acciones de Karpenter en un nodo específico, puede añadir el siguiente filtro de línea que especifica el ID de la instancia a la consulta antes mencionada:

```
|filter @message like /[.replaceable]`i-12345678910123456`/
```

> **Note**
>
> Para utilizar esta consulta, el registro del plano de control debe estar habilitado en el clúster de EKS. Si aún no lo ha hecho, consulte the section called “Registros del plano de control”.

Solución de problemas de los controladores incluidos en el modo automático

Si surge un problema con un controlador, averigüe lo siguiente:

- Si los recursos asociados a ese controlador tienen el formato adecuado y son válidos.
- Si los recursos RBAC de Kubernetes y AWS IAM están configurados correctamente para el clúster. Para obtener más información, consulte the section called “Identidad y acceso”.

Recursos relacionados

Use estos artículos de AWS re:Post para obtener pasos avanzados de solución de problemas:

- ¿Cómo se solucionan los problemas de escalado más comunes en el modo automático de EKS?
- ¿Cómo soluciono los problemas de aprovisionamiento de grupos de nodos personalizados y clases de nodos en el modo automático de Amazon EKS?
- ¿Cómo soluciono los problemas de los grupos de nodos integrados en el modo automático de EKS con un estado desconocido?

Revisión de las notas de la versión del modo automático de EKS

En esta página, se documentan las actualizaciones del modo automático de Amazon EKS. Puede consultar periódicamente esta página para ver anuncios sobre características, correcciones de errores, problemas conocidos y funciones obsoletas.

Para recibir notificaciones de todos los cambios en el archivo de origen de esta página de documentación específica, puede suscribirse a la siguiente URL con un lector de RSS:

```
https://github.com/awsdocs/amazon-eks-user-guide/commits/mainline/latest/ug/automode/auto-change.adoc.atom
```

19 de noviembre de 2025

Característica: se activó el desempaquetado y la extracción en paralelo de Seekable OCI (SOCI) para instancias de las familias G, P y Trn con almacenamiento de NVMe local. El desempaquetado y la extracción en paralelo de SOCI siempre se usa para estas familias de instancias con el modo automático de EKS y no se requieren cambios en la configuración activarlo. Para obtener más información sobre SOCI, consulte el blog de lanzamiento.

19 de noviembre de 2025

Característica: se agregó soporte para grupos de nodos con capacidad estática que mantienen un número fijo de nodos. Para obtener más información, consulte the section called “Grupos de nodos con capacidad estática”.

23 de octubre de 2025

Característica: los usuarios con clústeres en las regiones de EE. UU. ahora pueden solicitar AMI compatibles con FIPS si especifican `spec.advancedSecurity.fips` en su definición de NodeClass.

1 de octubre de 2025

Característica: el modo automático de EKS ahora admite la implementación de nodos en Zonas locales de AWS. Para obtener más información, consulte the section called “Uso de Zonas locales”.

30 de septiembre de 2025

Característica: se agregó soporte para instanceProfile a la NodeClass `spec.instanceProfile` que se excluye mutuamente del campo `spec.role`.

29 de septiembre de 2025

Actualmente, el modo automático de EKS no admite DRA.

10 de septiembre de 2025

Tarea: los eventos activados desde el controlador de computación del modo automático ahora usarán el nombre `eks-auto-mode/compute` en lugar de `karpenter`.

24 de agosto de 2025

Corrección de errores: las VPC que utilizaban un conjunto de opciones de DHCP con un nombre de dominio personalizado que contenía letras en mayúscula hacían que los nodos no pudieran unirse al clúster debido a la generación de un nombre de host no válido. Este problema se ha resuelto y los nombres de dominio con letras en mayúscula ahora funcionan correctamente.

15 de agosto de 2025

Corrección de errores: el agente de Pod Identity ahora solo escucha en la dirección local del enlace IPv4 en un clúster de EKS IPv4 para evitar problemas en los que el pod no puede alcanzar la dirección IPv6.

6 de agosto de 2025

Característica: se agregó una nueva configuración en la NodeClass `spec.advancedNetworking.associatePublicIPAddress` que se puede usar para evitar que se asignen direcciones IP públicas a los nodos de modo automático de EKS.

30 de junio de 2025

Característica: la NodeClass del Modo automático ahora utiliza la clave de KMS personalizada configurada para cifrar el volumen raíz de solo lectura de la instancia, además del volumen de datos de lectura y escritura. Anteriormente, la clave de KMS personalizada solo se utilizaba para cifrar el volumen de datos.

20 de junio de 2025

Característica: compatibilidad para controlar la implementación de cargas de trabajo en reservas de capacidad bajo demanda (ODCR) de EC2. Esto añade la clave opcional `capacityReservationSelectorTerms` a la NodeClass, lo que le permite controlar explícitamente cuáles ODCR utilizan sus cargas de trabajo. Para obtener más información, consulte the section called “Control de las reservas de capacidad”.

13 de junio de 2025

Característica: compatibilidad con subredes de pods independientes en `NodeClass`. Esto agrega las claves opcionales `` `podSubnetSelectorTerms `` y `podSecurityGroupSelectorTerms` para configurar las subredes y los grupos de seguridad de los pods. Para obtener más información, consulte the section called “Selección de subredes para los pods”.

30 de abril de 2025

Característica: compatibilidad con proxys de red directa en el `NodeClass`. Esto agrega la clave opcional `advancedNetworking` para configurar el proxy HTTPS. Para obtener más información, consulte the section called “Especificación de clase de nodos”.

18 de abril de 2025

Característica: soporte para resolver dominios .local (normalmente reservado para DNS de multidifusión) mediante DNS de unidifusión.

11 de abril de 2025

Característica: se agregó `certificateBundles` y `ephemeralStorage.kmsKeyID` a `NodeClass`. Para obtener más información, consulte the section called “Especificación de clase de nodos”.

Característica: se mejoró la velocidad de extracción de imágenes, especialmente para los tipos de instancias con almacenamiento de instancias local que pueden aprovechar la descompresión de imágenes más rápida.

Corrección de errores: se resolvió una condición de carrera que provocaba el error FailedCreatePodSandBox. Al marcar tcp 127.0.0.1:50051: connect:, la conexión a veces fallaba para pods que se programaban en un nodo inmediatamente después del arranque.

4 de abril de 2025

Característica: aumento de `registryPullQPS` de 5 a 25 y de `registryBurst` de 10 a 50 para reducir la limitación de extracción de imágenes impuesta por el cliente (`Failed to pull image xyz: pull QPS exceeded`).

31 de marzo de 2025

Corrección de error: soluciona un problema por el que, si un pod de DNS principal se estaba ejecutando en un nodo de modo automático, las consultas de DNS de los pods del nodo llegaban a ese pod de DNS principal en lugar de llegar al servidor de DNS local del nodo. Las consultas de DNS de los pods de un nodo de modo automático siempre irán al DNS local del nodo.

21 de marzo de 2025

Corrección de error: los nodos del modo automático ahora resuelven `kube-dns.kube-system.svc.cluster.local` correctamente cuando no hay ningún servicio de `kube-dns` instalado en el clúster. Aborda el problema #2546 de GitHub.

14 de marzo de 2025

Característica: la salida de `IPv4` está habilitada en los clústeres `IPv6`. El tráfico de `IPv4` que sale de los clústeres `IPv6` del modo automático ahora se traducirá automáticamente a la dirección `v4` de la ENI principal del nodo.

Ciclo de vida y configuración del clúster de Amazon EKS

En esta sección, se proporciona una guía detallada sobre la administración del ciclo de vida completo de los clústeres de Kubernetes y las diferentes formas de configurarlos. Abarca la creación de clústeres, la supervisión de su estado, la actualización de las versiones de Kubernetes y la eliminación de clústeres. También se incluyen temas avanzados sobre cómo configurar el acceso a los puntos de conexión, activar o desactivar la compatibilidad con Windows, configurar clústeres privados, implementar el escalado automático y cómo mejorar la resiliencia con el cambio de zona para la redirección del tráfico en configuraciones multi-AZ.

Temas

- Cómo crear un clúster del modo automático de Amazon EKS
- Creación de un clúster de Amazon EKS
- Plano de control aprovisionado de Amazon EKS
- Preparación para las actualizaciones de las versiones de Kubernetes y solución de problemas de configuración incorrecta con información sobre clústeres
- Actualización del clúster existente a la nueva versión de Kubernetes
- Eliminar un clúster
- Punto de conexión del servidor de API del clúster
- Implementación de nodos de Windows en clústeres de EKS
- Deshabilitar la compatibilidad con Windows
- Implementación de clústeres privados con acceso limitado a Internet
- Escale la computación en clústeres con Karpenter y el Escalador automático de clústeres
- Información sobre el cambio de zona del Controlador de recuperación de aplicaciones de Amazon (ARC) en Amazon EKS
- Activación del cambio de zona de EKS para evitar zonas de disponibilidad deterioradas

Cómo crear un clúster del modo automático de Amazon EKS

En este tema se ofrecen instrucciones detalladas para crear un clúster del modo automático de Amazon EKS mediante opciones de configuración avanzadas. Abarca los requisitos previos, las opciones de red y las configuraciones de complementos. El proceso incluye la configuración de roles

de IAM, la configuración de los ajustes de los clústeres, la especificación de los parámetros de red y la selección de complementos. Los usuarios pueden crear clústeres a través de la Consola de administración de AWS o de la AWS CLI, con instrucciones paso a paso para ambos métodos.

Para los usuarios que busquen un proceso de configuración menos complejo, consulte los siguientes pasos simplificados para la creación de clústeres:

- the section called “CLI de eksctl”
- the section called “ AWS CLI”
- the section called “Consola de administración”

Esta guía de configuración avanzada está dirigida a usuarios que requieren un control más detallado sobre la configuración del clúster del modo automático de EKS y que están familiarizados con los conceptos y requisitos de Amazon EKS. Antes de continuar con la configuración avanzada, asegúrese de haber cumplido con todos los requisitos previos y de tener un conocimiento completo de los requisitos de red y IAM para los clústeres del modo automático de EKS.

El modo automático de EKS requiere permisos de IAM adicionales. Para obtener más información, consulte:

- the section called “Roles de IAM para clústeres de modo automático de EKS”
- the section called “Identidad y acceso”

> **Note**
>
> Si desea crear un clúster sin el modo automático de EKS, consulte the section called “Creación de un clúster ”.
> En este tema se trata la configuración avanzada. Si desea comenzar a utilizar el modo automático de EKS, consulte the section called “Crear un clúster”.

Requisitos previos

- Una VPC y subredes existentes que cumplen con los Requisitos de Amazon EKS. Antes de implementar un clúster para su uso en producción, le recomendamos que conozca a fondo los requisitos de VPC y subred. Si no tiene una VPC y subredes, puede crearlas mediante una plantilla de AWS CloudFormation proporcionada por Amazon EKS.

- La herramienta de línea de comandos de `kubectl` está instalada en su dispositivo o AWS CloudShell. La versión puede ser la misma o hasta una versión secundaria anterior o posterior a la versión de Kubernetes de su clúster. Por ejemplo, si la versión del clúster es `1.29`, puede usar la versión `1.28`, `1.29` o `1.30` de `kubectl` con él. Para instalar o actualizar `kubectl`, consulte the section called “Configure kubectl y eksctl”.
- La versión `2.12.3` o posterior, o bien, la versión `1.27.160` o posterior de la AWS interfaz de la línea de comandos (AWS CLI) instalada y configurada en su dispositivo o AWS CloudShell. Para comprobar su versión actual, utilice `aws --version`. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS.
- Una entidad principal de IAM con permisos para crear y modificar los recursos de EKS e IAM.

Crear un clúster: consola de AWS

1. Abra la consola de Amazon EKS.
2. Elija Agregar clúster y, a continuación, elija Crear.
3. En Opciones de configuración, seleccione Configuración personalizada.
   - En este tema se trata la configuración personalizada. Para obtener información sobre la configuración rápida, consulte the section called “Consola de administración”.
4. Confirme que la opción Usar el modo automático de EKS esté habilitada.
   - En este tema se trata la creación de clústeres con el modo automático de EKS. Para obtener más información sobre la creación de clústeres sin el modo automático de EKS, consulte the section called “Creación de un clúster ”.
5. En la página Configurar clúster rellene los siguientes campos:
   - Nombre: un nombre único para el clúster. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones bajos. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster.
   - Rol de IAM del clúster: elija el rol de IAM del clúster de Amazon EKS que creó para permitir que el plano de control de Kubernetes administre los recursos de AWS en su nombre. Si no ha creado previamente un rol de IAM de clúster para el modo automático de EKS, seleccione el botón Crear rol recomendado para crear el rol con los permisos necesarios en la consola de IAM.

- Versión de Kubernetes: la versión de Kubernetes que debe utilizarse para el clúster. Le recomendamos que seleccione la versión más reciente, a menos que necesite una versión anterior.
- Política de actualización: la política de la versión de Kubernetes que desea establecer para el clúster. Si quiere que el clúster solo se ejecute en una versión de soporte estándar, puede elegir Estándar. Si quiere que el clúster entre en el soporte extendido al final del soporte estándar de una versión, puede elegir Extendido. Si selecciona una versión de Kubernetes que actualmente tenga soporte extendido, no podrá seleccionar el soporte estándar como opción.

6. En la sección Computación en modo automático de la página de configuración del clúster, introduzca los siguientes campos:
   - Grupos de nodos: determine si desea utilizar los grupos de nodos integrados. Para obtener más información, consulte the section called “Revisión de los grupos de nodos integrados”.
   - Rol de IAM de nodo: si habilita alguno de los grupos de nodos integrados, debe seleccionar un rol de IAM de nodo. El modo automático de EKS asignará este rol a los nuevos nodos. No podrá cambiar este valor después de crear el clúster. Si no ha creado previamente un rol IAM de nodo para el modo automático de EKS, seleccione el botón Crear rol recomendado para crear el rol con los permisos necesarios. Para obtener más información acerca de este rol, consulte the section called “Identidad y acceso”.
7. En la sección Acceso al clúster de la página de configuración del clúster, introduzca los siguientes campos:
   - Acceso de administrador al clúster de arranque: el creador del clúster se convierte automáticamente en un administrador de Kubernetes. Si desea desactivar esta opción, seleccione Denegar acceso de administrador al clúster.
   - Modo de autenticación del clúster: el modo automático de EKS requiere entradas de acceso de EKS, el modo de autenticación de la API de EKS. Si lo desea, puede habilitar el modo de autenticación `ConfigMap`. Para ello, seleccione API de EKS y ConfigMap.
8. Ingrese los campos restantes en la página de configuración del clúster:
   - Cifrado de secretos: (opcional) elija habilitar el cifrado de secretos de los secretos de Kubernetes con una clave de KMS. También puede habilitarlo después de crear el clúster. Antes de habilitar esta capacidad, asegúrese de estar familiarizado con la información de the section called “Habilitación del cifrado de secretos”.
   - Cambio de zona de ARC: el modo automático de EKS no admite el cambio de zona de ARC.
   - Etiquetas: (opcional) agregue las etiquetas a su clúster. Para obtener más información, consulte the section called “Etiquetado de recursos”.

Cuando haya terminado con esta página, seleccione Siguiente.

9. En la página Especificar red seleccione valores para los siguientes campos:

- VPC: Elija una VPC existente que cumpla con los Requisitos de Amazon EKS VPC en el que crear el clúster. Antes de elegir una VPC, le recomendamos que se haya familiarizado con todos los requisitos y consideraciones de the section called “Requisitos de VPC y subred”. No puede cambiar la VPC que desea utilizar después de crear un clúster. Si no hay ninguna VPC en la lista, debe crear una primero. Para obtener más información, consulte the section called “Creación de una VPC”.
- Subredes: de forma predeterminada, se preseleccionan todas las subredes disponibles de la VPC especificada en el campo anterior. Debe seleccionar al menos dos.

  Las subredes que elija deben cumplir los Requisitos de subred de Amazon EKS. Antes de seleccionar subredes, recomendamos que esté familiarizado con todas las Consideraciones y requisitos de subred y VPC de Amazon EKS.

  Grupos de seguridad: (opcional) especifique uno o varios grupos de seguridad que desea que Amazon EKS asocie a las interfaces de red que crea.

  Independientemente de que elija grupos de seguridad o no, Amazon EKS crea un grupo de seguridad que permite la comunicación entre el clúster y la VPC. Amazon EKS asocia este grupo de seguridad, y el que elija, a las interfaces de red que crea. Para obtener más información acerca del grupo de seguridad de clúster que crea Amazon EKS, consulte the section called “Requisitos del grupo de seguridad”. Puede modificar las reglas del grupo de seguridad en el clúster que crea Amazon EKS.
- Elija la familia de direcciones IP del clúster: Puede elegir IPv4 e IPv6.

  Kubernetes asigna direcciones `IPv4` a sus pods y servicios de manera predeterminada. Antes de decidir utilizar la familia `IPv6`, asegúrese de estar familiarizado con todas las consideraciones y requisitos en los temas Requisitos y consideraciones de la VPC, the section called “Requisitos y consideraciones de la subred”, the section called “Requisitos del grupo de seguridad” y the section called “IPv6”. Si elige la familia `IPv6`, no puede especificar un rango de direcciones para que Kubernetes asigne direcciones de servicio `IPv6` desde el mismo origen que para la familia `IPv4`. Kubernetes asigna direcciones de servicio del rango de direcciones local exclusivo (`fc00::/7`).
- (Opcional) Elija Configuración del rango de direcciones IP de Kubernetes Service y especifique un Rango de servicio **`IPv4`**.

Especificar su propio intervalo puede ayudar a evitar conflictos entre servicios de Kubernetes y otras redes interconectadas o conectadas a la VPC. Escriba un rango en notación de CIDR. Por ejemplo: `10.2.0.0/16`.

El bloque de CIDR debe cumplir los siguientes requisitos:

- Se encuentra dentro de una de las siguientes gamas: `10.0.0.0/8`, `172.16.0.0/12` o `192.168.0.0/16`.
- Tiene un tamaño mínimo de `/24` y un tamaño máximo de `/12`.
- No se superponen con el rango de la VPC de los recursos de Amazon EKS.

Solo se puede especificar esta opción cuando se utiliza la familia `IPv4` de direcciones y solo en la creación de clústeres. Si no especifica esto, Kubernetes asignará direcciones IP de servicio desde los bloques de CIDR `10.100.0.0/16` o `172.20.0.0/16`.

- Para el Acceso al punto de conexión del clúster, seleccione una opción. Una vez que se crea el clúster, puede cambiar esta opción. Antes de seleccionar una opción no predeterminada, asegúrese de familiarizarse con las opciones y sus implicaciones. Para obtener más información, consulte the section called “Acceso al punto de conexión del clúster”.

  Cuando haya terminado con esta página, seleccione Siguiente.

10. (Opcional) En la página Configurar la observabilidad, seleccione qué opciones de métricas y registro de planos de control quiere activar. De forma predeterminada, cada tipo de registro está desactivado.
    - Para obtener más información sobre la opción de las métricas de Prometheus, consulte the section called “Paso 1: activación de métricas de Prometheus”.
    - Para obtener más información sobre las opciones de registro de plano de control, consulte the section called “Registros del plano de control”.
    - Cuando haya terminado con esta página, seleccione Siguiente.
11. En la página Seleccionar complementos, elija los complementos que desea agregar al clúster. Puede elegir tantos complementos de Amazon EKS y complementos de Marketplace de AWS como necesite. Si los complementos de AWS Marketplace que desea instalar no aparecen en la lista, puede hacer clic en la numeración de páginas para ver resultados de páginas adicionales o buscar complementos disponibles en AWS Marketplace al ingresar texto en el cuadro de búsqueda. También puede filtrar por categoría, proveedor o modelo de precios y, a continuación, elegir los complementos en los resultados de la búsqueda. Al crear un clúster, puede ver,

seleccionar e instalar cualquier complemento compatible con EKS Pod Identities, como se detalla en the section called “Pod Identity”.

- El modo automático de EKS automatiza la funcionalidad de algunos complementos. Si tiene previsto implementar grupos de nodos administrados de EKS en el clúster del modo automático de EKS, seleccione Complementos adicionales de Amazon EKS y revise las opciones. Es posible que necesite instalar complementos, como CoreDNS y kube-proxy. EKS solo instalará los complementos de esta sección en nodos y grupos de nodos autoadministrados.
- Cuando haya terminado con esta página, seleccione Siguiente.

12 En la página Configurar las opciones de complementos seleccionados, seleccione la versión que desee instalar. Siempre puede actualizar a una versión posterior después de crear el clúster.

En cuanto a los complementos compatibles con EKS Pod Identities, puede utilizar la consola para generar automáticamente el rol con el nombre, la política administrada por AWS y la política de confianza previamente insertados para cada complemento. Puede reutilizar roles existentes o crear nuevos roles para los complementos compatibles. Si desea conocer los pasos que hay que seguir en la consola para crear roles para complementos compatibles con EKS Pod Identities, consulte the section called “Creación de complemento (consola de AWS)”. Si un complemento no es compatible con EKS Pod Identity, aparecerá un mensaje con instrucciones sobre cómo utilizar el asistente para crear los roles de IAM para cuentas de servicio (IRSA) después de crear el clúster.

Puede actualizar la configuración de cada complemento después de crear el clúster. Para obtener más información acerca de la configuración de complementos, consulte the section called “Cómo actualizar un complemento”. Cuando haya terminado con esta página, seleccione Siguiente.

13 En la página Revisar y crear, revise la información que introdujo o seleccionó en las páginas anteriores. Si necesita realizar cambios, seleccione Edit (Editar). Cuando esté satisfecho, seleccione Crear. El campo Estado muestra CREANDO mientras se aprovisiona el clúster.

> **Note**
>
> Es posible que reciba un error que indique que una de las zonas de disponibilidad de la solicitud no tiene la capacidad suficiente para crear un clúster de Amazon EKS. Si esto ocurre, el mensaje de error indicará las zonas de disponibilidad que admiten un clúster nuevo. Intente crear el clúster de nuevo con al menos dos subredes ubicadas en las zonas de disponibilidad admitidas para su cuenta. Para obtener más información, consulte the section called “Capacidad insuficiente”.

El aprovisionamiento de clústeres tarda varios minutos.

Crear clúster: AWS CLI

Las siguientes instrucciones de la CLI cubren la creación de recursos de IAM y la creación del clúster.

Cómo crear un rol de IAM de clúster de modo automático de EKS

Paso 1: Creación de la política de confianza

Cree una política de confianza que permita al servicio de Amazon EKS asumir el rol. Guarde la política como `trust-policy.json`:

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

Paso 2: Creación del rol de IAM

Utilice la política de confianza para crear el rol de IAM del clúster:

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

Paso 3: Cómo anotar el ARN del rol

Recupere y guarde el ARN del nuevo rol para utilizarlo en pasos posteriores:

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

Paso 4: Asociación de las políticas requeridas

Asocie las siguientes políticas administradas por AWS al rol de IAM del clúster para conceder los permisos necesarios:

AmazonEKSClusterPolicy:

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

AmazonEKSComputePolicy:

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

AmazonEKSBlockStoragePolicy:

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

AmazonEKSLoadBalancingPolicy:

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

AmazonEKSNetworkingPolicy:

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

Cómo crear un rol de IAM de nodo del modo automático de EKS

Paso 1: Creación de la política de confianza

Cree una política de confianza que permita al servicio de Amazon EKS asumir el rol. Guarde la política como `node-trust-policy.json`:

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

Paso 2: Creación del rol de IAM del nodo

Utilice el archivo node-trust-policy.json del paso anterior para definir qué entidades pueden asumir el rol. Ejecute el siguiente comando para crear el rol de IAM del nodo:

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

Paso 3: Cómo anotar el ARN del rol

Después de crear el rol, recupere y guarde el ARN del rol de IAM del nodo. Necesitará este ARN en los pasos siguientes. Utilice el siguiente comando para obtener el ARN:

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

Paso 4: Asociación de las políticas requeridas

Asocie las siguientes políticas administradas por AWS al rol de IAM del nodo para proporcionar los permisos necesarios:

AmazonEKSWorkerNodeMinimalPolicy:

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

AmazonEC2ContainerRegistryPullOnly:

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

Creación de un clúster

1. Creación del clúster con el siguiente comando. Antes de ejecutar el comando, realice los siguientes reemplazos:
   - Reemplace *region-code* por la región de AWS en la que desea crear su clúster.
   - Reemplace *my-cluster* por el nombre de su clúster. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones bajos. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster.
   - Reemplace *1.30* por cualquier versión compatible con Amazon EKS.
   - Reemplace *111122223333* por el ID de su cuenta.
   - Si ha creado roles de IAM con nombres diferentes para los roles de clúster y nodo, sustituya los ARN.
   - Reemplace los valores `subnetIds` con valores propios. También puede agregar identificadores adicionales. Debe especificar al menos dos ID de subredes.

     Las subredes que elija deben cumplir los Requisitos de subred de Amazon EKS. Antes de seleccionar subredes, recomendamos que esté familiarizado con todas las Consideraciones y requisitos de subred y VPC de Amazon EKS.
   - Si no desea especificar un ID de grupo de seguridad, elimine `,securityGroupIds=sg-<ExampleID1>` del comando. Si desea especificar uno o varios ID de grupo de seguridad, sustituya los valores de `securityGroupIds` con la suya propia. También puede agregar identificadores adicionales.

     Independientemente de que elija grupos de seguridad o no, Amazon EKS crea un grupo de seguridad que permite la comunicación entre el clúster y la VPC. Amazon EKS asocia

este grupo de seguridad, y el que elija, a las interfaces de red que crea. Para obtener más información acerca del grupo de seguridad de clúster que crea Amazon EKS, consulte the section called “Requisitos del grupo de seguridad”. Puede modificar las reglas del grupo de seguridad en el clúster que crea Amazon EKS.

```
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**
>
> Es posible que reciba un error que indique que una de las zonas de disponibilidad de la solicitud no tiene la capacidad suficiente para crear un clúster de Amazon EKS. Si esto ocurre, el mensaje de error indicará las zonas de disponibilidad que admiten un clúster nuevo. Intente crear el clúster de nuevo con al menos dos subredes ubicadas en las zonas de disponibilidad admitidas para su cuenta. Para obtener más información, consulte the section called “Capacidad insuficiente”.

A continuación se muestra una configuración opcional que, si es necesario, debe agregarse al comando anterior. Solo puede habilitar estas opciones al crear el clúster, no después.

- Si quiere especificar desde qué bloque de enrutamiento entre dominios sin clase (CIDR) Kubernetes `IPv4` asigna direcciones IP de servicio, debe especificarlo agregando el `--kubernetes-network-config serviceIpv4Cidr=<cidr-block>` al siguiente comando.

  Especificar su propio intervalo puede ayudar a evitar conflictos entre servicios de Kubernetes y otras redes interconectadas o conectadas a la VPC. Escriba un rango en notación de CIDR. Por ejemplo: `10.2.0.0/16`.

El bloque de CIDR debe cumplir los siguientes requisitos:

- Se encuentra dentro de una de las siguientes gamas: `10.0.0.0/8`, `172.16.0.0/12` o `192.168.0.0/16`.
- Tiene un tamaño mínimo de `/24` y un tamaño máximo de `/12`.
- No se superponen con el rango de la VPC de los recursos de Amazon EKS.

  Solo se puede especificar esta opción cuando se utiliza la familia `IPv4` de direcciones y solo en la creación de clústeres. Si no especifica esto, Kubernetes asignará direcciones IP de servicio desde los bloques de CIDR `10.100.0.0/16` o `172.20.0.0/16`.

- Si está creando un clúster y desea que el clúster asigne direcciones `IPv6` a pods y servicios en lugar de direcciones `IPv4`, agregue `--kubernetes-network-config ipFamily=ipv6` al siguiente comando.

  Kubernetes asigna direcciones `IPv4` a sus pods y servicios de manera predeterminada. Antes de decidir utilizar la familia `IPv6`, asegúrese de estar familiarizado con todas las consideraciones y requisitos en los temas Requisitos y consideraciones de la VPC, the section called “Requisitos y consideraciones de la subred”, the section called “Requisitos del grupo de seguridad” y the section called “IPv6”. Si elige la familia `IPv6`, no puede especificar un rango de direcciones para que Kubernetes asigne direcciones de servicio `IPv6` desde el mismo origen que para la familia `IPv4`. Kubernetes asigna direcciones de servicio del rango de direcciones local exclusivo (`fc00::/7`).

2. La provisión del clúster puede tardar varios minutos. Puede consultar el estado del clúster con el siguiente comando.

```
aws eks describe-cluster --region region-code --name my-cluster --query
 "cluster.status"
```

Siguientes pasos

- the section called “Acceso al clúster con kubectl”
- the section called “Entradas de los registros”
- Habilitación del cifrado de secretos para su clúster.
- Configure el registro para el clúster.
- Agregue nodos al clúster.

Creación de un clúster de Amazon EKS

> **Note**
>
> En este tema, se trata la creación de clústeres de Amazon EKS sin el modo automático de EKS.
> Para obtener instrucciones detalladas sobre la creación de un clúster del modo automático de EKS, consulte the section called “Cómo crear un clúster automático”.
> Para comenzar a utilizar el modo automático de EKS, consulte the section called “Creación de un clúster (modo automático de EKS)”.

En este tema, se ofrece información general de las opciones disponibles y se describe qué debe tener en cuenta al crear un clúster de Amazon EKS. Si necesita crear un clúster con la infraestructura en las instalaciones como computación para los nodos, consulte the section called “Creación de un clúster”. Si es la primera vez que crea un clúster de Amazon EKS, recomendamos que siga una de nuestras guías en Introducción. Estas guías le ayudan a crear un clúster simple y predeterminado sin expandirse a todas las opciones disponibles.

Requisitos previos

- Una VPC y subredes existentes que cumplen con los Requisitos de Amazon EKS. Antes de implementar un clúster para su uso en producción, le recomendamos que conozca a fondo los requisitos de VPC y subred. Si no tiene una VPC y subredes, puede crearlas mediante una plantilla de AWS CloudFormation proporcionada por Amazon EKS.
- La herramienta de línea de comandos de `kubectl` está instalada en su dispositivo o AWS CloudShell. La versión puede ser la misma o hasta una versión secundaria anterior o posterior a la versión de Kubernetes de su clúster. Para instalar o actualizar `kubectl`, consulte the section called “Configure kubectl y eksctl”.
- La versión `2.12.3` o posterior, o bien, la versión `1.27.160` o posterior de la AWS interfaz de la línea de comandos (AWS CLI) instalada y configurada en su dispositivo o AWS CloudShell. Para comprobar su versión actual, utilice `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente.

Para actualizarla, consulte Instalación de la CLI de AWS en su directorio principal en la Guía del usuario de AWS CloudShell.

- Una entidad principal de IAM con permisos para `create` y `describe` un clúster de Amazon EKS. Para obtener más información, consulte the section called “Creación de un clúster local de Kubernetes en un Outpost” y the section called “Enumeración o descripción de todos los clústeres”.

Paso 1: creación de un rol de IAM del clúster

1. Si ya tiene un rol de IAM del clúster o va a crear su clúster con `eksctl`, puede omitir este paso. Por defecto, `eksctl` crea un rol.
2. Ejecute el siguiente comando para crear un archivo de política de confianza JSON de IAM.

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

3. Creación del rol de IAM del clúster de Amazon EKS. Si es necesario, introduzca *eks-cluster-role-trust-policy.json* con la ruta del equipo en la que escribió el archivo en el paso anterior. El comando asocia la política de confianza creada en el paso anterior al rol. Para crear un rol de IAM, a la entidad principal de IAM que está creando el rol se le debe asignar la acción `iam:CreateRole` (permiso).

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

4. Puede asignar la política administrada de Amazon EKS o bien crear su propia política personalizada. Para conocer los permisos mínimos que debe utilizar en su política personalizada, consulte the section called “Rol de IAM de clúster”.

Adjunte la política administrada de Amazon EKS llamada AmazonEKSClusterPolicy al rol. Para adjuntar una política de IAM a una entidad principal de IAM, se debe asignar una de las siguientes acciones de IAM (permisos) a la entidad principal que adjunta la política: `iam:AttachUserPolicy` o `iam:AttachRolePolicy`.

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

Función vinculada a un servicio

Amazon EKS crea automáticamente un rol vinculado al servicio llamado `AWSServiceRoleForAmazonEKS`.

Esto es además del rol de IAM del clúster. Un rol vinculado a servicios es un tipo único de rol de IAM que se encuentra vinculado directamente a Amazon EKS. El rol permite a Amazon EKS administrar clústeres de la cuenta. Para obtener más información, consulte the section called “Rol de del clúster”.

La identidad de IAM que utilice para crear el clúster de EKS debe tener permiso para crear el rol vinculado al servicio. Esto incluye el permiso `iam:CreateServiceLinkedRole`.

Si el rol vinculado al servicio aún no existe y el rol de IAM actual no tiene los permisos suficientes para crearlo, se producirá un error en la operación de creación del clúster.

Paso 2: creación de un clúster

Puede crear un clúster mediante:

- `eksctl`
- la Consola de administración de AWS
- La CLI de AWS

Crear un clúster: eksctl

1. Necesita la versión `0.215.0` o posterior de la herramienta de la línea de comandos de `eksctl` instalada en su dispositivo o AWS CloudShell. Para instalar o actualizar `eksctl`, consulte la sección de Instalación en la documentación de `eksctl`.

2. Cree un clúster `IPv4` de Amazon EKS con la versión más reciente de Kubernetes de Amazon EKS en su región de AWS predeterminada. Antes de ejecutar el comando, realice los siguientes reemplazos:
3. Reemplace *`region-code`* por la región de AWS en la que desea crear su clúster.
4. Reemplace *`my-cluster`* por el nombre de su clúster. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster.
5. Reemplace *`1.33`* por cualquier versión compatible con Amazon EKS.
6. Cambie los valores de `vpc-private-subnets` para satisfacer sus necesidades. También puede agregar identificadores adicionales. Debe especificar al menos dos ID de subredes. Si prefiere especificar subredes públicas, puede cambiar `--vpc-private-subnets` a `--vpc-public-subnets`. Las subredes públicas tienen una tabla de enrutamiento asociada a una ruta a una puerta de enlace de Internet, pero las subredes privadas no tienen una tabla de enrutamiento asociada. Recomendamos utilizar subredes privadas siempre que sea posible.

   Las subredes que elija deben cumplir los Requisitos de subred de Amazon EKS. Antes de seleccionar subredes, recomendamos que esté familiarizado con todas las Consideraciones y requisitos de subred y VPC de Amazon EKS.
7. Use el siguiente comando:

   ```
   eksctl create cluster --name my-cluster --region region-code --version 1.33 --vpc-private-subnets subnet-ExampleID1,subnet-ExampleID2 --without-nodegroup
   ```

   El aprovisionamiento de clústeres tarda varios minutos. Mientras se crea el clúster, aparecen varias líneas de salida. La última línea de salida es similar a la siguiente línea de ejemplo.

   ```
   [#]  EKS cluster "my-cluster" in "region-code" region is ready
   ```
8. Continúe con the section called “Paso 3: actualización de kubeconfig”

Optional Settings

Para ver la mayoría de las opciones que se pueden especificar al crear un clúster con `eksctl`, utilice el comando `eksctl create cluster --help`. Para ver todas las opciones disponibles, puede utilizar un archivo `config`. Para obtener más información, consulte Uso de archivos de configuración

y el esquema de archivos de configuración en la documentación de `eksctl`. Puede encontrar ejemplos de archivos de configuración en GitHub.

A continuación se muestra una configuración opcional que, si es necesario, debe agregarse al comando anterior. Solo puede habilitar estas opciones al crear el clúster, no después. Si necesita especificar estas opciones, debe crear el clúster con un archivo de configuración eksctl y especifique la configuración, en lugar de utilizar el comando anterior.

- Si desea especificar uno o varios grupos de seguridad que Amazon EKS asigna a las interfaces de red que crea, especifique la opción securityGroup.

  Independientemente de que elija grupos de seguridad o no, Amazon EKS crea un grupo de seguridad que permite la comunicación entre el clúster y la VPC. Amazon EKS asocia este grupo de seguridad, y el que elija, a las interfaces de red que crea. Para obtener más información acerca del grupo de seguridad de clúster que crea Amazon EKS, consulte the section called “Requisitos del grupo de seguridad”. Puede modificar las reglas del grupo de seguridad en el clúster que crea Amazon EKS.
- Si quiere especificar desde qué bloque de enrutamiento entre dominios sin clases (CIDR) `IPv4` de Kubernetes asigna direcciones IP de servicio, especifique la opción serviceIPv4CIDR.

  Especificar su propio intervalo puede ayudar a evitar conflictos entre servicios de Kubernetes y otras redes interconectadas o conectadas a la VPC. Escriba un rango en notación de CIDR. Por ejemplo: `10.2.0.0/16`.

  El bloque de CIDR debe cumplir los siguientes requisitos:

  - Se encuentra dentro de una de las siguientes gamas: `10.0.0.0/8`, `172.16.0.0/12` o `192.168.0.0/16`.
  - Tiene un tamaño mínimo de `/24` y un tamaño máximo de `/12`.
  - No se superponen con el rango de la VPC de los recursos de Amazon EKS.

    Solo se puede especificar esta opción cuando se utiliza la familia `IPv4` de direcciones y solo en la creación de clústeres. Si no especifica esto, Kubernetes asignará direcciones IP de servicio desde los bloques de CIDR `10.100.0.0/16` o `172.20.0.0/16`.
- Si va a crear un clúster y desea que el clúster asigne direcciones `IPv6` a pods y servicios en lugar de direcciones `IPv4`, especifique la opción ipFamily.

  Kubernetes asigna direcciones `IPv4` a sus pods y servicios de manera predeterminada. Antes de decidir utilizar la familia `IPv6`, asegúrese de haberse familiarizado con todas las consideraciones

y requisitos en los temas the section called “Requisitos y consideraciones de la VPC”, the section called “Requisitos y consideraciones de la subred”, the section called “Requisitos del grupo de seguridad” y the section called “IPv6”. Si elige la familia `IPv6`, no puede especificar un rango de direcciones para que Kubernetes asigne direcciones de servicio `IPv6` desde el mismo origen que para la familia `IPv4`. Kubernetes asigna direcciones de servicio del rango de direcciones local exclusivo (`fc00::/7`).

Crear un clúster: consola de AWS

1. Abra la consola de Amazon EKS.
2. Elija Agregar clúster y, a continuación, elija Crear.
3. En Opciones de configuración, seleccione Configuración personalizada.
   - Para obtener información sobre cómo crear rápidamente un clúster con el modo automático de EKS, consulte the section called “Consola de administración”.
4. En Modo automático de EKS, desactive Usar el modo automático de EKS.
   - Para obtener más información acerca de cómo crear un clúster del modo automático de EKS con una configuración personalizada, consulte the section called “Cómo crear un clúster automático”.
5. En la página Configurar clúster rellene los siguientes campos:
   - Nombre: un nombre único para el clúster. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones bajos. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster.
   - Rol de IAM del clúster: elija el rol de IAM del clúster de Amazon EKS que creó para permitir que el plano de control de Kubernetes administre los recursos de AWS en su nombre.
   - Versión de Kubernetes: la versión de Kubernetes que debe utilizarse para el clúster. Le recomendamos que seleccione la versión más reciente, a menos que necesite una versión anterior.
   - Tipo de soporte: la política de la versión de Kubernetes que se establecerá para su clúster. Si quiere que su clúster solo se ejecute en una versión de soporte estándar, puede elegir Soporte estándar. Si quiere que su clúster entre en el soporte extendido al final del soporte estándar de una versión, puede elegir Soporte extendido. Si selecciona una versión de Kubernetes que actualmente tenga soporte extendido, no podrá seleccionar el soporte estándar como opción.

- Cifrado de secretos: (opcional) elija habilitar el cifrado de secretos de los secretos de Kubernetes con una clave de KMS. También puede habilitarlo después de crear el clúster. Antes de habilitar esta capacidad, asegúrese de estar familiarizado con la información de the section called “Habilitación del cifrado de secretos”.
- Etiquetas: (opcional) agregue las etiquetas a su clúster. Para obtener más información, consulte the section called “Etiquetado de recursos”.
- Cambio de zona de ARC: (opcional) puede utilizar el controlador de recuperación de aplicaciones de Route53 para mitigar las zonas de disponibilidad afectadas. Para obtener más información, consulte the section called “Información sobre el cambio de zona”.

6. En la sección Acceso al clúster de la página de configuración del clúster, introduzca los siguientes campos:
   - Acceso de administrador al clúster de arranque: el creador del clúster se convierte automáticamente en un administrador de Kubernetes. Si desea desactivar esta opción, seleccione Denegar acceso de administrador al clúster.
   - Modo de autenticación de clústeres: determine cómo quiere conceder a los usuarios y roles de IAM el acceso a las API de Kubernetes. Para obtener más información, consulte the section called “Establecimiento de nodos de autenticación del clúster”.

   Cuando haya terminado con esta página, seleccione Siguiente.
7. En la página Especificar red seleccione valores para los siguientes campos:
   - VPC: Elija una VPC existente que cumpla con los Requisitos de Amazon EKS VPC en el que crear el clúster. Antes de elegir una VPC, le recomendamos que se haya familiarizado con todos los requisitos y consideraciones de the section called “Requisitos de VPC y subred”. No puede cambiar la VPC que desea utilizar después de crear un clúster. Si no hay ninguna VPC en la lista, debe crear una primero. Para obtener más información, consulte the section called “Creación de una VPC”.
   - Subredes: de forma predeterminada, se preseleccionan todas las subredes disponibles de la VPC especificada en el campo anterior. Debe seleccionar al menos dos.

     Las subredes que elija deben cumplir los Requisitos de subred de Amazon EKS. Antes de seleccionar subredes, recomendamos que esté familiarizado con todas las Consideraciones y requisitos de subred y VPC de Amazon EKS.

     Grupos de seguridad: (opcional) especifique uno o varios grupos de seguridad que desea que Amazon EKS asocie a las interfaces de red que crea.

Independientemente de que elija grupos de seguridad o no, Amazon EKS crea un grupo de seguridad que permite la comunicación entre el clúster y la VPC. Amazon EKS asocia este grupo de seguridad, y el que elija, a las interfaces de red que crea. Para obtener más información acerca del grupo de seguridad de clúster que crea Amazon EKS, consulte the section called “Requisitos del grupo de seguridad”. Puede modificar las reglas del grupo de seguridad en el clúster que crea Amazon EKS.

- Elija la familia de direcciones IP del clúster: Puede elegir IPv4 e IPv6.

  Kubernetes asigna direcciones `IPv4` a sus pods y servicios de manera predeterminada. Antes de decidir utilizar la familia `IPv6`, asegúrese de estar familiarizado con todas las consideraciones y requisitos en los temas Requisitos y consideraciones de la VPC, the section called “Requisitos y consideraciones de la subred”, the section called “Requisitos del grupo de seguridad” y the section called “IPv6”. Si elige la familia `IPv6`, no puede especificar un rango de direcciones para que Kubernetes asigne direcciones de servicio `IPv6` desde el mismo origen que para la familia `IPv4`. Kubernetes asigna direcciones de servicio del rango de direcciones local exclusivo (`fc00::/7`).

- (Opcional) Elija Configuración del rango de direcciones IP de Kubernetes Service y especifique un Rango de servicio **`IPv4`**.

  Especificar su propio intervalo puede ayudar a evitar conflictos entre servicios de Kubernetes y otras redes interconectadas o conectadas a la VPC. Escriba un rango en notación de CIDR. Por ejemplo: `10.2.0.0/16`.

  El bloque de CIDR debe cumplir los siguientes requisitos:

  - Se encuentra dentro de una de las siguientes gamas: `10.0.0.0/8`, `172.16.0.0/12` o `192.168.0.0/16`.
  - Tiene un tamaño mínimo de `/24` y un tamaño máximo de `/12`.
  - No se superponen con el rango de la VPC de los recursos de Amazon EKS.

Solo se puede especificar esta opción cuando se utiliza la familia `IPv4` de direcciones y solo en la creación de clústeres. Si no especifica esto, Kubernetes asignará direcciones IP de servicio desde los bloques de CIDR `10.100.0.0/16` o `172.20.0.0/16`.

- Para el Acceso al punto de conexión del clúster, seleccione una opción. Una vez que se crea el clúster, puede cambiar esta opción. Antes de seleccionar una opción no predeterminada, asegúrese de familiarizarse con las opciones y sus implicaciones. Para obtener más información, consulte the section called “Acceso al punto de conexión del clúster”.

Cuando haya terminado con esta página, seleccione Siguiente.

8. (Opcional) En la página Configurar la observabilidad, seleccione qué opciones de métricas y registro de planos de control quiere activar. De forma predeterminada, cada tipo de registro está desactivado.
   - Para obtener más información sobre la opción de las métricas de Prometheus, consulte the section called “Paso 1: activación de métricas de Prometheus”.
   - Para obtener más información sobre las opciones de registro de plano de control, consulte the section called “Registros del plano de control”.

   Cuando haya terminado con esta página, seleccione Siguiente.
9. En la página Seleccionar complementos, elija los complementos que desea agregar al clúster. Algunos complementos están preseleccionados. Puede elegir tantos complementos de Amazon EKS y complementos de Marketplace de AWS como necesite. Si los complementos de AWS Marketplace que desea instalar no aparecen en la lista, puede hacer clic en la numeración de páginas para ver resultados de páginas adicionales o buscar complementos disponibles en AWS Marketplace al ingresar texto en el cuadro de búsqueda. También puede filtrar por categoría, proveedor o modelo de precios y, a continuación, elegir los complementos en los resultados de la búsqueda. Al crear un clúster, puede ver, seleccionar e instalar cualquier complemento compatible con EKS Pod Identities, como se detalla en the section called “Pod Identity”.

   Cuando haya terminado con esta página, seleccione Siguiente.

   Algunos complementos, como la CNI de Amazon VPC, CoreDNS y kube-proxy, se instalan de forma predeterminada. Si desactiva alguno de los complementos predeterminados, esto puede afectar a su capacidad para ejecutar aplicaciones de Kubernetes.
10. En la página Configurar las opciones de complementos seleccionados, seleccione la versión que desee instalar. Siempre puede actualizar a una versión posterior después de crear el clúster.

    En cuanto a los complementos compatibles con EKS Pod Identities, puede utilizar la consola para generar automáticamente el rol con el nombre, la política administrada por AWS y la política de confianza previamente insertados para cada complemento. Puede reutilizar roles existentes o crear nuevos roles para los complementos compatibles. Si desea conocer los pasos que hay que seguir en la consola para crear roles para complementos compatibles con EKS Pod Identities, consulte the section called “Creación de complemento (consola de AWS)”. Si un complemento no es compatible con EKS Pod Identity, aparecerá un mensaje con instrucciones sobre cómo

utilizar el asistente para crear los roles de IAM para cuentas de servicio (IRSA) después de crear el clúster.

Puede actualizar la configuración de cada complemento después de crear el clúster. Para obtener más información acerca de la configuración de complementos, consulte the section called “Cómo actualizar un complemento”. Cuando haya terminado con esta página, seleccione Siguiente.

11. En la página Revisar y crear, revise la información que introdujo o seleccionó en las páginas anteriores. Si necesita realizar cambios, seleccione Edit (Editar). Cuando esté satisfecho, seleccione Crear. El campo Estado muestra CREANDO mientras se aprovisiona el clúster.

> **Note**
>
> Es posible que reciba un error que indique que una de las zonas de disponibilidad de la solicitud no tiene la capacidad suficiente para crear un clúster de Amazon EKS. Si esto ocurre, el mensaje de error indicará las zonas de disponibilidad que admiten un clúster nuevo. Intente crear el clúster de nuevo con al menos dos subredes ubicadas en las zonas de disponibilidad admitidas para su cuenta. Para obtener más información, consulte the section called “Capacidad insuficiente”.

El aprovisionamiento de clústeres tarda varios minutos.

12. Continúe con the section called “Paso 3: actualización de kubeconfig”

Crear clúster: AWS CLI

1. Creación del clúster con el siguiente comando. Antes de ejecutar el comando, realice los siguientes reemplazos:
   - Reemplace *region-code* por la región de AWS en la que desea crear su clúster.
   - Reemplace *my-cluster* por el nombre de su clúster. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones bajos. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster.
   - Reemplace *1.33* por cualquier versión compatible con Amazon EKS.
   - Reemplace *111122223333* por el ID de su cuenta y *myAmazonEKSClusterRole* por el nombre de su rol de IAM del clúster.

- Reemplace los valores `subnetIds` con valores propios. También puede agregar identificadores adicionales. Debe especificar al menos dos ID de subredes.

  Las subredes que elija deben cumplir los Requisitos de subred de Amazon EKS. Antes de seleccionar subredes, recomendamos que esté familiarizado con todas las Consideraciones y requisitos de subred y VPC de Amazon EKS.

- Si no desea especificar un ID de grupo de seguridad, elimine `,securityGroupIds=sg-<ExampleID1>` del comando. Si desea especificar uno o varios ID de grupo de seguridad, sustituya los valores de `securityGroupIds` con la suya propia. También puede agregar identificadores adicionales.

  Independientemente de que elija grupos de seguridad o no, Amazon EKS crea un grupo de seguridad que permite la comunicación entre el clúster y la VPC. Amazon EKS asocia este grupo de seguridad, y el que elija, a las interfaces de red que crea. Para obtener más información acerca del grupo de seguridad de clúster que crea Amazon EKS, consulte the section called “Requisitos del grupo de seguridad”. Puede modificar las reglas del grupo de seguridad en el clúster que crea Amazon EKS.

```
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**
>
> Es posible que reciba un error que indique que una de las zonas de disponibilidad de la solicitud no tiene la capacidad suficiente para crear un clúster de Amazon EKS. Si esto ocurre, el mensaje de error indicará las zonas de disponibilidad que admiten un clúster nuevo. Intente crear el clúster de nuevo con al menos dos subredes ubicadas en las zonas de disponibilidad admitidas para su cuenta. Para obtener más información, consulte the section called “Capacidad insuficiente”.

A continuación se muestra una configuración opcional que, si es necesario, debe agregarse al comando anterior. Solo puede habilitar estas opciones al crear el clúster, no después.

- De forma predeterminada, EKS instala varios complementos de red durante la creación del clúster. Esto incluye la CNI de Amazon VPC, CoreDNS y kube-proxy.

  Si desea desactivar la instalación de estos complementos de red predeterminados, utilice el siguiente parámetro. Esto se puede usar para CNI alternativos, como Cilium. Para obtener más información, consulte la Referencia de la API de EKS.

  ```
  aws eks create-cluster --bootstrapSelfManagedAddons false
  ```

- Si quiere especificar desde qué bloque de enrutamiento entre dominios sin clase (CIDR) Kubernetes `IPv4` asigna direcciones IP de servicio, debe especificarlo agregando el `--kubernetes-network-config serviceIpv4Cidr=<cidr-block>` al siguiente comando.

  Especificar su propio intervalo puede ayudar a evitar conflictos entre servicios de Kubernetes y otras redes interconectadas o conectadas a la VPC. Escriba un rango en notación de CIDR. Por ejemplo: `10.2.0.0/16`.

  El bloque de CIDR debe cumplir los siguientes requisitos:

  - Se encuentra dentro de una de las siguientes gamas: `10.0.0.0/8`, `172.16.0.0/12` o `192.168.0.0/16`.
  - Tiene un tamaño mínimo de `/24` y un tamaño máximo de `/12`.
  - No se superponen con el rango de la VPC de los recursos de Amazon EKS.

  Solo se puede especificar esta opción cuando se utiliza la familia `IPv4` de direcciones y solo en la creación de clústeres. Si no especifica esto, Kubernetes asignará direcciones IP de servicio desde los bloques de CIDR `10.100.0.0/16` o `172.20.0.0/16`.

- Si está creando un clúster y desea que el clúster asigne direcciones `IPv6` a pods y servicios en lugar de direcciones `IPv4`, agregue `--kubernetes-network-config ipFamily=ipv6` al siguiente comando.

  Kubernetes asigna direcciones `IPv4` a sus pods y servicios de manera predeterminada. Antes de decidir utilizar la familia `IPv6`, asegúrese de estar familiarizado con todas las consideraciones y requisitos en los temas Requisitos y consideraciones de la VPC, the section called “Requisitos y consideraciones de la subred”, the section called “Requisitos del grupo de seguridad” y the section called “IPv6”. Si elige la familia `IPv6`, no puede especificar un rango de direcciones para que Kubernetes asigne direcciones de servicio `IPv6` desde el mismo origen que para la familia `IPv4`. Kubernetes asigna direcciones de servicio del rango de direcciones local exclusivo (`fc00::/7`).

2. La provisión del clúster puede tardar varios minutos. Puede consultar el estado del clúster con el siguiente comando.

```
aws eks describe-cluster --region region-code --name my-cluster --query
 "cluster.status"
```

   No continúe con el siguiente paso hasta que la salida devuelta sea `ACTIVE`.

3. Continúe con the section called “Paso 3: actualización de kubeconfig”

Paso 3: actualización de kubeconfig

1. Si ha creado el clúster mediante `eksctl`, puede omitir este paso. Esto se debe a que `eksctl` ya ha completado este paso. Habilite `kubectl` para comunicarse con el clúster agregando un nuevo contexto al archivo `kubectl config`. Para obtener más información acerca de cómo crear y actualizar el archivo, consulte the section called “Acceso al clúster con kubectl”.

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

   Un ejemplo de salida sería el siguiente.

```
Added new context arn:aws:eks:region-code:111122223333:cluster/my-cluster to /home/
username/.kube/config
```

2. Confirme la comunicación con el clúster ejecutando el siguiente comando.

```
kubectl get svc
```

   Un ejemplo de salida sería el siguiente.

```
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   28h
```

Paso 4: configuración del clúster

1. (Recomendado) Para utilizar algunos complementos de Amazon EKS o a fin de permitir que las cargas de trabajo individuales de Kubernetes cuenten con permisos de AWS Identity and

Access Management (IAM) específicos, cree un proveedor de OpenID Connect de IAM (OIDC) para el clúster. Solo necesita crear un proveedor de OIDC de IAM para su clúster una vez. Para obtener más información sobre los complementos de Amazon EKS, consulte the section called "Complementos de Amazon EKS". Para obtener más información sobre la asignación de permisos de IAM específicos a sus cargas de trabajo, consulte the section called "Credenciales con IRSA".

2. (Recomendado) Configure el clúster para el complemento CNI de Amazon VPC para Kubernetes antes de implementar nodos de Amazon EC2 en su clúster. De forma predeterminada, el complemento se instaló con el clúster. Cuando agrega nodos de Amazon EC2 a su clúster, el complemento se implementa automáticamente en cada nodo de Amazon EC2 que agregue. El complemento requiere que adjunte una de las siguientes políticas de IAM a un rol de IAM. Si el clúster usa la familia `IPv4`, utilice la política de IAM administrada AmazonEKS_CNI_Policy. Si el clúster utiliza la familia `IPv6`, utilice una política de IAM que cree.

   El rol de IAM al que adjunta la política puede ser el rol de IAM de nodo o un rol dedicado que se usa solo para el complemento. Recomendamos adjuntar la política a este rol. Para obtener más información sobre la creación del rol, consulte the section called "Configuración para IRSA" o Rol de IAM de nodo de Amazon EKS.

3. Si ha implementado el clúster mediante el Consola de administración de AWS, puede omitir este paso. La Consola de administración de AWS implementa los complementos CNI de Amazon VPC para Kubernetes, CoreDNS y `kube-proxy` de Amazon EKS de forma predeterminada.

   Si implementa el clúster mediante `eksctl` o la AWS CLI, se implementan los complementos CNI de Amazon VPC para Kubernetes, CoreDNS y `kube-proxy` autoadministrados. Puede migrar los complementos CNI de Amazon VPC para Kubernetes, CoreDNS y `kube-proxy` que se implementan con su clúster en complementos de Amazon EKS. Para obtener más información, consulte the section called "Complementos de Amazon EKS".

4. (Opcional) Si aún no lo ha hecho, puede habilitar las métricas de Prometheus para su clúster. Para obtener más información, consulte Crear un raspador en la Guía del usuario de Amazon Managed Service para Prometheus.

5. Si tiene previsto implementar cargas de trabajo que utilicen volúmenes de Amazon EBS en su clúster, deberá instalar el CSI de Amazon EBS en el clúster antes de implementar las cargas de trabajo.

Siguientes pasos

- La entidad principal de IAM que creó el clúster es la única entidad principal de IAM que tiene acceso al clúster. Conceda permisos a otras entidades principales de IAM para que puedan acceder al clúster.
- Si la entidad principal de IAM que creó el clúster solo tiene los permisos de IAM mínimos a los que se hace referencia en los requisitos previos, es posible que desee agregar permisos de Amazon EKS adicionales para esa entidad principal. Para obtener más información sobre cómo conceder permisos de Amazon EKS a entidades principales de IAM, consulte the section called “Referencia de IAM”.
- Si desea que la entidad principal de IAM que creó el clúster o cualquier otra entidad principal vea los recursos de Kubernetes en la consola de Amazon EKS, conceda los Permisos necesarios a las entidades.
- Si desea que los nodos y las entidades principales de IAM accedan al clúster desde su VPC, habilite el punto de conexión privado para el clúster. El punto de conexión público está habilitado de forma predeterminada. Si lo desea, puede desactivar el punto de conexión público una vez que haya activado el punto de conexión privado. Para obtener más información, consulte the section called “Acceso al punto de conexión del clúster”.
- Habilitación del cifrado de secretos para su clúster.
- Configure el registro para el clúster.
- Agregue nodos al clúster.

Plano de control aprovisionado de Amazon EKS

Descripción general

El plano de control aprovisionado de Amazon EKS es una característica que permite a los administradores de clústeres seleccionar entre un conjunto de niveles de escalado y designar el nivel que elijan para obtener un rendimiento muy alto y predecible desde el plano de control del clúster. Esto permite a los administradores de clústeres garantizar que el plano de control siempre esté aprovisionado con la capacidad especificada.

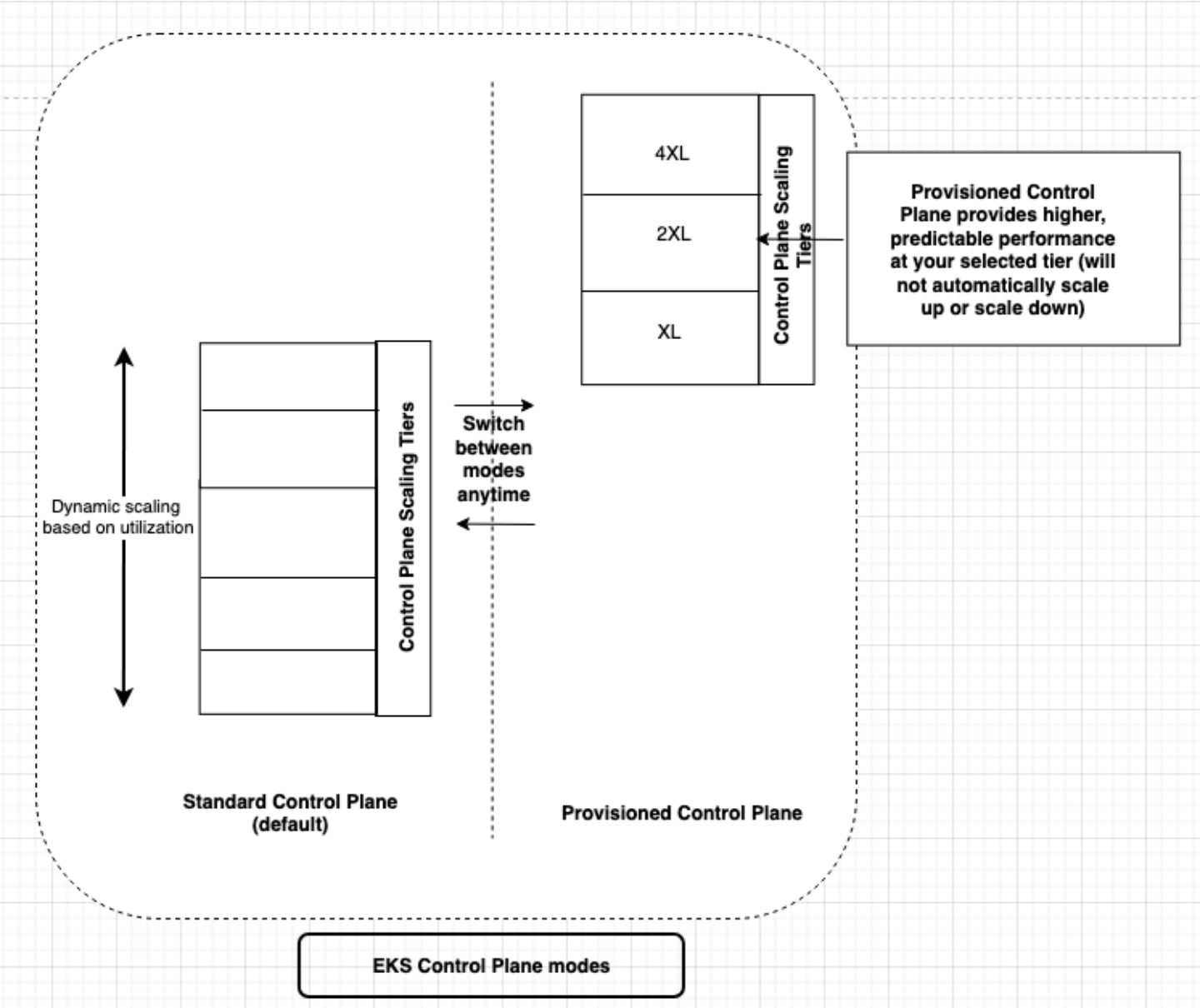
Amazon EKS ofrece dos modos de operación para el plano de control del clúster. De forma predeterminada, los clústeres de Amazon EKS utilizan el modo estándar, en el que el plano de control se escala y se reduce verticalmente en función de las exigencias de la carga de trabajo. El

modo estándar asigna dinámicamente la capacidad suficiente del plano de control para satisfacer sus necesidades de carga de trabajo y es la solución recomendada para la mayoría de los casos de uso. Sin embargo, para las cargas de trabajo especializadas que no toleran ninguna variabilidad en el rendimiento debido al escalado del plano de control o aquellas que requieren una capacidad muy alta del plano de control, puede utilizar opcionalmente el modo aprovisionado. El modo aprovisionado le permite asignar previamente la capacidad del plano de control para que siempre esté lista para gestionar los exigentes requisitos de la carga de trabajo.

> **Note**
>
> El modo aprovisionado es un modo de operaciones del plano de control adicional junto con el modo estándar predeterminado. La introducción del modo aprovisionado no cambia el comportamiento del modo estándar.

Con el plano de control aprovisionado de EKS, los administradores de clústeres pueden aprovisionar previamente la capacidad deseada del plano de control con antelación, lo que proporciona un rendimiento alto y predecible desde el plano de control del clúster, que siempre está disponible. El plano de control aprovisionado de EKS también permite a los administradores de clústeres aprovisionar la misma capacidad de plano de control en todos los entornos, desde los sitios de pruebas a los de producción y recuperación ante desastres. Es importante para garantizar que el rendimiento del plano de control obtenido en todos los entornos sea coherente y predecible. Por último, el plano de control aprovisionado de EKS le permite acceder a niveles muy altos de rendimiento del plano de control, lo que le permite ejecutar cargas de trabajo de IA escalables de forma masiva, computación de alto rendimiento y cargas de trabajo de procesamiento de datos a gran escala en Kubernetes.

Todos los clústeres de Amazon EKS nuevos y existentes funcionan en modo estándar de forma predeterminada. Para los clústeres que requieren un rendimiento alto y predecible desde el plano de control, puede optar por utilizar la característica del plano de control aprovisionado de EKS. Se le facturará la tarifa por hora correspondiente al nivel de escalado del plano de control concreto, además de las tarifas por hora de EKS de soporte estándar o ampliado. Para obtener más información acerca de los precios, consulte Precios de Amazon EKS.



Casos de uso

El plano de control aprovisionado de EKS está diseñado para abordar situaciones específicas en las que un rendimiento alto y predecible del plano de control es fundamental para sus operaciones. Comprender estos casos de uso puede ayudarlo a determinar si el plano de control aprovisionado de EKS es la solución adecuada para sus cargas de trabajo.

Cargas de trabajo críticas para el rendimiento: para las cargas de trabajo que exigen una latencia mínima y un rendimiento máximo del plano de control de Kubernetes, el plano de control aprovisionado de EKS proporciona una capacidad que elimina la variabilidad del rendimiento al escalar el plano de control.

Cargas de trabajo con escalabilidad masiva: si ejecuta cargas de trabajo con alta escalabilidad, como inferencia y entrenamiento de IA, computación de alto rendimiento o procesamiento de datos a gran escala que requieren la ejecución de una gran cantidad de nodos en el clúster, el plano de control aprovisionado proporciona la capacidad de plano de control necesaria para soportar estas cargas de trabajo exigentes.

Eventos anticipados de alta demanda: cuando espere un aumento repentino de las solicitudes del plano de control debido a un evento próximo, como rebajas o promociones de comercio electrónico, lanzamientos de productos, temporadas de compras festivas o eventos deportivos o de entretenimiento importantes, el plano de control aprovisionado le permite escalar la capacidad del plano de control por adelantado. Este enfoque proactivo garantiza que el plano de control esté preparado para soportar el aumento de carga sin tener que esperar a que se escale automáticamente para responder a la demanda.

Coherencia del entorno: el plano de control aprovisionado le permite igualar la capacidad y el rendimiento del plano de control en los entornos de preparación y producción, lo que lo ayuda a identificar posibles problemas con antelación antes de la implementación en producción. Al mantener el mismo nivel del plano de control en todos los entornos, puede asegurarse de que los resultados de las pruebas reflejen con precisión el comportamiento de la producción, lo que reduce el riesgo de sorpresas relacionadas con el rendimiento durante la implementación.

Recuperación ante desastres y continuidad empresarial: para escenarios de recuperación ante desastres, el plano de control aprovisionado le permite aprovisionar entornos de conmutación por error con el mismo nivel de capacidad que el entorno principal. Esto garantiza una interrupción mínima y una recuperación rápida durante los eventos de conmutación por error, ya que el clúster de recuperación ante desastres tendrá características de rendimiento en el plano de control idénticas a las del clúster de producción desde el momento en que se active.

Niveles de escalado del plano de control

El plano de control aprovisionado de EKS ofrece niveles de escalado con el mismo nombre que las tallas de camisetas (XL, 2XL, 4XL). Cada nivel define su capacidad mediante tres atributos clave de Kubernetes que determinan las características de rendimiento del plano de control del clúster. Comprender estos atributos lo ayuda a seleccionar el nivel adecuado para sus requisitos de cargas de trabajo.

La simultaneidad de solicitudes de API mide la cantidad de solicitudes que el servidor de API del plano de control de Kubernetes puede procesar simultáneamente, lo cual es fundamental para las cargas de trabajo de alto rendimiento.

La frecuencia de programación de los pods indica la rapidez con la que el programador predeterminado de Kubernetes puede programar los pods en los nodos, que se mide en pods por segundo.

El tamaño de la base de datos del clúster indica el espacio de almacenamiento asignado a etcd, la base de datos que contiene el estado y los metadatos del clúster.

Al aprovisionar el plano de control del clúster en un nivel de escalado determinado mediante el plano de control aprovisionado, EKS garantiza que el plano de control del clúster mantenga los límites correspondientes a ese nivel. Los límites de los niveles de escalado del plano de control varían en función de la versión de Kubernetes, como se muestra en las siguientes tablas.

EKS v1.28 y v1.29

Nivel de escalado del plano de control aprovisionado	Simultaneidad de solicitudes de la API (plazas)	Frecuencia de programación de los pods (pods/seg)	Tamaño de la base de datos del clúster (GB)
XL	1700	100	16
2XL	3400	100	16
4XL	6800	100	16

EKS v1.30 y versiones posteriores

Nivel de escalado del plano de control aprovisionado	Simultaneidad de solicitudes de la API (plazas)	Frecuencia de programación de los pods (pods/seg)	Tamaño de la base de datos del clúster (GB)
XL	1700	167	16
2XL	3400	283	16
4XL	6800	400	16

Supervisión del uso de los niveles de escalado del plano de control

Amazon EKS proporciona varias métricas para ayudarlo a supervisar el uso de los niveles del plano de control. Estas métricas se publican como métricas de Amazon CloudWatch y se puede acceder a ellas a través de las consolas de CloudWatch y EKS. Además, estas métricas se pueden extraer del punto de conexión de Prometheus del clúster de EKS (consulte esta página).

	Métrica de Prometheus	Métrica de CloudWatch
Simultaneidad de solicitudes de la API	apiserver_flowcontrol_current_executing_seats	apiserver_flowcontrol_current_executing_seats
Frecuencia de programación de pod	scheduler_schedule_attempts_total	scheduler_schedule_attempts_total, scheduler_schedule_attempts_SCHEDULED, scheduler_schedule_attempts_UNSCHEDULABLE
Tamaño de la base de datos del clúster	apiserver_storage_size_bytes	apiserver_storage_size_bytes

Descripción de la capacidad de los niveles frente al rendimiento real

Cuando selecciona un nivel de escalado del plano de control aprovisionado, los atributos del nivel representan las configuraciones subyacentes que aplica Amazon EKS al plano de control. Sin embargo, el rendimiento real que logre depende de los patrones de carga de trabajo específicos, las configuraciones y el cumplimiento de las prácticas recomendadas de Kubernetes. Por ejemplo, si bien un nivel 4XL configura la prioridad y equidad de la API (APF) con 6800 solicitudes simultáneas, el rendimiento real de las solicitudes que se obtienen desde el plano de control depende del tipo de operaciones que se lleven a cabo. Por ejemplo, Kubernetes penaliza más las solicitudes de lista que las de obtención y, por lo tanto, la cantidad efectiva de solicitudes de lista procesadas simultáneamente por el plano de control es inferior al de las solicitudes de obtención (consulte esta página). Del mismo modo, aunque el QPS del programador predeterminado se establece en 400 para un nivel 4XL, la tasa real de programación del pod depende de factores como la preparación de los nodos y su estado para la programación. Para lograr un rendimiento óptimo, asegúrese de que las aplicaciones sigan las prácticas recomendadas de Kubernetes (consulte esta página) y de que estén configuradas correctamente según las características de la carga de trabajo.

Consideraciones

- Capacidad del plano de control estándar: el modo del plano de control estándar de EKS ofrece la mejor relación entre precio y rendimiento y es la opción recomendada para la gran mayoría de los casos de uso. Sin embargo, para las cargas de trabajo especializadas que no toleran ninguna variabilidad en el rendimiento debido al escalado del plano de control o aquellas que requieren una capacidad muy alta del plano de control, puede plantearse opcionalmente el modo aprovisionado.
- Suscripción obligatoria: los clústeres existentes no se escalarán verticalmente de forma automática desde el plano de control estándar a un nivel del plano de control aprovisionado de EKS más caro. Debe suscribirse de forma explícita a uno de los nuevos niveles de escalado del plano de control aprovisionado de EKS.
- Restricción de salida: el modo de plano de control estándar admite hasta 8 GB de tamaño de base de datos del clúster (etcd). Si el tamaño de la base de datos del clúster supera los 8 GB mientras utiliza el modo aprovisionado, no podrá volver al modo estándar hasta que reduzca el tamaño de la base de datos a menos de 8 GB. Por ejemplo, si utiliza 14 GB de almacenamiento de base de datos en el modo aprovisionado, debe reducir previamente el uso de la base de datos a menos de 8 GB antes de volver al modo estándar.
- Sin escalado automático de niveles: el plano de control aprovisionado de EKS no escala automáticamente entre niveles. Una vez que selecciona un nivel de escalado, el plano de control del clúster permanece fijo en ese nivel, lo que garantiza un rendimiento coherente y predecible. Sin embargo, tiene la flexibilidad de implementar su propia solución de escalado automático mediante la supervisión de las métricas de uso de los niveles y el uso de las API del plano de control aprovisionado de EKS para reducir o escalar verticalmente cuando estas métricas superen los umbrales definidos, lo que le ofrece un control total sobre su estrategia de escalado y la optimización de costos.
- Visualización del nivel actual: puede utilizar la consola de Amazon EKS, la Amazon Web Services CLI o la API para ver el nivel de escalado del plano de control actual. En la CLI, puede ejecutar el comando `describe-cluster`: `aws eks describe-cluster --name cluster-name`
- Tiempo de transición entre niveles: puede utilizar la consola de Amazon EKS, las API de Amazon EKS o la CLI para salir de los niveles de escalado o moverse entre ellos. Amazon EKS ha introducido un nuevo tipo de actualización de clústeres denominado `ScalingTierConfigUpdate`, que puede inspeccionar para supervisar el progreso de la transición. Después de ejecutar un comando de cambio de nivel, puede enumerar las actualizaciones del clúster para ver una nueva actualización del tipo `ScalingTierConfigUpdate` con el estado `Updating`. El estado cambia a `Successful` al

finalizar la actualización o a `Failed` si se produce un error. El campo de error de la actualización indica el motivo del error. No hay restricciones en cuanto a la frecuencia con la que puede cambiar de nivel. El cambio del nivel del plano de control tarda varios minutos en completarse.

- Selección del nivel óptimo: para determinar el nivel de escalado del plano de control aprovisionado óptimo para el clúster, puede llevar a cabo pruebas de carga mediante el aprovisionamiento del clúster en el nivel más alto (4XL). A continuación, lleve a cabo una prueba de carga para simular los picos de demanda en el plano de control del clúster. Observe las métricas de uso de los niveles del plano de control en los momentos de máxima carga y utilice estas observaciones como factor guía para seleccionar el nivel adecuado para el modo aprovisionado.
- Precios del plano de control aprovisionado: se le facturará según la tarifa por hora correspondiente al nivel de escalado del plano de control aprovisionado en el que se encuentre el clúster. Esto se suma a los cargos por hora de soporte estándar o ampliado. Consulte la página de precios de EKS para obtener información detallada.
- Nivel de escalado más grande: si tiene previsto ejecutar el clúster en un nivel de escalado superior a 4XL, contacte con el equipo de cuentas de Amazon Web Services para obtener información adicional sobre los precios.
- Compatibilidad con la versión y la región de Kubernetes: el plano de control aprovisionado de EKS se admite en todas las regiones comerciales de Amazon Web Services, GovCloud y China. El plano de control aprovisionado funciona en EKS v1.28 y versiones posteriores.

Introducción al plano de control aprovisionado de Amazon EKS

En esta guía, se explican los pasos para configurar y usar el plano de control aprovisionado de EKS mediante la AWS CLI y la Consola de administración de AWS.

Requisitos previos

Antes de comenzar, asegúrese de que dispone de lo siguiente:

- AWS CLI: una herramienta de línea de comandos para trabajar con servicios de AWS, incluido Amazon EKS. Para obtener más información, consulte Instalación en la Guía del usuario de la Interfaz de la línea de comandos de AWS. Después de instalar la AWS CLI, recomendamos que también la configure. Para obtener más información, consulte Configuración rápida con aws configure en la Guía del usuario de la Interfaz de la línea de comandos de AWS. Tenga en cuenta que necesita la v2 de la AWS CLI para utilizar la opción update-kubeconfig que se muestra en esta página.

- Permisos de IAM necesarios: la entidad principal de seguridad de IAM que está utilizando debe contar con permisos para trabajar con los roles de IAM de Amazon EKS, los roles vinculados al servicio, AWS CloudFormation, una VPC y recursos relacionados. Para obtener más información, consulte Acciones y Uso de roles vinculados a servicios en la Guía del usuario de IAM. Debe completar todos los pasos de esta guía como el mismo usuario. Ejecute el siguiente comando para comprobar el usuario actual:

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

> **Note**
>
> Le recomendamos que siga los pasos de este tema en un intérprete de comandos Bash. Si no está utilizando un intérprete de comandos Bash, algunos comandos de script, como los caracteres de continuación de línea y la forma en que se establecen y utilizan las variables, requieren ajustes para su intérprete de comandos. Además, las reglas de entrecomillado y escape de su intérprete de comandos pueden ser diferentes. Para obtener más información, consulte Uso de entrecomillado de cadenas en la AWS CLI en la Guía del usuario de la Interfaz de la línea de comandos de AWS.

Plano de control aprovisionado de EKS: AWS CLI

Creación de un clúster con el nivel de escalado del plano de control aprovisionado de EKS

```
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
```

Respuesta:

```
{
    "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"
        }
    }
}
```

Visualización del nivel de escalado del plano de control del clúster

```
aws eks describe-cluster --name prod-cluster
```

Respuesta:

```
{
    "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"
        }
    }
}
```

Actualización del clúster para usar el plano de control aprovisionado de EKS

```
aws eks update-cluster-config --name prod-cluster \
--control-plane-scaling-config tier=tier-2xl
```

Respuesta:

```
{
    "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": []
    }
}
```

Visualización de la actualización de escalado del plano de control

```
aws eks list-updates --name example
```

Respuesta:

```
{
    "updateIds": [
        "7551c64b-1d27-4b1e-9f8e-c45f056eb6fd1"
    ]
}
```

Salida del plano de control aprovisionado al plano de control estándar

```
aws eks update-cluster-config --name prod-cluster \
```

```
--control-plane-scaling-config tier=standard
```

Respuesta:

```
{
    "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": []
    }
}
```

Plano de control aprovisionado de EKS: Consola de administración de AWS

1. Abra la consola de Amazon EKS.
2. Elija Create cluster.
3. En Opciones de configuración, seleccione Configuración personalizada.
4. Desplácese hacia abajo hasta Nivel de escalado del plano de control. Seleccione Usar un nivel de escalado para activar el plano de control aprovisionado.
5. Seleccione el nivel de escalado del plano de control que desee aprovisionar para el clúster entre varias opciones de niveles de escalado, como XL, 2XL y 4XL.
6. Seleccione otras opciones de configuración del clúster según sea necesario. En el último paso, seleccione Crear clúster. Tenga en cuenta que la creación del clúster puede tardar varios minutos en completarse.

Preparación para las actualizaciones de las versiones de Kubernetes y solución de problemas de configuración incorrecta con información sobre clústeres

La información sobre clústeres de Amazon EKS permite detectar problemas y ofrece recomendaciones para resolverlos y ayudarlo a administrar el clúster. Todos los clústeres de Amazon EKS se someten a comprobaciones automáticas y periódicas con una lista de información seleccionada por Amazon EKS. Amazon EKS administra en su totalidad estas comprobaciones de información y ofrece recomendaciones sobre cómo abordar cualquier resultado.

Tipos de información del clúster

- Información de configuración: identifica errores de configuración en la configuración de los nodos híbridos de EKS que podrían afectar a la funcionalidad del clúster o las cargas de trabajo.
- Información de actualización: identifica los problemas que podrían afectar a su capacidad de actualizar a nuevas versiones de Kubernetes.

Consideraciones

- Frecuencia: Amazon EKS actualiza la información de los clústeres cada 24 horas, pero también puede actualizarla manualmente para ver el estado más reciente. Por ejemplo, puede actualizar manualmente la información del clúster después de solucionar un problema para comprobar si el problema se ha resuelto.
- Permisos: Amazon EKS crea automáticamente una entrada de acceso al clúster para su información en cada clúster de EKS. Esta entrada concede permiso a EKS para ver información sobre su clúster. Amazon EKS usa esta información para generar la información. Para obtener más información, consulte the section called “AmazonEKSClusterInsightsPolicy”.

Casos de uso

La información sobre clústeres de Amazon EKS proporciona comprobaciones automatizadas para ayudar a mantener el buen estado, la fiabilidad y la configuración óptima de los clústeres de Kubernetes. A continuación, se presentan los principales casos de uso para obtener información sobre los clústeres, incluida la preparación para la actualización y la resolución de problemas de configuración.

Información de actualización

La información de actualización es un tipo específico de comprobación de información dentro de la información de los clústeres. Estas comprobaciones devuelven información relacionada con la preparación para la actualización de la versión de Kubernetes. Amazon EKS lleva a cabo comprobaciones de información de actualización en todos los clústeres de EKS.

> **Important**
>
> Amazon EKS ha revertido temporalmente una característica que obligaba a utilizar una marca `--force` para actualizar el clúster cuando se producían determinados problemas de conocimiento del clúster. Para obtener más información, consulte Temporary rollback of enforcing upgrade insights on update cluster version en GitHub.
> Para obtener más información sobre la actualización del clúster, consulte the section called "Paso 3: actualización del plano de control del clúster".

Antes de actualizar la versión de Kubernetes del clúster, puede usar la pestaña Información sobre la actualización en el panel de observabilidad de la consola de Amazon EKS. Si su clúster ha identificado problemas, revíselos y aplique las correcciones adecuadas. Los problemas incluyen enlaces a Amazon EKS y documentación de Kubernetes. Después de solucionar el problema, actualice la información del clúster bajo demanda para obtener la información más reciente. Si se han resuelto todos los problemas, actualice el clúster.

Amazon EKS devuelve información relacionada con la preparación para la actualización de la versión de Kubernetes. La información sobre las actualizaciones identifica los posibles problemas que podrían afectar a las actualizaciones del clúster de Kubernetes. Esto minimiza el esfuerzo que los administradores dedican a preparar las actualizaciones y aumenta la fiabilidad de las aplicaciones en las versiones más recientes de Kubernetes. Amazon EKS analiza automáticamente los clústeres para compararlos con una lista de posibles problemas que podrían afectar a las actualizaciones de la versión de Kubernetes. Amazon EKS actualiza con frecuencia la lista de comprobaciones de información en función de las revisiones de los cambios hechos en cada lanzamiento de versión de Kubernetes.

La información sobre las actualizaciones de Amazon EKS acelera el proceso de prueba y verificación de las nuevas versiones. También permiten a los administradores de clústeres y a los desarrolladores de aplicaciones aprovechar las capacidades más recientes de Kubernetes, ya que destacan las inquietudes y ofrecen consejos para solucionarlas.

Información de configuración

La información del clúster de EKS escanea automáticamente los clústeres de Amazon EKS con nodos híbridos para identificar problemas de configuración que afectan a la comunicación entre el plano de control y el webhook de Kubernetes, comandos de kubectl como exec y registros, etc. La información de configuración detecta problemas y ofrece recomendaciones para solucionarlos, lo que acelera el proceso de configuración de los nodos híbridos en pleno funcionamiento.

Introducción

Para ver la lista de comprobaciones de información hechas y cualquier problema pertinente que Amazon EKS haya identificado, puede usar la operación Consola de administración de AWS, la CLI de AWS, los SDK de AWS y la API `ListInsights` de Amazon EKS. Para empezar, consulte the section called “Consulta de la información del clúster”.

Consulta de la información del clúster

Amazon EKS proporciona dos tipos de información: información de configuración e información de actualización. La información de configuración identifica errores de configuración en la configuración de los nodos híbridos de EKS que podrían afectar a la funcionalidad del clúster o las cargas de trabajo. La información de actualización identifica los problemas que podrían afectar a su capacidad de actualizar a nuevas versiones de Kubernetes.

Para ver la lista de comprobaciones de información hechas y cualquier problema pertinente que Amazon EKS haya identificado, puede llamar a la operación de búsqueda en la Consola de administración de AWS, la AWS CLI, los AWS SDK y la API `ListInsights` de Amazon EKS.

Consulta de la información de configuración (consola)

1. Abra la consola de Amazon EKS.
2. En la lista de clústeres, elija el nombre del clúster de Amazon EKS del que desea ver la información.
3. Elija Supervisar clúster.
4. Elija la pestaña Estado del clúster.
5. En la tabla Información de configuración, verá las siguientes columnas:
   - Nombre: la comprobación realizada por Amazon EKS en relación con el clúster.
   - Estado de la información: una información con un estado de `Error` significa que hay un error de configuración que probablemente esté afectando a la funcionalidad del clúster. Una

información con un estado de `Warning` significa que la configuración no coincide con el enfoque documentado, pero que la funcionalidad del clúster podría funcionar si la configuró intencionadamente. Una información con el estado de `Passing` significa que Amazon EKS no encontró ningún problema relacionado con esta comprobación de información en su clúster.

- Versión: la versión aplicable.
- Hora de la última actualización: la hora en que se actualizó por última vez el estado de la información para este clúster.
- Descripción: información de la comprobación de información, que incluye la alerta y las acciones recomendadas para su corrección.

Consulta de la información de actualización (consola)

1. Abra la consola de Amazon EKS.
2. En la lista de clústeres, elija el nombre del clúster de Amazon EKS del que desea ver la información.
3. Elija Supervisar clúster.
4. Seleccione la pestaña Información sobre actualizaciones.
5. Para ver los datos más recientes, pulse Actualizar información y espere a que finalice la operación de actualización.
6. En la tabla Información sobre actualizaciones, verá las siguientes columnas:
   - Nombre: la comprobación realizada por Amazon EKS en relación con el clúster.
   - Estado de la información: una información con un estado de “Error” normalmente significa que la versión de Kubernetes afectada es N+1 de la versión actual del clúster, mientras que un estado de “Advertencia” significa que la información se aplica a una versión futura de Kubernetes N +2 o posterior. Una información con el estado “Aprobado” significa que Amazon EKS no ha encontrado ningún problema relacionado con esta comprobación de información en su clúster. Un estado de información “Desconocido” significa que Amazon EKS no puede determinar si su clúster se ve afectado por esta comprobación de información.
   - Versión: la versión de Kubernetes que la información comprobó para detectar posibles problemas.
   - Hora de la última actualización: la hora en que se actualizó por última vez el estado de la información para este clúster.
   - Hora de la última transición: la hora en que se modificó por última vez el estado de esta información.

- Descripción: información de la comprobación de información, que incluye la alerta y las acciones recomendadas para su corrección.

Consulta de la información del clúster (AWS CLI)

1. Para ver los datos más recientes, actualice la información de un clúster especificado. Lleve a cabo las siguientes modificaciones en el comando según sea necesario y, a continuación, ejecute el comando modificado.
   - Reemplace *region-code* por el código de la región de AWS.
   - Reemplace *my-cluster* por el nombre de su clúster.

   ```
   aws eks start-insight-refresh --region region-code --cluster-name my-cluster
   ```

2. Para hacer un seguimiento del estado de una actualización de información, ejecute el siguiente comando. Reemplace *my-cluster* por el nombre de su clúster.

   ```
   aws eks describe-insights-refresh --cluster-name my-cluster
   ```

   Un ejemplo de salida sería el siguiente.

   ```
   {
       "message": "Insights refresh is in progress",
       "status": "IN_PROGRESS",
       "startedAt": "2025-07-30T13:36:09-07:00"
   }
   ```

3. Enumere la información de un clúster especificado. Lleve a cabo las siguientes modificaciones en el comando según sea necesario y, a continuación, ejecute el comando modificado.
   - Reemplace *region-code* por el código de la región de AWS.
   - Reemplace *my-cluster* por el nombre de su clúster.

   ```
   aws eks list-insights --region region-code --cluster-name my-cluster
   ```

   Un ejemplo de salida sería el siguiente.

```
{
"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. Para obtener información descriptiva, ejecuta el siguiente comando. Lleve a cabo las siguientes modificaciones en el comando según sea necesario y, a continuación, ejecute el comando modificado.
   - Reemplace *region-code* por el código de la región de AWS.
   - Sustituya *a1b2c3d4-5678-90ab-cdef-EXAMPLE22222* por un ID de información recuperado de la lista de información del clúster.
   - Reemplace *my-cluster* por el nombre de su clúster.

```
aws eks describe-insight --region region-code --id a1b2c3d4-5678-90ab-cdef-
EXAMPLE22222 --cluster-name my-cluster
```

Un ejemplo de salida sería el siguiente.


```
{
  "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": [] },
    },
}
```


Actualización del clúster existente a la nueva versión de Kubernetes

Cuando hay una nueva versión de Kubernetes disponible en Amazon EKS, puede actualizar el clúster de Amazon EKS a la versión más reciente.

> **Important**
>
> Una vez que actualice un clúster, no podrá cambiarlo a una versión anterior. Antes de actualizar a una nueva versión de Kubernetes, le recomendamos que revise la información en Descripción del ciclo de vida de las versiones de Kubernetes en EKS y los pasos de actualización de este tema.

Las nuevas versiones de Kubernetes suelen presentar cambios significativos. Por ende, recomendamos que pruebe el comportamiento de las aplicaciones en la nueva versión de Kubernetes antes de realizar la actualización en los clústeres de producción. Para ello, puede crear un flujo de trabajo de integración continua con el fin de probar el comportamiento de la aplicación antes de pasar a una nueva versión de Kubernetes.

El proceso de actualización consiste en que Amazon EKS lance nodos de servidor de API nuevos con la versión actualizada de Kubernetes para sustituir a los existentes. Amazon EKS lleva a cabo una infraestructura estándar y una comprobación de estado de la disponibilidad del tráfico de red en estos nodos nuevos para verificar que funcionan según lo esperado. Sin embargo, una vez que haya iniciado la actualización del clúster, no podrá pausarla ni detenerla. Si cualquiera de estas comprobaciones falla, Amazon EKS revierte la implementación de la infraestructura y su clúster se mantiene en la versión anterior de Kubernetes. Las aplicaciones en ejecución no se ven afectadas y su clúster nunca queda en un estado no determinista o irrecuperable. Si fuese necesario, Amazon EKS realiza copias de seguridad de forma habitual a todos los clústeres administrados y existe un mecanismo de recuperación de clústeres. Evaluamos y mejoramos de forma constante nuestros procesos de administración de la infraestructura de Kubernetes.

Para actualizar el clúster, Amazon EKS requiere hasta cinco direcciones IP disponibles de las subredes que se especificaron cuando creó el clúster. Amazon EKS crea nuevas interfaces de red elástica de clúster (interfaces de red) en cualquiera de las subredes especificadas. Las interfaces de red se pueden crear en subredes diferentes a las que están las interfaces de red existentes, así que asegúrese de que las reglas del grupo de seguridad permitan la comunicación de clúster necesaria para cualquiera de las subredes que especificó al crear su clúster. Si alguna de las subredes especificadas al crear el clúster no existe, no tiene suficientes direcciones IP disponibles o no tiene reglas de grupo de seguridad que permitan la comunicación del clúster necesaria, la actualización puede tener errores.

Para garantizar que el punto de conexión del servidor de API de su clúster esté siempre accesible, Amazon EKS ofrece un plano de control de Kubernetes y lleva a cabo actualizaciones sucesivas de

las instancias del servidor de API durante las operaciones de actualización. Para tener en cuenta los cambios en las direcciones IP de las instancias del servidor de API que admiten su punto de conexión del servidor de API de Kubernetes, debe asegurarse de que los clientes de su servidor de API gestionen las reconexiones de manera eficaz. Versiones recientes de `kubectl` y las bibliotecas del cliente de Kubernetes que cuentan con soporte oficial llevan a cabo este proceso de reconexión de forma transparente.

> **Note**
>
> Para obtener más información sobre lo que implica una actualización de clúster, consulte Best Practices for Cluster Upgrades en la Guía de prácticas recomendadas de EKS. Este recurso le ayuda a planificar una actualización y a comprender la estrategia de actualización de un clúster.

Consideraciones para el modo automático de Amazon EKS

- La capacidad de computación del modo automático de Amazon EKS controla la versión de Kubernetes de los nodos. Después de actualizar el plano de control, el modo automático de EKS comenzará a actualizar de forma incremental los nodos administrados. El modo automático de EKS respeta los presupuestos de interrupción de pods.
- No es necesario actualizar manualmente las capacidades del modo automático de Amazon EKS, incluidas las capacidades de escalado automático de computación, almacenamiento en bloque y equilibrio de carga.

Resumen

A continuación, se ofrece un resumen general del proceso de la actualización del clúster de Amazon EKS:

1. Asegúrese de que el clúster tenga un estado que se pueda actualizar. Esto incluye comprobar las API de Kubernetes que utilizan los recursos implementados en el clúster, a fin de garantizar que el clúster no presente ningún problema de estado. Debe utilizar la información sobre actualizaciones de Amazon EKS al evaluar si su clúster está preparado para la actualización.
2. Actualice el plano de control a la siguiente versión secundaria (por ejemplo, de la 1.32 a la 1.33).
3. Actualice los nodos del plano de datos para que coincidan con los del plano de control.

4. Actualice cualquier aplicación adicional que se ejecute en el clúster (por ejemplo, `cluster-autoscaler`).
5. Actualice los complementos proporcionados por Amazon EKS, como los que se incluyen de forma predeterminada:
   - Versión recomendada de la CNI de Amazon VPC
   - Versión recomendada de CoreDNS
   - `kube-proxy` Versión recomendada de
6. Actualice todos los clientes que se comuniquen con el clúster (por ejemplo, `kubectl`).

Paso 1: preparación para la actualización

Compare la versión de Kubernetes de su plano de control de clúster con la versión de Kubernetes de sus nodos.

- Obtenga la versión de Kubernetes del plano de control de clúster.

```
kubectl version
```

- Obtenga la versión de Kubernetes de sus nodos. Este comando devuelve todos los nodos autoadministrados y administrados de Amazon EC2, Fargate e híbridos. Cada pod de Fargate aparece como su propio nodo.

```
kubectl get nodes
```

Antes de actualizar un plano de control a una nueva versión de Kubernetes, asegúrese de que la versión secundaria de Kubernetes de ambos notos administrados y de Fargate en el clúster debe ser la misma que la de la versión actual del plano de control. Por ejemplo, si el plano de control se ejecuta con la versión `1.29` y uno de los nodos con la versión `1.28`, debe actualizar los nodos a la versión `1.29` antes de actualizar el plano de control a la 1.30. Además, recomendamos actualizar los nodos autoadministrados y los nodos híbridos a la misma versión que el plano de control antes de actualizar el plano de control. Para obtener más información, consulte the section called “Actualización”, the section called “Métodos de actualización” y the section called “Actualización de nodos híbridos”. Si tiene nodos de Fargate con una versión secundaria inferior a la versión del plano de control, elimine primero el pod que representa el nodo. Luego, actualice su plano de control. Los pods restantes se actualizarán a la nueva versión después de volver a implementarlos.

Paso 2: revisión de las consideraciones de actualización

La información proporcionada por el clúster de Amazon EKS analiza automáticamente los clústeres en función de una lista de posibles problemas que afectan a la actualización de la versión de Kubernetes, como el uso obsoleto de la API de Kubernetes. Amazon EKS actualiza periódicamente la lista de comprobaciones de información que se deben realizar en función de las evaluaciones de los cambios en el proyecto de Kubernetes. Amazon EKS también actualiza la lista de verificación de información a medida que se introducen cambios en el servicio Amazon EKS junto con las nuevas versiones. Para obtener más información, consulte the section called “Información sobre clústeres”.

Consulte Deprecated API Migration Guide en la documentación de Kubernetes.

Revisión de la información sobre actualizaciones

Utilice la información sobre actualizaciones de Amazon EKS para identificar problemas. Para obtener más información, consulte the section called “Consulta de la información de actualización (consola)”.

Consideraciones detalladas

- Puesto que Amazon EKS ejecuta un plano de control de alta disponibilidad, puede actualizar solo una versión secundaria a la vez. Para obtener más información acerca de este requisito, consulte Política de compatibilidad de versiones y diferencia de versiones de Kubernetes. Supongamos que la versión del clúster actual es la `1.28` y quiere actualizarla a la `1.30`. Primero debe actualizar su clúster de versión `1.28` a la versión `1.29` y, a continuación, actualizar su clúster de versión `1.29` a la versión `1.30`.
- Revise la compatibilidad de versiones entre `kube-apiserver` de Kubernetes y el `kubelet` en sus nodos.
  - A partir de la versión de Kubernetes `1.28`, en `kubelet` puede haber hasta tres versiones secundarias anteriores a `kube-apiserver`. Consulte Política de compatibilidad de escalado entre versiones de Kubernetes.
  - Si el `kubelet` de sus nodos administrados y de Fargate corresponde a la versión de Kubernetes `1.25` o una más reciente, puede actualizar su clúster hasta tres versiones más avanzadas sin necesidad de actualizar la versión de `kubelet`. Por ejemplo, si `kubelet` está en la versión `1.25`, puede actualizar la versión del clúster de Amazon EKS de `1.25` a `1.26` a `1.27` y a `1.28`, mientras que `kubelet` permanezca en la versión `1.25`.
- Como práctica recomendada antes de iniciar una actualización, asegúrese de que el `kubelet` de sus nodos esté en la misma versión de Kubernetes que la de su plano de control.

- Si el clúster está configurado con una versión del complemento CNI de Amazon VPC para Kubernetes anterior a la `1.8.0`, le recomendamos actualizar el complemento a la versión más reciente antes de actualizar el clúster. Para actualizar el complemento, consulte the section called “CNI de Amazon VPC”.
- Puede hacer una copia de seguridad del clúster de Amazon EKS para poder restaurar el estado del clúster de Amazon EKS y el almacenamiento persistente en caso de que se produzcan errores durante el proceso de actualización. Consulte the section called “ AWS Backup”

Paso 3: actualización del plano de control del clúster

> **Important**
>
> Amazon EKS ha revertido temporalmente una característica que obligaba a utilizar una marca `--force` para actualizar el clúster cuando se producían determinados problemas de conocimiento del clúster. Para obtener más información, consulte Temporary rollback of enforcing upgrade insights on update cluster version en GitHub.
> Amazon EKS actualiza la información de un clúster 24 horas después de la “hora de la última actualización”. Puede comparar la hora en que se ha abordado un problema con la “hora de la última actualización” de la información del clúster.
> Además, el estado de la información puede tardar hasta 30 días en actualizarse después de abordar el uso obsoleto de la API. La información sobre actualizaciones siempre busca el uso obsoleto de la API durante un periodo continuo de 30 días.

Puede enviar la solicitud para actualizar la versión de su plano de control de Amazon EKS mediante:

- eksctl
- La consola de AWS
- la CLI de AWS

Actualizar clúster: eksctl

En este procedimiento, se requiere la versión `0.215.0` o posterior de `eksctl`. Puede verificar la versión con el siguiente comando:

```
eksctl version
```

Para obtener instrucciones sobre cómo instalar y actualizar `eksctl`, consulte Instalación en la documentación de `eksctl`.

Actualice la versión de Kubernetes de su plano de control de Amazon EKS. Reemplace `<cluster-name>` por el nombre del clúster. Reemplace `<version-number>` por el número de versión compatible con Amazon EKS al que desea actualizar el clúster. Para ver una lista de los números de versión compatibles, consulte Versiones compatibles con Amazon EKS.

```
eksctl upgrade cluster --name <cluster-name> --version <version-number> --approve
```

La actualización puede tardar varios minutos en completarse.

Continuar con the section called “Paso 4: actualización de los componentes del clúster”.

Actualizar clúster: consola de AWS

1. Abra la consola de Amazon EKS.
2. Seleccione Actualizar ahora para el clúster que desee actualizar.
3. Seleccione la versión a la que desea actualizar el clúster y elija Actualizar.
4. La actualización puede tardar varios minutos en completarse. Continuar con the section called “Paso 4: actualización de los componentes del clúster”.

Actualizar clúster: AWS CLI

1. Compruebe que la CLI de AWS esté instalada y que haya iniciado sesión. Para obtener más información, consulte Instalación o actualización de la versión más reciente de la CLI de AWS.
2. Actualice el clúster de Amazon EKS con el siguiente comando de la AWS CLI. Reemplace `<cluster-name>` y `<region-code>` del clúster que desee actualizar. Reemplace `<version-number>` por el número de versión compatible con Amazon EKS al que desea actualizar el clúster. Para ver una lista de los números de versión compatibles, consulte Versiones compatibles con Amazon EKS.

   ```
   aws eks update-cluster-version --name <cluster-name> \
     --kubernetes-version <verion-number> --region <region-code>
   ```

   Un ejemplo de salida sería el siguiente.

   ```
   {
   ```

```
    "update": {
        "id": "<update-id>",
        "status": "InProgress",
        "type": "VersionUpdate",
        "params": [
            {
                "type": "Version",
                "value": "<version-number>"
            },
            {
                "type": "PlatformVersion",
                "value": "eks.1"
            }
        ],
[...]
        "errors": []
    }
```

3. La actualización puede tardar varios minutos en completarse. Monitoree el estado de la actualización del clúster con el siguiente comando. Además de usar el mismo `<cluster-name>` y `<region-code>`, use el `<update-id>` que devolvió el comando anterior.

```
aws eks describe-update --name <cluster-name> \
   --region <region-code> --update-id <update-id>
```

   Cuando se muestra el estado `Successful`, la actualización se ha completado.

4. Continuar con the section called “Paso 4: actualización de los componentes del clúster”.

Paso 4: actualización de los componentes del clúster

1. Una vez que se complete la actualización del clúster, actualice los nodos a la misma versión secundaria de Kubernetes de su clúster actualizado. Para obtener más información, consulte the section called “Métodos de actualización”, the section called “Actualización” y the section called “Actualización de nodos híbridos”. Los pods nuevos que se lancen en Fargate tienen una versión de `kubelet` que coinciden con la versión del clúster. Los pods de Fargate existentes no cambian.
2. (Opcional) Si implementó el Cluster Autoscaler de Kubernetes en el clúster antes de actualizar este último, actualice dicho Cluster Autoscaler a la versión más reciente que coincida con la versión principal y secundaria de Kubernetes a las que actualizó.

a. Abra la página de versiones del Escalador automático de clústeres en un navegador web y busque la versión más reciente del Escalador automático de clústeres que coincida con la versión principal y secundaria de Kubernetes de su clúster. Por ejemplo, si la versión de Kubernetes del clúster es `1.30`, busque la última versión del escalador automático del clúster que comience por `1.30`. Registre el número de versión semántica (`1.30.n`, por ejemplo) de esa versión para usarlo en el siguiente paso.

b. Establezca la etiqueta de la imagen del Escalador automático de clústeres en la versión que ha registrado en el paso anterior con el siguiente comando. Si es necesario, reemplace `X.XX.X` por su propio valor.

```
kubectl -n kube-system set image deployment.apps/cluster-autoscaler cluster-
autoscaler=registry.k8s.io/autoscaling/cluster-autoscaler:vX.XX.X
```

3. (Solo para clústeres con nodos de GPU) Si el clúster tiene grupos de nodos compatibles con GPU (por ejemplo, `p3.2xlarge`), debe actualizar el DaemonSet del complemento del dispositivo NVIDIA para Kubernetes de su clúster. Reemplace <vX.X.X> con la versión Plugin de dispositivo NVidia/K8S deseada antes de ejecutar el siguiente comando.

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/<vX.X.X>/
deployments/static/nvidia-device-plugin.yml
```

4. Actualice los complementos CNI de Amazon VPC para Kubernetes, CoreDNS y `kube-proxy`. Recomendamos actualizar los complementos a las versiones mínimas que figuran en los tokens de las cuentas de servicio.

   - Si está usando complementos de Amazon EKS, seleccione Clústeres en la consola de Amazon EKS y, a continuación, seleccione el nombre del clúster que actualizó en el panel de navegación izquierdo. Las notificaciones aparecen en la consola. Le informan que hay una versión nueva disponible para cada complemento que tenga una actualización disponible. Para actualizar un complemento, seleccione la pestaña Complementos. En uno de los cuadros de un complemento que tenga una actualización disponible, seleccione Actualizar ahora, seleccione una versión disponible y, a continuación, seleccione Actualizar.
   - Como alternativa, puede utilizar la AWS CLI o `eksctl` para actualizar los complementos. Para obtener más información, consulte the section called “Cómo actualizar un complemento”.

5. De ser necesario, actualice su versión de `kubectl`. Debe utilizar una versión de `kubectl` con una diferencia de versión de menos de un número que el plano del control del clúster de Amazon EKS.

Actualización a una versión anterior de Kubernetes en un clúster de Amazon EKS

No puede actualizar a una versión anterior de Kubernetes en un clúster de Amazon EKS. En su lugar, cree un clúster nuevo en una versión anterior de Amazon EKS y migre las cargas de trabajo.

Eliminar un clúster

Cuando termine de utilizar un clúster de Amazon EKS, debe eliminar los recursos asociados para no incurrir en costos innecesarios.

Puede eliminar un clúster mediante `eksctl`, Consola de administración de AWS o AWS CLI.

Consideraciones

- Si recibe un error porque se ha eliminado el creador del clúster, consulte este artículo para resolver el problema.
- Los recursos de Amazon Managed Service para Prometheus están fuera del ciclo de vida del clúster y deben mantenerse por fuera del clúster. Al eliminar el clúster, asegúrese de eliminar, también, cualquier raspador para reducir los costos aplicables. Para más información, consulte Búsqueda y eliminación de rapsadores en la Guía de usuario de Amazon Managed Service para Prometheus.
- Para eliminar un clúster conectado, consulte . the section called “Anulación del registro de un clúster”

Consideraciones para el modo automático de EKS

- Se eliminarán todos los nodos de modo automático de EKS, incluidas las instancias administradas por EC2.
- Se eliminarán todos los equilibradores de carga

Para obtener más información, consulte the section called “Cómo desactivar el modo automático de EKS”.

Eliminación del clúster (eksctl)

En este procedimiento, se requiere la versión `0.215.0` o posterior de `eksctl`. Puede verificar la versión con el siguiente comando:

```
eksctl version
```

Para obtener instrucciones sobre cómo instalar o actualizar `eksctl`, consulte Instalación en la documentación de `eksctl`.

1. Enumere todos los servicios que se ejecutan en el clúster.

   ```
   kubectl get svc --all-namespaces
   ```

   a. Eliminación de los servicios que tengan asociado un valor `EXTERNAL-IP`. Estos servicios se presentan por medio de un balanceador de carga de Elastic Load Balancing y debe eliminarlos en Kubernetes para que el balanceador y los recursos asociados se lancen correctamente. Sustituya *`service-name`* por el nombre de cada servicio de la lista, tal y como se describe.

      ```
      kubectl delete svc service-name
      ```

2. Elimine el clúster y sus nodos asociados con el siguiente comando, al reemplazar*`prod`* por el nombre de su clúster.

   ```
   eksctl delete cluster --name prod
   ```

   Salida:

   ```
   [#]  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
   ```

Eliminar un clúster (Consola de AWS)

1. Enumere todos los servicios que se ejecutan en el clúster.

```
kubectl get svc --all-namespaces
```

2. Eliminación de los servicios que tengan asociado un valor `EXTERNAL-IP`. Estos servicios se presentan por medio de un balanceador de carga de Elastic Load Balancing y debe eliminarlos en Kubernetes para que el balanceador y los recursos asociados se lancen correctamente. Sustituya *`service-name`* por el nombre de cada servicio de la lista, tal y como se describe.

```
kubectl delete svc service-name
```

3. Eliminación de todos los grupos de nodos y perfiles de Fargate.
   a. Abra la consola de Amazon EKS.
   b. En el panel de navegación izquierdo, seleccione Clústeres de Amazon EKS y, a continuación, en la lista de clústeres con pestañas, seleccione el nombre del clúster que desea eliminar.
   c. Elija la pestaña Compute (Informática) y elija un grupo de nodos para eliminar. Elija Delete (Eliminar), introduzca el nombre del grupo de nodos y, a continuación, elija Delete (Eliminar). Eliminación de todos los grupos de nodos del clúster.

> **Note**
>
> Los grupos de nodos enumerados solo son los grupos de nodos administrados.

   d. Seleccione un Fargate Profile (Perfil de Fargate) para eliminar, seleccione Delete (Eliminar), ingrese el nombre del perfil y, a continuación, seleccione Delete (Eliminar). Eliminación de todos los perfiles de Fargate en el clúster.
4. Eliminación de todas las pilas de AWS CloudFormation de nodos autoadministrados.
   a. Abra la Consola de AWS CloudFormation.
   b. Seleccione la pila de nodos que desea eliminar y, luego, elija Delete (Eliminar).
   c. En el cuadro de diálogo de confirmación Delete stack (Eliminar pila), elija Delete stack (Eliminar pila). Eliminación de todas las pilas de nodos autoadministradas del clúster.
5. Elimine el clúster.
   a. Abra la consola de Amazon EKS.
   b. Seleccione el clúster que desea eliminar y elija Delete (Eliminar).

c. En la pantalla de confirmación de eliminación del clúster, elija Delete (Eliminar).

6. (Opcional) Eliminación de la pila de la VPC de AWS CloudFormation.
   a. Abra la Consola de AWS CloudFormation.
   b. Seleccione la pila de VPC que desea eliminar y, luego, elija Delete (Eliminar).
   c. En el cuadro de diálogo de confirmación Eliminar pila, elija Eliminar pila.

Eliminación de un clúster (AWS CLI)

1. Enumere todos los servicios que se ejecutan en el clúster.

```
kubectl get svc --all-namespaces
```

2. Eliminación de los servicios que tengan asociado un valor `EXTERNAL-IP`. Estos servicios se presentan por medio de un balanceador de carga de Elastic Load Balancing y debe eliminarlos en Kubernetes para que el balanceador y los recursos asociados se lancen correctamente. Sustituya *`service-name`* por el nombre de cada servicio de la lista, tal y como se describe.

```
kubectl delete svc service-name
```

3. Eliminación de todos los grupos de nodos y perfiles de Fargate.
   a. Enumere los grupos de nodos del clúster con el siguiente comando.

```
aws eks list-nodegroups --cluster-name my-cluster
```

> **Note**
>
> Los grupos de nodos enumerados son solo los grupos de nodos administrados.

   b. Eliminación de cada grupo de nodos con el siguiente comando. Eliminación de todos los grupos de nodos del clúster.

```
aws eks delete-nodegroup --nodegroup-name my-nodegroup --cluster-name my-cluster
```

   c. Enumere los perfiles de Fargate del clúster con el siguiente comando.

```
aws eks list-fargate-profiles --cluster-name my-cluster
```

d. Eliminación de cada perfil de Fargate con el siguiente comando. Eliminación de todos los perfiles de Fargate en el clúster.

```
aws eks delete-fargate-profile --fargate-profile-name my-fargate-profile --
cluster-name my-cluster
```

4. Eliminación de todas las pilas de AWS CloudFormation de nodos autoadministrados.
   a. Muestre las pilas de AWS CloudFormation disponibles con el siguiente comando. Busque el nombre de la plantilla de nodos en la salida resultante.

```
aws cloudformation list-stacks --query "StackSummaries[].StackName"
```

   b. Eliminación de cada pila de nodos con el siguiente comando y reemplace *node-stack* por el nombre de su pila de nodos. Eliminación de todas las pilas de nodos autoadministradas del clúster.

```
aws cloudformation delete-stack --stack-name node-stack
```

5. Eliminación del clúster con el siguiente comando, sustituyendo *my-cluster* por el nombre de su clúster.

```
aws eks delete-cluster --name my-cluster
```

6. (Opcional) Eliminación de la pila de AWS CloudFormation de la VPC.
   a. Muestre las pilas de AWS CloudFormation disponibles con el siguiente comando. Busque el nombre de la plantilla de VPC en la salida resultante.

```
aws cloudformation list-stacks --query "StackSummaries[].StackName"
```

   b. Elimine la pila de VPC con el siguiente comando, sustituyendo *my-vpc-stack* por el nombre de la pila de VPC.

```
aws cloudformation delete-stack --stack-name my-vpc-stack
```

Protección de los clústeres de EKS contra la eliminación accidental

La eliminación accidental de un clúster de EKS puede afectar las operaciones del clúster de Kubernetes.

Ahora puede proteger los clústeres de EKS contra la eliminación accidental. Si habilita la protección contra la eliminación en un clúster, primero debe deshabilitar la protección contra eliminación antes de eliminarlo.

El objetivo de la protección contra la eliminación es evitar accidentes. Debe restringir cuidadosamente quién está autorizado a eliminar clústeres.

Si intenta eliminar un clúster activo que tiene activada la protección contra la eliminación, recibirá un`InvalidRequestException`.

> **Important**
>
> Si habilita la protección contra la eliminación en un clúster, debe tener los permisos de IAM UpdateClusterConfig y DeleteCluster para eliminar primero la protección contra la eliminación y, finalmente, eliminar el clúster.

> **Note**
>
> Si el estado del clúster es de creación, error o eliminación, puede eliminarlo aunque la protección contra eliminaciones esté activada.

Habilitación de la protección contra la eliminación para un clúster existente

Solo puede ejecutar esto en un clúster en estado activo.

```
aws eks update-cluster-config --name <cluster-name> --region <aws-region> --deletion-protection
```

Deshabilitación de la protección contra la eliminación para un clúster existente

```
aws eks update-cluster-config --name <cluster-name> --region <aws-region> --no-deletion-protection
```

Punto de conexión del servidor de API del clúster

Esto lo ayudará a habilitar el acceso privado al punto de conexión del servidor de API de Kubernetes de su clúster de Amazon EKS y a limitar, o a desactivar por completo, el acceso público desde Internet.

Cuando se crea un clúster nuevo, Amazon EKS crea un punto de conexión para el servidor de la API de Kubernetes administrado que utiliza a fin de comunicarse con su clúster (mediante herramientas de administración de Kubernetes como, por ejemplo, `kubectl`). De forma predeterminada, este punto de conexión del servidor de API es público en Internet y el acceso al servidor de API está protegido mediante una combinación de AWS Identity and Access Management (IAM) y el Control de acceso basado en rol (RBAC) nativo de Kubernetes. Este punto de conexión se conoce como punto de conexión público del clúster. También hay un punto de conexión privado del clúster. Para obtener más información sobre el punto de conexión privado del clúster, consulte la siguiente sección the section called “Punto de conexión privado del clúster”.

formato de punto de conexión del clúster **`IPv6`**

EKS crea un punto de conexión único de doble pila con el siguiente formato para los nuevos clústeres `IPv6` que se creen después de octubre de 2024. Un clúster IPv6 es un clúster en el que selecciona `IPv6` en la configuración de la familia IP (`ipFamily`) del clúster.

Example

AWS

Punto de conexión público o privado del clúster de EKS: `eks-cluster.`*`region`*`.api.aws`

AWS GovCloud (US)

Punto de conexión público o privado del clúster de EKS: `eks-cluster.`*`region`*`.api.aws`

Amazon Web Services in China

Punto de conexión público o privado del clúster de EKS: `eks-cluster.`*`region`*`.api.amazonwebservices.com.cn`

> **Note**
>
> El punto de conexión de clúster de doble pila se introdujo en octubre de 2024. Para obtener más información acerca de los clústeres `IPv6`, consulte the section called “IPv6”. En su

lugar, los clústeres creados antes de octubre de 2024 utilizan el siguiente formato de punto de conexión.

Formato de punto de conexión del clúster **IPv4**

EKS crea un punto de conexión único en el siguiente formato para cada clúster en el que se selecciona `IPv4` en la configuración de familia IP (ipFamily) del clúster:

Example

AWS

Punto de conexión público o privado del clúster de EKS `eks-cluster.`*`region`*`.eks.amazonaws.com`

AWS GovCloud (US)

Punto de conexión público o privado del clúster de EKS `eks-cluster.`*`region`*`.eks.amazonaws.com`

Amazon Web Services in China

Punto de conexión público o privado del clúster de EKS `eks-cluster.`*`region`*`.amazonwebservices.com.cn`

> **Note**
>
> Antes de octubre de 2024, los clústeres `IPv6` también utilizaban este formato de punto de conexión. En el caso de estos clústeres, tanto en el punto de conexión público como en el privado, solo se resuelven direcciones `IPv4` a través de este punto de conexión.

Punto de conexión privado del clúster

Puede habilitar el acceso privado al servidor de la API de Kubernetes para que toda la comunicación entre los nodos y el servidor de la API permanezcan dentro de su VPC. Puede limitar las direcciones IP que pueden acceder a su servidor de API desde Internet o desactivar por completo el acceso a Internet al servidor de API.

> **Note**
>
> Dado que este punto de conexión es para el servidor de la API de Kubernetes y no un punto de conexión de AWS PrivateLink tradicional que sirve para comunicarse con una API de AWS, no aparece como un punto de conexión en la consola de Amazon VPC.

Al habilitar el acceso privado al punto de conexión para el clúster, Amazon EKS crea una zona alojada privada de Route 53 en su nombre y la asocia a la VPC de su clúster. Esta zona alojada privada se administra mediante Amazon EKS y no aparece en los recursos de Route 53 de su cuenta. Para que la zona privada alojada dirija el tráfico adecuadamente a su servidor de API, su VPC debe tener `enableDnsHostnames` y `enableDnsSupport` establecidos en `true` y las opciones de DHCP establecidas para su VPC deben incluir `AmazonProvidedDNS` en su lista de servidores de nombres de dominios. A fin de obtener más información, consulte Actualización de soporte de DNS para su VPC en la guía del usuario de Amazon VPC.

Puede definir los requisitos de acceso al punto de conexión del servidor de la API al crear un nuevo clúster; puede actualizar el acceso al punto de conexión del servidor de la API para un clúster en cualquier momento.

Modificar el acceso al punto de conexión del clúster

Utilice los procedimientos de esta sección para modificar el acceso al punto de conexión para un clúster existente. En la siguiente tabla se muestran las combinaciones de acceso al punto de conexión del servidor de la API y sus comportamientos asociados.

Acceso público al punto de conexión	Acceso privado al punto de conexión	Comportamiento
Habilitado	Deshabilitado	• Este es el comportamiento predeterminado para los clústeres de Amazon EKS nuevos. • Las solicitudes de la API de Kubernetes que provienen de dentro de la VPC de su clúster (como comunicación desde el nodo al plano de control) dejan la VPC, pero no la red de Amazon.

Acceso público al punto de conexión	Acceso privado al punto de conexión	Comportamiento
		• Se puede acceder al servidor de la API del clúster desde Internet. Si lo desea, puede utilizar la lista de CIDR de acceso público para controlar el acceso al punto de conexión público mediante una lista de bloques de CIDR. Si limita el acceso a bloques de CIDR específicos, le recomendamos que active también el punto de conexión privado o se asegure de que los bloques de CIDR que especifique incluyan las direcciones desde las que los nodos y los pods de Fargate (si los utiliza) acceden al punto de conexión público.
Habilitado	Habilitado	• Las solicitudes de la API de Kubernetes de dentro de la VPC de su clúster (como comunicación desde el nodo al plano de control) utilizan el punto de conexión de VPC privado. Utilice las reglas del grupo de seguridad del clúster para controlar el acceso al punto de conexión privado. • Se puede acceder al servidor de la API del clúster desde Internet. Si lo desea, puede utilizar la lista de CIDR de acceso público para controlar el acceso al punto de conexión público mediante una lista de bloques de CIDR. • Si utiliza nodos híbridos con el clúster de Amazon EKS, no se recomienda tener activado el acceso a puntos de conexión de clúster públicos y privados. Dado que los nodos híbridos se ejecutan fuera de la VPC, resolverá n el punto de conexión del clúster a las direcciones IP públicas. Se recomienda utilizar el acceso a puntos de conexión de clúster público o privado para los clústeres con nodos híbridos.

Acceso público al punto de conexión	Acceso privado al punto de conexión	Comportamiento
Deshabilitado	Habilitado	• Todo el tráfico al servidor de la API del clúster debe proceder desde dentro de la VPC de su clúster o de una red conectada. • No hay acceso público al servidor de la API desde Internet. Cualquier comando `kubectl` debe provenir de dentro de la VPC o de una red conectada. Para ver las opciones de conectividad, consulte the section called “Acceso a un servidor de API solo privado”. • Utilice las reglas del grupo de seguridad del clúster para controlar el acceso al punto de conexión privado. Tenga en cuenta que los CIDR de acceso público no afectan al punto de conexión privado. • Los servidores DNS públicos resuelven el punto de conexión del servidor de API del clúster en una dirección IP privada desde la VPC. En el pasado, el punto de conexión se podía resolver desde dentro de la VPC. Si el punto de conexión no se resuelve en una dirección IP privada dentro de la VPC para un clúster existente, puede: • Habilitar el acceso público y, a continuación, volver a deshabilitarlo. Solo tiene que hacerlo una vez para un clúster y el punto de conexión se resolverá en una dirección IP privada a partir de ese momento. • Actualice el clúster.

Controles de acceso de puntos de conexión

Tenga en cuenta que cada uno de los siguientes métodos para controlar el acceso de puntos de conexión solo afecta al punto de conexión correspondiente.

Grupo de seguridad de clúster

El grupo de seguridad del clúster controla dos tipos de conexiones: las conexiones a la API de kubelet y al punto de conexión privado. Las conexiones a la API `kubelet` se utilizan en los comandos `kubectl attach`, `kubectl cp`, `kubectl exec`, `kubectl logs` y `kubectl port-forward`. El grupo de seguridad del clúster no afecta al punto de conexión público.

Acceso público: CIDR

Los CIDR de acceso público controlan el acceso al punto de conexión público mediante una lista de bloques de CIDR. Tenga en cuenta que los CIDR de acceso público no afectan al punto de conexión privado. Los CIDR de acceso público se comportan de forma diferente en los clústeres `IPv6` e `IPv4` en función de la fecha en que se crearon, como se describe a continuación:

Bloques de CIDR en el punto de conexión público (clúster **`IPv6`**)

Puede agregar bloques de CIDR `IPv6` y `IPv4` al punto de conexión público de un clúster `IPv6`, ya que el punto de conexión público es de doble pila. Esto solo se aplica a los clústeres nuevos con `ipFamily` configurado como `IPv6` que creó en octubre de 2024 o después. Puede identificar estos clústeres por el nuevo nombre de dominio del punto de conexión `api.aws`.

Bloques de CIDR en el punto de conexión público (clúster **`IPv4`**)

Puede agregar bloques de CIDR `IPv4` al punto de conexión público de un clúster `IPv4`. No puede agregar bloques de CIDR `IPv6` al punto de conexión público de un clúster `IPv4`. Si lo intenta, EKS devuelve el siguiente mensaje de error: `The following CIDRs are invalid in publicAccessCidrs`

Bloques de CIDR en el punto de conexión público (clúster **`IPv6`** creado antes de octubre de 2024)

Puede agregar bloques de CIDR `IPv4` al punto de conexión público de los clústeres `IPv6` antiguos que creó antes de octubre de 2024. Puede identificar estos clústeres por el punto de conexión `eks.amazonaws.com`. No puede agregar bloques de CIDR `IPv6` al punto de conexión público de los clústeres `IPv6` antiguos que creó antes de octubre de 2024. Si lo intenta, EKS devuelve el siguiente mensaje de error: `The following CIDRs are invalid in publicAccessCidrs`

Acceso a un servidor de API solo privado

Si ha deshabilitado el acceso público al punto de conexión del servidor de API de Kubernetes del clúster, solo puede obtener acceso al servidor de API desde dentro de la VPC o desde una

red conectada. Hay varias formas de obtener acceso al punto de conexión del servidor de API de Kubernetes:

Red conectada

Conecte su red a la VPC con una puerta de enlace de tránsito de AWS u otra opción de conectividad y, a continuación, utilice un equipo en la red conectada. Debe asegurarse de que el grupo de seguridad del plano de control de Amazon EKS tiene reglas para permitir el tráfico de entrada en el puerto 443 desde la red conectada.

Host bastión de Amazon EC2

Puede lanzar una instancia de Amazon EC2 en una subred pública de la VPC del clúster y, a continuación, iniciar sesión mediante SSH en esa instancia para ejecutar comandos de `kubectl`. Para obtener más información, consulte Hosts bastión de Linux en AWS. Debe asegurarse de que el grupo de seguridad del plano de control de Amazon EKS tiene reglas para permitir el tráfico de entrada en el puerto 443 desde su host bastión. Para obtener más información, consulte the section called “Requisitos del grupo de seguridad”.

Cuando configure `kubectl` para el host bastión, asegúrese de utilizar las credenciales de AWS que ya están asignadas a su configuración de RBAC del clúster o agregue la entidad principal de IAM que utilizará el bastión a la configuración de RBAC antes de eliminar el acceso público al punto de conexión. Para obtener más información, consulte the section called “Acceso a la API de Kubernetes” y the section called “Acceso denegado o no autorizado (`kubectl`)”.

IDE de AWS Cloud9

AWS Cloud9 es un entorno de desarrollo integrado (IDE) basado en la nube que permite escribir, ejecutar y depurar su código con solo un navegador. Puede crear un IDE de AWS Cloud9 en la VPC de su clúster y utilizar el IDE para comunicarse con el clúster. Para obtener más información, consulte Creación de un entorno en AWS Cloud9. Debe asegurarse de que el grupo de seguridad del plano de control de Amazon EKS tiene reglas para permitir el tráfico de entrada en el puerto 443 desde el grupo de seguridad de IDE. Para obtener más información, consulte the section called “Requisitos del grupo de seguridad”.

Cuando configure `kubectl` para el IDE de AWS Cloud9, asegúrese de utilizar las credenciales de AWS que ya están asignadas a su configuración de RBAC del clúster o agregue la entidad principal de IAM que utilizará el IDE a la configuración de RBAC antes de eliminar el acceso público al punto de conexión. Para obtener más información, consulte the section called “Acceso a la API de Kubernetes” y the section called “Acceso denegado o no autorizado (`kubectl`)”.

Edite esta página en GitHub

Configuración de acceso de la red al punto de conexión del servidor de API del clúster

Puede modificar el acceso al punto de conexión del servidor de la API del clúster mediante la Consola de administración de AWS o la AWS CLI en las secciones siguientes.

Configuración del acceso al punto de conexión: consola de AWS

1. Abra la consola de Amazon EKS.
2. Elija el nombre del clúster para mostrar la información del clúster.
3. Elija la pestaña Redes y, a continuación, elija Administrar el acceso a los puntos de conexión.
4. Para el acceso privado, decida si desea habilitar o deshabilitar el acceso privado al punto de conexión del servidor de API de Kubernetes del clúster. Si habilita el acceso privado, las solicitudes de la API de Kubernetes que provengan desde dentro de la VPC del clúster utilizan el punto de conexión de VPC privada. Debe habilitar el acceso privado para deshabilitar el acceso público.
5. Para el acceso público, decida si desea habilitar o deshabilitar el acceso público al punto de conexión del servidor de API de Kubernetes del clúster. Si deshabilita el acceso público, el servidor de la API de Kubernetes del clúster solo puede recibir solicitudes que provengan desde dentro de la VPC del clúster.
6. (Opcional) Si ha habilitado el acceso público, puede especificar qué direcciones de Internet pueden comunicarse con el punto de conexión público. Seleccione Advanced Settings (Configuración avanzada). Introduzca un bloque de CIDR, como *`203.0.113.5/32`*. El bloque no puede incluir direcciones reservadas. Puede introducir bloques adicionales seleccionando Add Source (Agregar origen). Hay un número máximo de bloques de CIDR que puede especificar. Para obtener más información, consulte the section called “Service Quotas”. Si no especifica ningún bloque, el punto de conexión del servidor de API público recibe solicitudes de todas las direcciones IP tanto `IPv4` (`0.0.0.0/0`) como `IPv6` (`::/0`) para el clúster `IPv6` de doble pila. Si restringe el acceso a su punto de conexión público mediante bloques de CIDR, le recomendamos activar también el acceso al punto de conexión privado para que los nodos y los Pods de Fargate (si los utiliza) puedan comunicarse con el clúster. Si el punto de conexión privado no está habilitado, los orígenes de CIDR del punto de conexión de acceso público deben incluir los orígenes de salida de la VPC. Por ejemplo, si tiene un nodo en una subred privada que se comunica con Internet a través de una puerta de enlace NAT, deberá agregar la dirección IP

saliente de la puerta de enlace NAT como parte de un bloque de CIDR permitido en su punto de conexión público.

7. Elija Update (Actualizar) para finalizar.

Configuración del acceso al punto de conexión: AWS CLI

Complete los siguientes pasos con la versión `1.27.160` o posterior de la CLI de AWS. Puede comprobar su versión actual con `aws --version`. Para instalar o actualizar la AWS CLI, consulte Instalación de la AWS CLI.

1. Actualice el acceso al punto de conexión del servidor de la API del clúster con el siguiente comando de la AWS CLI. Sustituya el nombre de su clúster y los valores de acceso de punto de conexión deseados. Si configura el `endpointPublicAccess=true`, podrá introducir un solo bloque de CIDR o una lista separada por comas de bloques de CIDR para `publicAccessCidrs`. Los bloques no pueden incluir direcciones reservadas. Si especifica bloques de CIDR, el punto de conexión del servidor de API público solo recibirá solicitudes de los bloques enumerados. Hay un número máximo de bloques de CIDR que puede especificar. Para obtener más información, consulte the section called “Service Quotas”. Si restringe el acceso a su punto de conexión público mediante bloques de CIDR, se recomienda habilitar también el acceso al punto de conexión privado para que los nodos y los Pods de Fargate (si los utiliza) puedan comunicarse con el clúster. Si el punto de conexión privado no está habilitado, los orígenes de CIDR del punto de conexión de acceso público deben incluir los orígenes de salida de la VPC. Por ejemplo, si tiene un nodo en una subred privada que se comunica con Internet a través de una puerta de enlace NAT, deberá agregar la dirección IP saliente de la puerta de enlace NAT como parte de un bloque de CIDR permitido en su punto de conexión público. Si no especifica ningún bloque de CIDR, el punto de conexión del servidor de API público recibe solicitudes de todas las direcciones IP (0.0.0.0/0) y `IPv6` (`::/0`) para el clúster `IPv6` de doble pila.

> **Note**
>
> El siguiente comando habilita el acceso privado y público desde una única dirección IP al punto de conexión del servidor de API. Reemplace *`203.0.113.5/32`* por un único bloque de CIDR o una lista separada por comas de bloques de CIDR a los que desea restringir el acceso a la red.

```
aws eks update-cluster-config \
```

```
    --region region-code \
    --name my-cluster \
    --resources-vpc-config
 endpointPublicAccess=true,publicAccessCidrs="203.0.113.5/32",endpointPrivateAccess=true
```

Un ejemplo de salida sería el siguiente.

```
{
    "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. Monitoree el estado de la actualización del acceso al punto de conexión con el siguiente comando, utilizando el nombre del clúster y el ID de actualización devueltos por el comando anterior. Su actualización se habrá completado cuando el estado mostrado sea `Successful`.

```
aws eks describe-update \
    --region region-code \
    --name my-cluster \
    --update-id e6f0905f-a5d4-4a2a-8c49-EXAMPLE00000
```

Un ejemplo de salida sería el siguiente.

```
{
    "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": []
    }
}
```


Edite esta página en GitHub

Implementación de nodos de Windows en clústeres de EKS

Obtenga información sobre cómo habilitar y administrar la compatibilidad con Windows para el clúster de Amazon EKS para ejecutar contenedores de Windows junto con contenedores de Linux.

Consideraciones

Antes de implementar nodos de Windows, tenga en cuenta lo siguiente.

- El modo automático de EKS no es compatible con los nodos de Windows
- Puede utilizar redes de host en nodos de Windows mediante Pods `HostProcess`. Para obtener más información, consulte Create a Windows HostProcessPod en la documentación de Kubernetes.

- Los clústeres de Amazon EKS deben contener uno o varios nodos de Linux o Fargate para ejecutar pods del sistema principal que solo se ejecutan en Linux, como CoreDNS.
- Los registros de eventos `kubelet` y `kube-proxy` se redirigen al registro de eventos de `EKS Windows` y se establecen en un límite de 200 MB.
- No puede usar Asignación de los grupos de seguridad a pods individuales con pods que se ejecuten en nodos de Windows.
- No puede usar redes personalizadas con nodos de Windows.
- No puede usar `IPv6` con nodos de Windows.
- Los nodos de Windows admiten una interfaz de red elástica por nodo. De forma predeterminada, la cantidad de pods que puede ejecutar por nodo de Windows es igual a la cantidad de direcciones IP disponibles por interfaz de red elástica para el tipo de instancia del nodo, menos uno. Para obtener más información, consulte Direcciones IP por interfaz de red por tipo de instancia en la Guía del usuario de Amazon EC2.
- En un clúster de Amazon EKS, un único servicio con un equilibrador de carga puede admitir hasta 1024 pods de backend. Cada pod tiene su propia dirección IP única. El límite anterior de 64 pods dejará de aplicarse después de una actualización de Windows Server a partir de la compilación del SO 17763.2746.
- No se admiten contenedores de Windows para pods de Amazon EKS en Fargate.
- No puede utilizar Nodos híbridos de Amazon EKS con Windows como sistema operativo del host.
- No puede recuperar registros del pod de `vpc-resource-controller`. Anteriormente podía hacerlo cuando se implementaba el controlador en el plano de datos.
- Hay un periodo de enfriamiento antes de que se asigne una dirección `IPv4` a un nuevo pod. Esto evita que el tráfico fluya hacia un pod anterior con la misma dirección `IPv4` debido a reglas caducadas de `kube-proxy`.
- El origen del controlador se administra en GitHub. Para registrar problemas con el controlador u ofrecer ayuda con estos, visite el proyecto en GitHub.
- Al especificar un ID de AMI personalizado para los grupos de nodos administrados de Windows, agregue `eks:kube-proxy-windows` al mapa de configuración del Autenticador de AWS IAM. Para obtener más información, consulte the section called “Límites y condiciones al especificar un ID de AMI”.
- Si conservar las direcciones IPv4 disponibles es fundamental para su subred, consulte la Guía de prácticas recomendadas de EKS: Administración de direcciones IP de redes de Windows para obtener orientación.

- Consideraciones para las entradas de acceso de EKS
  - Las entradas de acceso para su uso con los nodos de Windows necesitan el tipo de `EC2_WINDOWS`. Para obtener más información, consulte the section called “Creación de entradas de acceso”.

    Para crear una entrada de acceso para un nodo de Windows:

    ```
    aws eks create-access-entry --cluster-name my-cluster --principal-arn
     arn:aws:iam::111122223333:role/<role-name> --type EC2_Windows
    ```

Requisitos previos

- Un clúster existente.
- El clúster debe tener al menos un nodo de Linux o pod de Fargate (recomendamos al menos dos) para ejecutar CoreDNS. Si habilita la compatibilidad con Windows heredado, debe utilizar un nodo de Linux (no puede usar un pod de Fargate) para ejecutar CoreDNS.
- Un rol de IAM de clúster de Amazon EKS existente.

Habilitación de la compatibilidad con Windows

1. Si no tiene nodos de Amazon Linux en su clúster y utiliza grupos de seguridad para pods, avance al paso siguiente. De lo contrario, confirme que la política administrada `AmazonEKSVPCResourceController` esté adjunta al rol del clúster. Reemplace *`eksClusterRole`* por el nombre del rol del clúster.

   ```
   aws iam list-attached-role-policies --role-name eksClusterRole
   ```

   Un ejemplo de salida sería el siguiente.

   ```
   {
       "AttachedPolicies": [
           {
               "PolicyName": "AmazonEKSClusterPolicy",
               "PolicyArn": "arn:aws:iam::aws:policy/AmazonEKSClusterPolicy"
           },
           {
               "PolicyName": "AmazonEKSVPCResourceController",
   ```

```
            "PolicyArn": "arn:aws:iam::aws:policy/AmazonEKSVPCResourceController"
        }
    ]
}
```

Si la política está adjunta, como en la salida anterior, omita el siguiente paso.

2. Agregue la política de administrada AmazonEKSVPCResourceController al rol de IAM asociado a su clúster de Amazon EKS. Reemplace *eksClusterRole* por el nombre del rol del clúster.

```
aws iam attach-role-policy \
  --role-name eksClusterRole \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSVPCResourceController
```

3. Actualice el ConfigMap de la CNI de la VPC para habilitar el IPAM de Windows. Tenga en cuenta que si la CNI de la VPC está instalada en el clúster mediante un gráfico de Helm o como un complemento de Amazon EKS, es posible que no pueda modificar directamente el ConfigMap. Para obtener información sobre la configuración de complementos de Amazon EKS, consulte the section called “Campos que puede personalizar”.

   a. Cree un archivo llamado *vpc-resource-controller-configmap.yaml* con el siguiente contenido.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: amazon-vpc-cni
  namespace: kube-system
data:
  enable-windows-ipam: "true"
```

   b. Aplique el `ConfigMap` en su clúster.

```
kubectl apply -f vpc-resource-controller-configmap.yaml
```

4. Si su clúster tiene el modo de autenticación configurado para habilitar el configmap de `aws-auth`:

   - Verifique que el `ConfigMap` de `aws-auth` contenga una asignación para que el rol de instancia del nodo de Windows incluya el grupo de permisos RBAC de `eks:kube-proxy-windows`. Puede verificar ejecutando el siguiente comando:

```
kubectl get configmap aws-auth -n kube-system -o yaml
```

Un ejemplo de salida sería el siguiente.

```
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` debería aparecer en la lista de grupos. Si el grupo no está especificado, debe actualizar `ConfigMap` o crearlo para incluir el grupo requerido. Para obtener más información sobre `ConfigMap` de `aws-auth`, consulte the section called “Aplique el `ConfigMap` de `aws-auth` en su clúster”.

5. Si su clúster tiene el modo de autenticación configurado para deshabilitar el configmap de `aws-auth`, puede usar las entradas de acceso de EKS. Cree un nuevo rol de nodo para usarlo con las instancias de Windows, y EKS creará automáticamente una entrada de acceso de tipo `EC2_WINDOWS`.

Implementación de pods de Windows

Cuando implementa pods en el clúster, debe especificar el sistema operativo que utilizan si ejecuta una combinación de tipos de nodos.

Para pods de Linux, use el siguiente texto del selector de nodos en sus manifiestos.

```
nodeSelector:
        kubernetes.io/os: linux
        kubernetes.io/arch: amd64
```

Para pods de Windows, use el siguiente texto del selector de nodos en sus manifiestos.

```
nodeSelector:
        kubernetes.io/os: windows
        kubernetes.io/arch: amd64
```

Puede implementar una aplicación de muestra para ver los selectores de nodos en uso.

Soporte para una mayor densidad de pods en los nodos de Windows

En Amazon EKS, a cada pod se le asigna una dirección `IPv4` desde su VPC. Debido a esto, la cantidad de pods que puede implementar en un nodo está limitada por las direcciones IP disponibles, incluso si hay recursos suficientes para ejecutar más pods en el nodo. Dado que un nodo de Windows solo admite una interfaz de red elástica, de forma predeterminada, la cantidad máxima de direcciones IP disponibles en un nodo de Windows es igual a:

```
Number of private IPv4 addresses for each interface on the node - 1
```

Se utiliza una dirección IP como dirección IP principal de la interfaz de red, por lo que no se puede asignar a pods.

Puede habilitar una mayor densidad de pods en los nodos de Windows si habilita la delegación de prefijos de IP. Esta característica le permite asignar un prefijo `/28` `IPv4` a la interfaz de red principal, en lugar de asignar direcciones `IPv4` secundarias. La asignación de un prefijo IP aumenta el número máximo de direcciones `IPv4` disponibles en el nodo a:

```
(Number of private IPv4 addresses assigned to the interface attached to the node - 1) *
 16
```

Con esta cantidad significativamente mayor de direcciones IP disponibles, las direcciones IP disponibles no deberían limitar su capacidad para escalar la cantidad de pods en sus nodos. Para obtener más información, consulte the section called “Aumento de las direcciones IP”.

Deshabilitar la compatibilidad con Windows

1. Si el clúster contiene nodos de Amazon Linux y utiliza grupos de seguridad para pods, omita este paso.

   Eliminación de política administrada de IAM `AmazonVPCResourceController` de su rol de clúster. Sustituya *`eksClusterRole`* por el nombre del rol del clúster.

```
aws iam detach-role-policy \
    --role-name eksClusterRole \
    --policy-arn arn:aws:iam::aws:policy/AmazonEKSVPCResourceController
```

2. Deshabilite IPAM de Windows en el ConfigMap de `amazon-vpc-cni`.

```
kubectl patch configmap/amazon-vpc-cni \
                    -n kube-system \
                    --type merge \
                    -p '{"data":{"enable-windows-ipam":"false"}}'
```

Implementación de clústeres privados con acceso limitado a Internet

En este tema, se describe cómo implementar un clúster de Amazon EKS que se implementa en la nube de AWS sin acceso a Internet saliente. Si tiene un clúster local en AWS Outposts, consulte the section called “Nodos” en lugar de este tema.

Si no está familiarizado con las redes de Amazon EKS, consulte Desmitificación de las redes de clústeres para nodos de trabajo de Amazon EKS. Si su clúster no tiene acceso a Internet saliente, debe cumplir con los siguientes requisitos:

Requisitos de la arquitectura del clúster

- El clúster debe extraer imágenes de un registro de contenedores que esté en su VPC. Puede crear un Amazon Elastic Container Registry en su VPC y copiar las imágenes del contenedor para que sus nodos puedan extraerlas. Para obtener más información, consulte the section called “Copiar una imagen en un repositorio”.
- Su clúster debe tener habilitado el acceso privado a los puntos de conexión. Esto es necesario para que los nodos se registren en el punto de conexión del clúster. El acceso público del punto de conexión es opcional. Para obtener más información, consulte the section called “Acceso al punto de conexión del clúster”.

Requisitos del nodo

- Los nodos autoadministrados de Linux y Windows deben incluir los siguientes argumentos de arranque antes de lanzarlos. Estos argumentos omiten la introspección de Amazon EKS y no requieren acceso a la API de Amazon EKS desde dentro de la VPC.

  a. Determine el valor del punto de conexión del clúster con el siguiente comando. Reemplace *my-cluster* por el nombre de su clúster.

  ```
  aws eks describe-cluster --name my-cluster --query cluster.endpoint --output text
  ```

  Un ejemplo de salida sería el siguiente.

  ```
  https://EXAMPLE108C897D9B2F1B21D5EXAMPLE.sk1.region-code.eks.amazonaws.com
  ```

  b. Determine el valor de la autoridad de certificación del clúster con el siguiente comando. Reemplace *my-cluster* por el nombre de su clúster.

  ```
  aws eks describe-cluster --name my-cluster --query cluster.certificateAuthority --
  output text
  ```

  El resultado devuelto es una cadena larga.

  c. Reemplace los valores de `apiServerEndpoint` y `certificateAuthority` en el objeto NodeConfig por los valores devueltos en la salida de los comandos anteriores. Para obtener más información acerca de cómo especificar argumentos de arranque al lanzar nodos de Amazon Linux 2023 autoadministrados, consulte the section called “Amazon Linux” y the section called “Windows”.

  - En el caso de los nodos Linux:

  ```
  ---
  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=
    ...
```

Para obtener argumentos adicionales, consulte el script de arranque en GitHub.

- Para los usuarios de Windows:

> **Note**
>
> Si utiliza un CIDR de servicio personalizado, debe especificarlo con el parámetro `-ServiceCIDR`. De lo contrario, se producirá un error en la resolución de DNS del clúster para pods.

```
-APIServerEndpoint cluster-endpoint -Base64ClusterCA certificate-authority
```

Para obtener argumentos adicionales, consulte the section called “Parámetros de configuración del script de arranque”.

- El `aws-auth` `ConfigMap` del clúster debe crearse desde su VPC. Para obtener más información sobre cómo crear y añadir entradas al `ConfigMap` de `aws-auth`, ingrese `eksctl create iamidentitymapping --help` en su terminal. Si el `ConfigMap` no existe en su servidor, `eksctl` lo creará cuando utilice el comando para añadir una asignación de identidad.

Requisitos del pod

- Pod Identity: los pods configurados con Pod Identity de EKS adquieren las credenciales de la API de autenticación de EKS. Si no hay acceso a Internet de salida, debe crear y utilizar un punto de conexión de VPC para la API de autenticación de EKS: `com.amazonaws.region-code.eks-auth`. Para obtener más información sobre los puntos de conexión de VPC de EKS y de autenticación de EKS, consulte the section called “ AWS PrivateLink”.
- IRSA: los pods configurados con roles de IAM para cuentas de servicio adquieren credenciales de una llamada a la API de AWS Security Token Service (AWS STS). Si no hay acceso a Internet

saliente, debe crear y utilizar un punto de conexión de VPC de AWS STS en su VPC. La mayoría de los SDK de AWS v1 utilizan el punto de conexión global de AWS STS de forma predeterminada (`sts.amazonaws.com`), que no utiliza el punto de conexión de VPC de AWS STS. Para utilizar el punto de conexión de VPC de AWS STS, es posible que tenga que configurar el SDK de modo que utilice el punto de conexión de AWS STS regional (`sts.region-code.amazonaws.com`). Para obtener más información, consulte the section called “Puntos de conexión de STS”.

- Las subredes de VPC de su clúster deben tener un punto de conexión de interfaz de VPC para todas las subredes de VPC de los servicios de AWS a los que los pods necesiten acceder. Para obtener más información, consulte Acceso a un servicio de AWS a través de un punto de conexión de VPC de interfaz. Algunos servicios y puntos de conexión de uso común se enumeran en la siguiente tabla. Para obtener la lista completa de los puntos de conexión, consulte Servicios de AWS que se integran con AWS PrivateLink en la Guía de AWS PrivateLink.

  Recomendamos que habilite nombres de DNS privados para los puntos de conexión de la VPC, de modo que las cargas de trabajo aún puedan utilizar los puntos de conexión de servicio de AWS públicos sin problemas.

Servicio	Punto de conexión
Amazon EC2	com.amazonaws.*region-code* .ec2
Amazon Elastic Container Registry (para extraer imágenes de contenedores)	com.amazonaws.*region-code* .ecr.api, com.amazonaws.*region-code* .ecr.dkr y com.amazonaws.*region-code* .s3
Equilibradores de carga de aplicación y Equilibradores de carga de red de Amazon	com.amazonaws.*region-code* .elasticloadbalancing
(Opcional) AWS X-Ray (necesario para rastreos enviados a AWS X-Ray)	com.amazonaws.*region-code* .xray
(Opcional) Amazon SSM (necesario para el Agente de SSM para las tareas de administración de nodos. Alternativa a SSH)	com.amazonaws.*region-code* .ssm
Registros de Amazon CloudWatch (necesario para los registros de nodos y pods enviados a Registros de Amazon CloudWatch)	com.amazonaws.*region-code* .logs

Servicio	Punto de conexión
AWS Security Token Service (requerido al usar roles de IAM para cuentas de servicio)	com.amazonaws.*region-code* .sts
Amazon EKS Auth (requerido al utilizar asociaciones de Pod Identity)	com.amazonaws.*region-code* .eks-auth
Amazon EKS	com.amazonaws.*region-code* .eks

- Todos los nodos autoadministrados deben implementarse en subredes que tengan los puntos de conexión de la interfaz de VPC que necesita. Si crea un grupo de nodos administrados, el grupo de seguridad del punto de conexión de la interfaz de VPC debe permitir el CIDR para las subredes. También puede agregar el grupo de seguridad del nodo creado al grupo de seguridad del punto de conexión de la interfaz de VPC.
- Almacenamiento de EFS: si sus pods utilizan volúmenes de Amazon EFS, antes de implementar el almacenamiento de un sistema de archivos con Amazon EFS, se debe cambiar el archivo kustomization.yaml del controlador para establecer las imágenes de contenedor de manera que utilicen la misma región de AWS que el clúster de Amazon EKS.
- Route 53 no es compatible con AWS PrivateLink. No puede administrar los registros de DNS de Route 53 desde un clúster privado de Amazon EKS. Esto afecta a external-dns de Kubernetes.
- Si usa la AMI optimizada para EKS, debe habilitar el punto de conexión de `ec2` en la tabla anterior. También puede configurar el nombre de DNS del nodo de forma manual. La AMI optimizada utiliza las API de EC2 para establecer el nombre de DNS del nodo automáticamente.
- Puede utilizar el controlador de equilibrador de carga de AWS para implementar los Equilibradores de carga de aplicación (ALB) y los equilibradores de carga de red de AWS en su clúster privado. Al implementarlo, debe usar los indicadores de línea de comandos para establecer `enable-shield`, `enable-waf` y `enable-wafv2` como falsos. No se admite la detección de certificados con nombres de host de los objetos de entrada. Esto se debe a que el controlador necesita llegar a AWS Certificate Manager, el cual no tiene un punto de conexión de la interfaz de VPC.

  El controlador es compatible con equilibradores de carga de red con destinos IP, que son necesarios para su uso con Fargate. Para obtener más información, consulte the section called “Equilibrio de carga de aplicaciones” y the section called “Crear un equilibrador de carga de red”.
- Se admite el Escalador automático de clústeres. Al implementar pods del Escalador automático de clústeres, asegúrese de que la línea de comandos incluya `--aws-use-static-instance-list=true`. Para obtener más información, consulte Use Static Instance List en GitHub. La VPC

del nodo de trabajo también debe incluir el punto de conexión de VPC de AWS STS y el punto de conexión de VPC de escalado automático.

- Algunos productos de software de contenedores utilizan llamadas API que acceden al servicio de medición de AWS Marketplace para supervisar el uso. Los clústeres privados no permiten estas llamadas, por lo que estos tipos de contenedores no se pueden utilizar en clústeres privados.

Escale la computación en clústeres con Karpenter y el Escalador automático de clústeres

El escalado automático es una función que escala y reduce horizontalmente los recursos de manera automática para satisfacer las demandas cambiantes. Esta es una función importante de Kubernetes que, de otro modo, requeriría muchos recursos humanos para funcionar manualmente.

Modo automático de EKS

El modo automático de Amazon EKS escala automáticamente los recursos de computación del clúster. Si un pod no cabe en los nodos existentes, el modo automático de EKS crea uno nuevo. El modo automático de EKS también consolida cargas de trabajo y elimina nodos. El modo automático de EKS se basa en Karpenter.

Para obtener más información, consulte:

- Modo automático de EKS
- the section called “Creación de un grupo de nodos”
- the section called “Implemente una carga de trabajo de expansión”

Soluciones adicionales

Amazon EKS admite dos productos adicionales de escalado automático:

Karpenter

Karpenter es un escalador automático de clústeres de Kubernetes flexible y de alto rendimiento que ayuda a mejorar la disponibilidad de las aplicaciones y la eficiencia del clúster. Karpenter lanza recursos de computación del tamaño correcto (por ejemplo, instancias de Amazon EC2) en respuesta a los cambios en la carga de la aplicación en menos de un minuto. Mediante la

integración de Kubernetes con AWS, Karpenter puede aprovisionar recursos informáticos justo a tiempo que cumplen con precisión los requisitos de su carga de trabajo. Karpenter aprovisiona automáticamente nuevos recursos informáticos en función de los requisitos específicos de las cargas de trabajo de un clúster. Estos incluyen requisitos de computación, almacenamiento, aceleración y programación. Amazon EKS admite clústeres que utilizan Karpenter, aunque Karpenter funciona con cualquier clúster de Kubernetes conforme. Para obtener más información, consulte la documentación de Karpenter.

> **Important**
>
> Karpenter es un software de código abierto que los clientes de AWS son responsables de instalar, configurar y administrar en sus clústeres de Kubernetes. AWS proporciona soporte técnico cuando Karpenter se ejecuta sin modificaciones utilizando una versión compatible en los clústeres de Amazon EKS. Es fundamental que los clientes aseguren la disponibilidad y seguridad del controlador Karpenter y mantengan procedimientos de prueba adecuados al actualizar tanto el controlador como el clúster de Kubernetes en el que se ejecuta, al igual que con cualquier otro software administrado por el cliente. Karpenter no tiene ningún Acuerdo de nivel de servicio (SLA) de AWS y los clientes son responsables de garantizar que las instancias EC2 lanzadas por Karpenter cumplan con sus requisitos empresariales.

Escalador automático de clústeres

El Cluster Autoscaler de Kubernetes ajusta de forma automática el número de nodos del clúster cuando los pods fallan o se reprograman en otros nodos. El Escalador automático de clústeres utiliza grupos de escalado automático. Para obtener más información, consulte Escalador automático de clústeres en AWS.

Información sobre el cambio de zona del Controlador de recuperación de aplicaciones de Amazon (ARC) en Amazon EKS

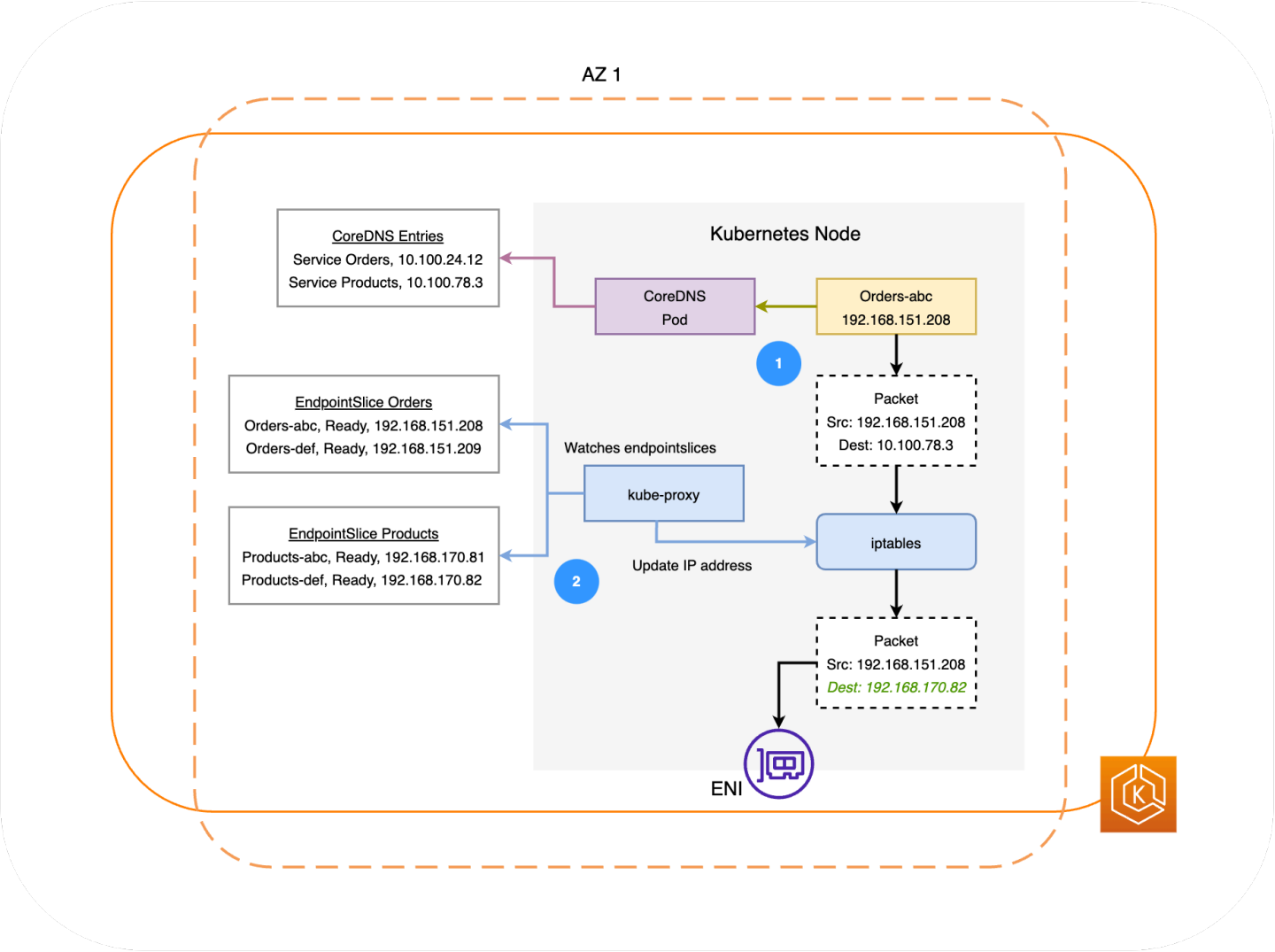
Kubernetes tiene características nativas que le permiten que sus aplicaciones sean más resistentes a eventos como el deterioro del estado o el deterioro de una zona de disponibilidad (AZ). Cuando ejecuta sus cargas de trabajo en un clúster de Amazon EKS, puede mejorar aún más la tolerancia a errores del entorno y la recuperación de aplicaciones mediante el cambio de zona del Controlador de recuperación de aplicaciones de Amazon (ARC) o el cambio de zona automático. El cambio de

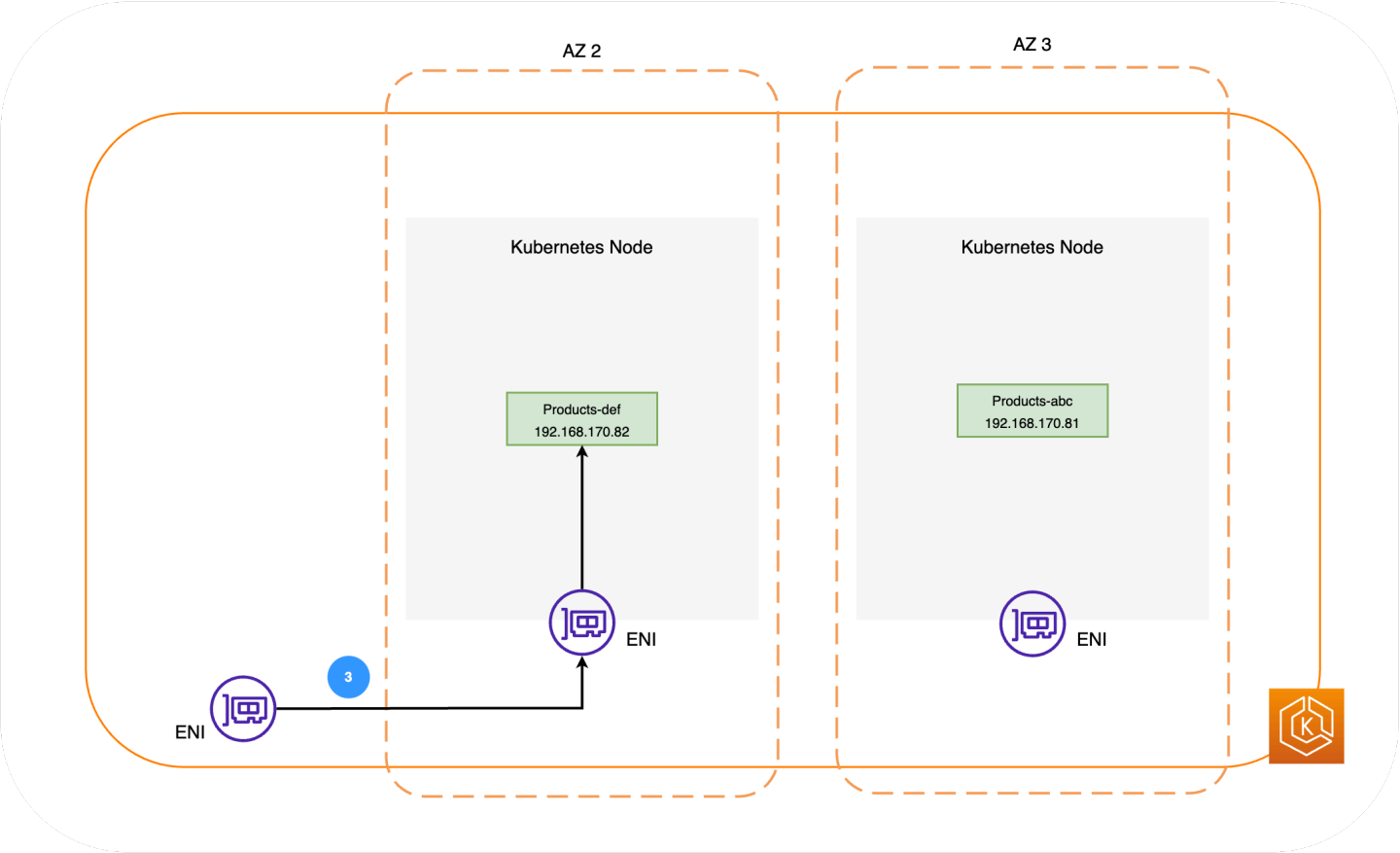
zona de ARC está diseñado como una medida temporal que le permite alejar el tráfico de un recurso de una AZ deteriorada hasta que el cambio de zona caduque o se cancele. Si es necesario, puede ampliar el cambio de zona.

Puede iniciar un cambio de zona para un clúster de EKS o permitir que AWS cambie el tráfico mediante la activación del cambio de zona automático. Este cambio actualiza el flujo del tráfico de red de este a oeste en su clúster para tener en cuenta únicamente los puntos de conexión de red de los pods que se ejecutan en nodos de trabajo en AZ en buen estado. Además, cualquier ALB o NLB que administre la entrada del tráfico de las aplicaciones de su clúster de EKS enrutará de forma automática el tráfico a los destinos de las AZ en buen estado. Para aquellos clientes que buscan los objetivos de disponibilidad más altos, en caso de que una AZ se estropee, puede ser importante poder desviar todo el tráfico de la AZ afectada hasta que se recupere. Para esto, también puede activar un ALB o un NLB con el cambio de zona de ARC.

Descripción del flujo de tráfico de red de este a oeste entre pods

El siguiente diagrama ilustra dos ejemplos de cargas de trabajo, Pedidos y Productos. El propósito de este ejemplo es mostrar cómo se comunican las cargas de trabajo y los pods en diferentes AZ.


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. Para que Pedidos se comunique con Productos, debe resolver antes el nombre de DNS del servicio de destino. Pedidos se comunica con CoreDNS para obtener la dirección IP virtual (IP del clúster) de ese servicio. Después de que Pedidos resuelva el nombre del servicio de Productos, envía el tráfico a esa dirección IP de destino.
2. El kube-proxy se ejecuta en todos los nodos del clúster y supervisa de manera continua los EndpointSlices en busca de servicios. Cuando se crea un servicio, el controlador EndpointSlice crea y administra un EndpointSlice en segundo plano. Cada EndpointSlice tiene una lista o tabla de puntos de conexión que contiene un subconjunto de direcciones de pods junto con los nodos en los que se ejecutan. El kube-proxy establece reglas de enrutamiento para cada uno de estos puntos de conexión de los pods mediante el uso de `iptables` en los nodos. El kube-proxy también es responsable de una forma básica de equilibrio de carga, ya que redirige el tráfico destinado a la dirección IP del clúster de un servicio para que, en su lugar, se envía directamente a la dirección IP del pod. Para hacerlo, el kube-proxy reescribe la dirección IP de destino en la conexión saliente.
3. A continuación, los paquetes de red se envían al pod Productos de la AZ 2 mediante los ENI de los nodos respectivos, como se muestra en el diagrama anterior.

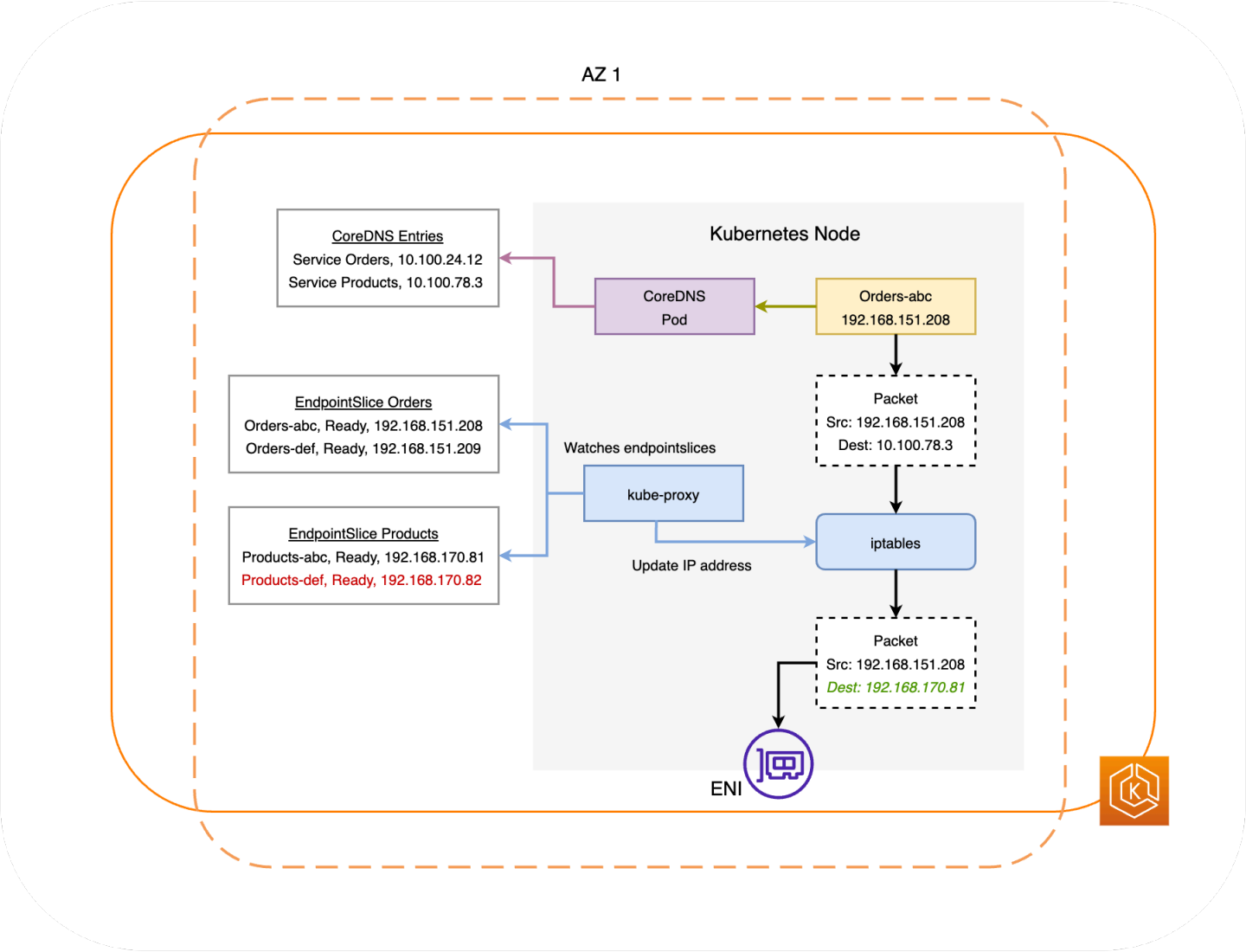
Descripción del cambio de zona de ARC en Amazon EKS

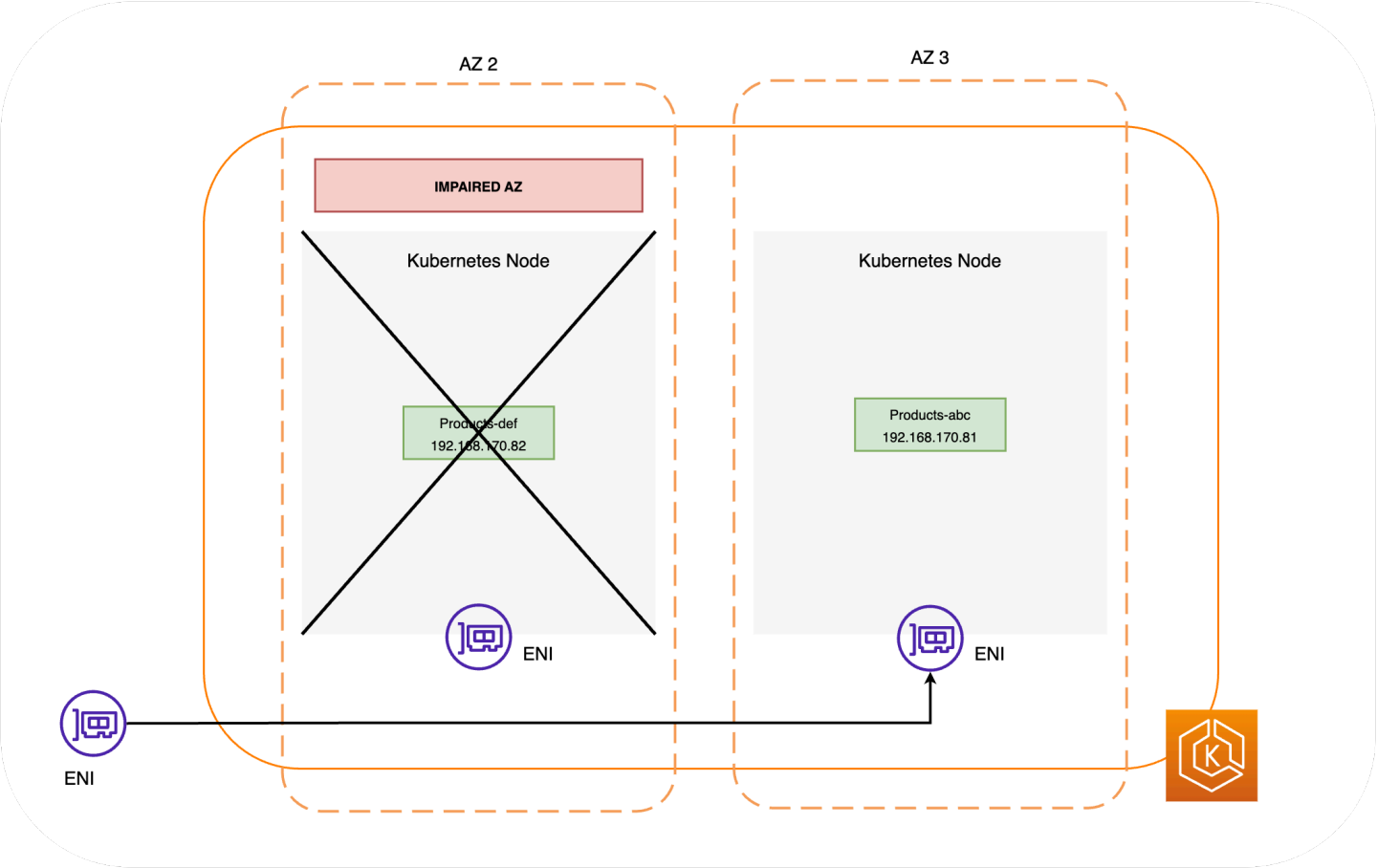
Si hay una alteración de la AZ en su entorno, puede iniciar un cambio de zona para su entorno de clústeres de EKS. Como alternativa, puede permitir que AWS administre el cambio de tráfico con el cambio de zona automático. Con el cambio de zona automático, AWS supervisa el estado general de la AZ y responde a una posible alteración de esta mediante el cambio automático del tráfico de la AZ dañada de su entorno de clústeres.

Cuando se haya activado el cambio de zona del clúster de Amazon EKS con ARC, puede iniciar un cambio de zona o activar un cambio de zona automático mediante la consola de ARC, la AWS CLI o las API de cambio de zona automático y cambio de zona. Durante un cambio de zona de EKS, ocurrirá lo siguiente de forma automática:

- Se acordonan todos los nodos de la AZ afectada. De este modo, se evita que el programador de Kubernetes programe nuevos pods en los nodos de la AZ en mal estado.
- Si utiliza grupos de nodos administrados, se suspende el reequilibrio de la zona de disponibilidad y se actualiza el grupo de escalado automático para garantizar que los nuevos nodos del plano de datos de Amazon EKS solo se lancen en las zonas de disponibilidad en buen estado.
- Los nodos de la AZ en mal estado no se terminan y los pods no se expulsan de los nodos. De este modo, se garantizar que, cuando un cambio de zona caduque o se cancele, el tráfico pueda devolverse con seguridad a la AZ para una capacidad plena.
- El controlador EndpointSlice busca todos los puntos de conexión del pod en la AZ defectuosa y los elimina de los EndpointSlices pertinentes. Esto garantiza que solo los puntos de conexión de pods situados en zonas de disponibilidad en buen estado estén destinados a recibir tráfico de red. Cuando un cambio de zona caduca o se cancela, el controlador EndpointSlice actualiza los EndpointSlices para incluir los puntos de conexión de la AZ restaurada.

Los diagramas siguientes proporcionan información general de alto nivel sobre cómo el cambio de zona de EKS garantiza que solo se usen como objetivo los puntos de conexión de pod en buen estado en el entorno de clústeres.


AZ 1
Kubernetes Node
CoreDNS Entries
Service Orders, 10.100.24.12
Service Products, 10.100.78.3
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
Update IP address
iptables
Packet
Src: 192.168.151.208
Dest: 192.168.170.81
ENI



Requisitos del cambio de zona de EKS

Para que el cambio de zona funcione de manera correcta con EKS, debe configurar previamente su entorno de clústeres para que sea resistente a un deterioro de la AZ. A continuación se muestra una lista de opciones de configuración que ayudan a garantizar la resiliencia.

- Aprovisione los nodos de trabajo de su clúster en varias zonas de disponibilidad.
- Aprovisione suficiente capacidad de computación para acomodar la eliminación de una sola AZ.
- Escale previamente sus pods, incluido CoreDNS, en todas las AZ.
- Distribuya varias réplicas de pods en todas las AZ para asegurarse de que dispondrá de suficiente capacidad si deja de utilizar una sola AZ.
- Ubique los pods interdependientes o relacionados en la misma AZ.
- Inicie un cambio de zona de una AZ de forma manual para probar si el entorno de clústeres funciona según lo esperado sin una AZ. Como alternativa, puede activar el cambio de zona automático y depender de las ejecuciones de práctica del cambio automático. Las pruebas con

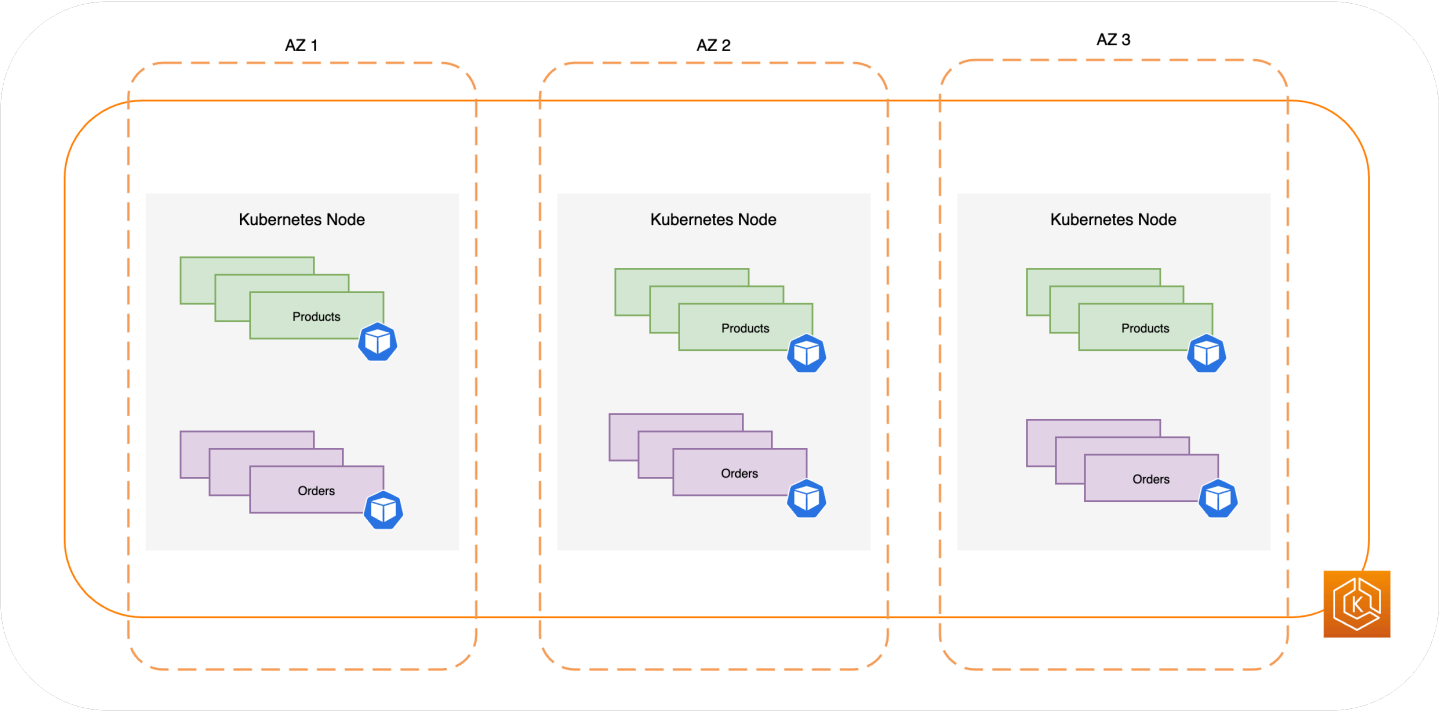
cambios de zona manuales o de práctica no son obligatorias para que el cambio de zona funcione en EKS, pero se recomiendan encarecidamente.

Aprovisionamiento de los nodos de trabajo de EKS en varias zonas de disponibilidad

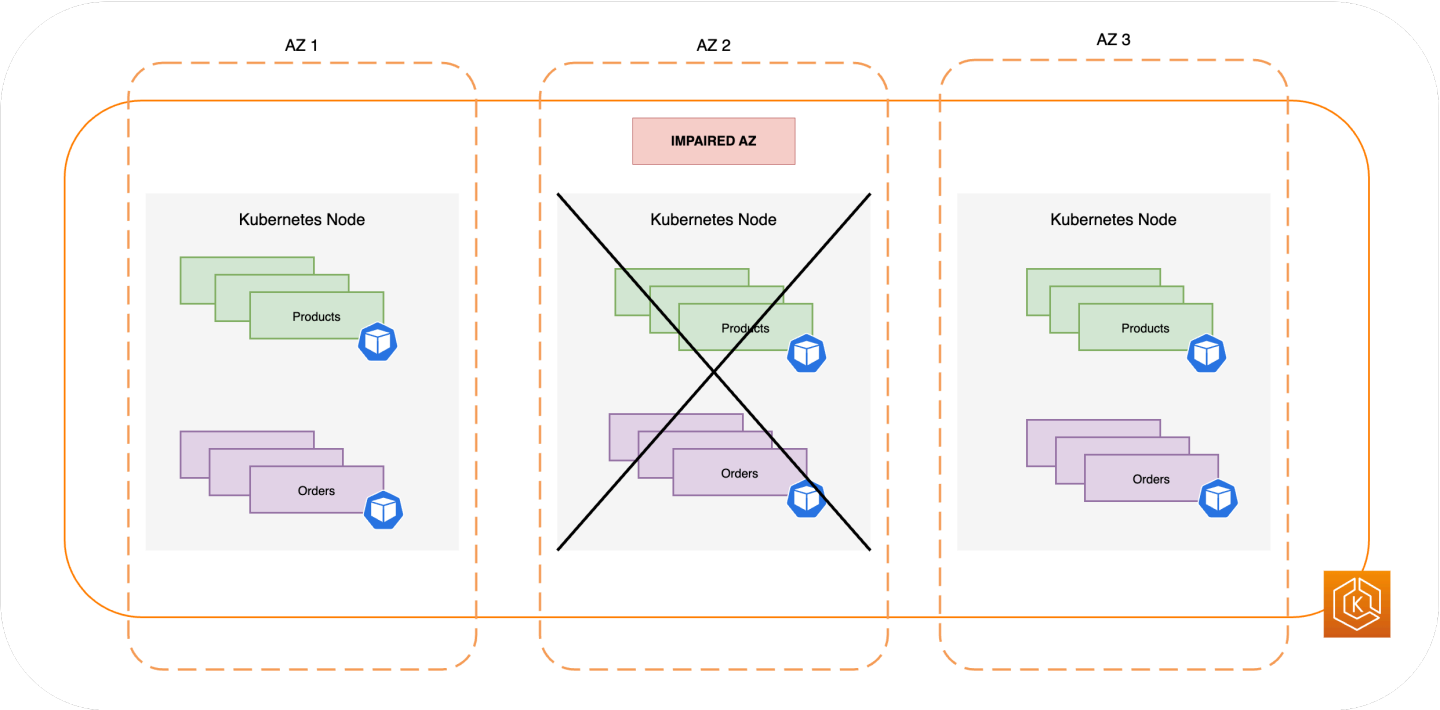
Las regiones de AWS tienen varias ubicaciones independientes con centros de datos físicos, conocidas como zonas de disponibilidad (AZ). Las AZ están diseñadas para estar aisladas físicamente unas de otras a fin de evitar un impacto simultáneo que podría afectar a toda una región. Al aprovisionar un clúster de EKS, le recomendamos que implemente los nodos de trabajo en varias AZ de una región. Esto ayudará a que su entorno ende clústeres sea más resistente a los daños de una sola AZ y le permitirá mantener una alta disponibilidad para las aplicaciones que se ejecuten en las demás AZ. Cuando inicie un cambio de zona para alejarse de la AZ afectada, la red integrada en el clúster de su entorno de Amazon EKS se actualizará automáticamente para utilizar solo las AZ en buen estado a fin de ayudar a mantener una alta disponibilidad para el clúster.

Asegurarse de disponer de una configuración multi-AZ para su entorno de Amazon EKS mejora la fiabilidad general de su sistema. Sin embargo, los entornos multi-AZ influyen en cómo se transfieren y procesan los datos de las aplicaciones, lo que, a su vez, repercute en las tarifas de red de su entorno. Concretamente, el tráfico de salida frecuente entre zonas (tráfico distribuido entre las AZ) puede tener un impacto importante en los costos relacionados con la red. Puede aplicar diferentes estrategias para controlar la cantidad de tráfico entre zonas en los pods de su clúster de Amazon EKS y reducir los costos asociados. Para obtener más información sobre cómo optimizar los costos de red cuando se utilizan entornos de Amazon EKS de alta disponibilidad, consulte estas prácticas recomendadas.

En el siguiente diagrama, se ilustra un entorno de EKS de alta disponibilidad con tres AZ en buen estado.



En el siguiente diagrama, se ilustra cómo un entorno de Amazon EKS con tres AZ es resistente a una AZ dañada y mantiene una alta disponibilidad gracias a dos AZ restantes en buen estado.



Aprovisionamiento de suficiente capacidad de computación para soportar la eliminación de una sola zona de disponibilidad

Para optimizar el uso de los recursos y los costos de su infraestructura de computación en el plano de datos de Amazon EKS, una práctica recomendada consiste e alinear la capacidad de computación con los requisitos de la carga de trabajo. Sin embargo, si todos sus nodos de trabajo están a plena capacidad, tiene que agregar nuevos nodos de trabajo al plano de datos de EKS antes de poder programar nuevos pods. Cuando ejecuta cargas de trabajo críticas, suele ser una práctica recomendada ejecutar una capacidad redundante en línea para gestionar distintos escenarios como el aumento repentino de la carga o los problemas de estado de los nodos. Si tiene previsto utilizar un cambio de zona, tiene previsto eliminar toda una zona de disponibilidad de capacidad si hubiera daños. Esto significa que debe ajustar la capacidad de computación redundante a fin de que sea suficiente para gestionar la carga incluso con una de las AZ desconectada.

Al escalar los recursos de computación, el proceso de agregar nuevos nodos al plano de datos de EKS lleva algún tiempo. Esto puede repercutir en el rendimiento y la disponibilidad en tiempo real de las aplicaciones, en especial si hay daños en una zona. Su entorno de Amazon EKS debe poder absorber la carga que supone perder una AZ sin que la experiencia de sus usuarios finales o clientes se deteriore. Esto significa minimizar o eliminar cualquier intervalo entre el momento en que se necesita un nuevo pod y el momento real en que se programa en un nodo de trabajo.

Además, si se producen daños en una zona, debe intentar reducir el riesgo de que se produzca una posible limitación de la capacidad de computación que impida agregar nuevos nodos necesarios al plano de datos de Amazon EKS en las AZ en buen estado.

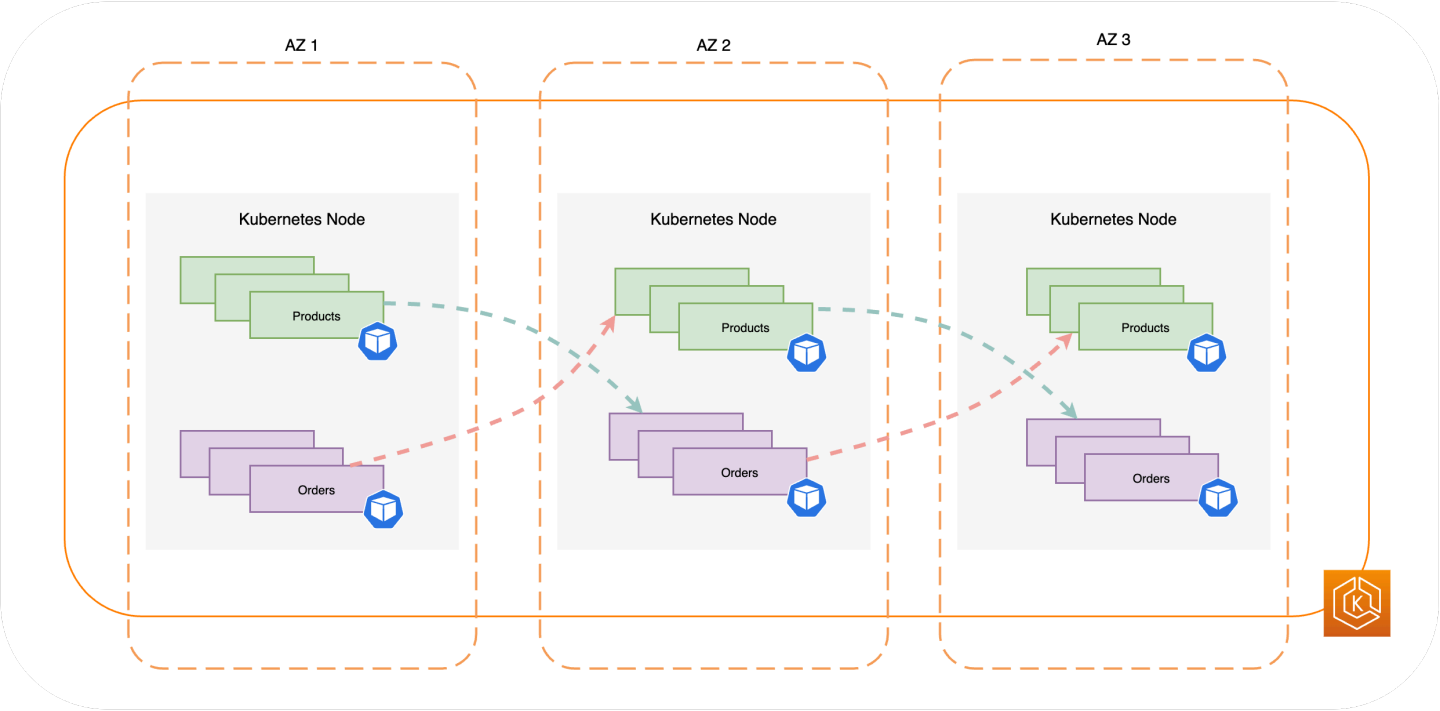
Para lograr reducir el riesgo de estos posibles impactos negativos, le recomendamos aprovisionar en exceso la capacidad de computación en algunos de los nodos de trabajo de cada AZ. De este modo, el programador de Kubernetes dispone de una capacidad preexistente para colocar nuevos pods, lo que es de especial importancia si se pierde una de las AZ del entorno.

Ejecución y distribución de varias réplicas de pods entre zonas de disponibilidad

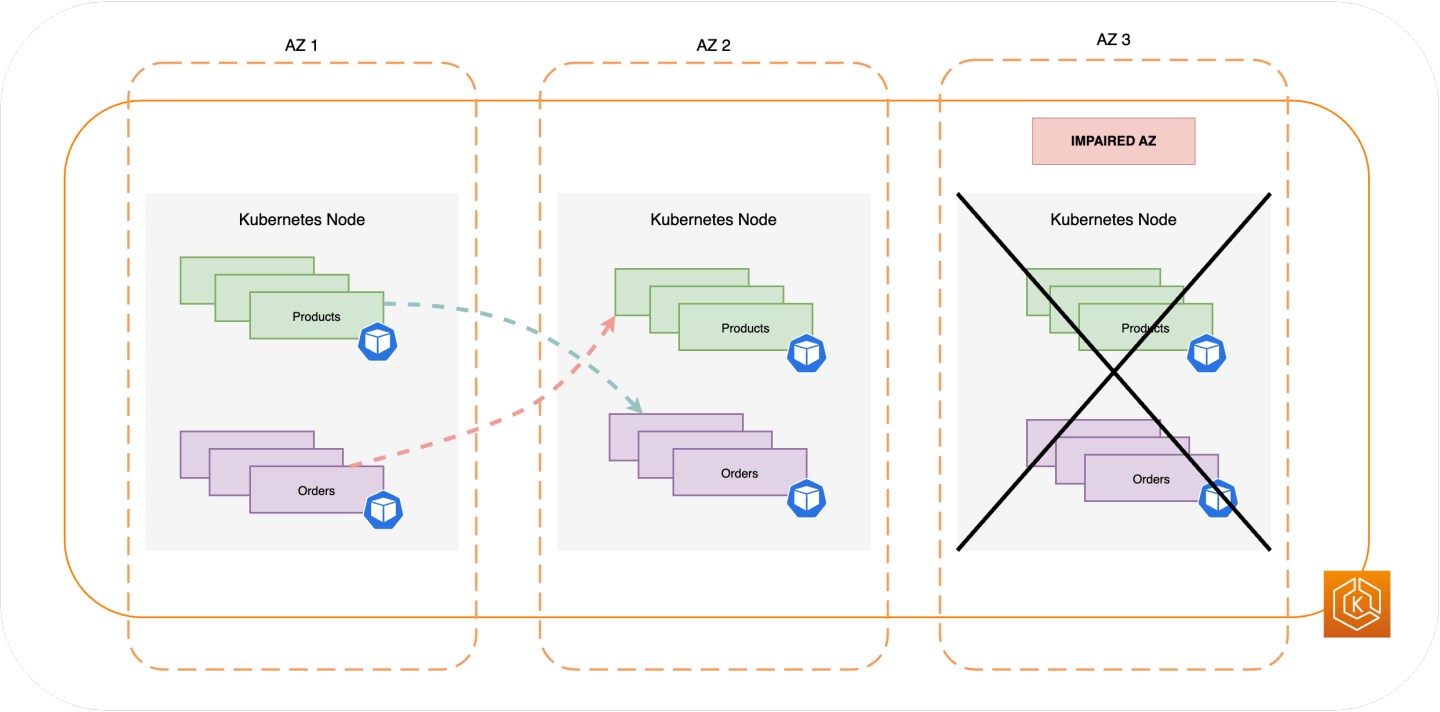
Kubernetes le permite escalar previamente sus cargas de trabajo mediante la ejecución de varias instancias (réplicas de pods) de una sola aplicación. Al ejecutar varias réplicas de pods para una aplicación, se eliminan puntos de error y se aumenta su rendimiento general, ya que se reduce la carga de recursos en una sola réplica. Sin embargo, para que sus aplicaciones tengan una alta disponibilidad y una mejor tolerancia a errores, le recomendamos que ejecute y distribuya varias réplicas de su aplicación en distintos dominios de errores, también denominados dominios de

topología. Los dominios de errores de este escenario son las zonas de disponibilidad. Al utilizar las restricciones de distribución topológica, puede configurar sus aplicaciones para que tengan una estabilidad estática preexistente. De este modo, cuando se produzcan daños en una zona de disponibilidad, su entorno dispondrá de suficientes réplicas en AZ en buen estado para gestionar inmediatamente cualquier pico o aumento del tráfico.

En el siguiente diagrama, se ilustra un entorno de Amazon EKS que tiene un flujo de tráfico de este a oeste cuando todas las AZ están en buen estado.



En el siguiente diagrama, se ilustra un entorno de Amazon EKS que tiene un flujo de tráfico de este a oeste en el que se ha producido un error en una sola zona de disponibilidad y ha iniciado un cambio de zona.



El siguiente fragmento de código es un ejemplo de cómo configurar la carga de trabajo con varias réplicas en Kubernetes.

```
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
```

Lo que es más importante, debe ejecutar varias réplicas del software de servidor de DNS (CoreDNS/kube-dns) y aplicar restricciones de distribución topológica similares si no están configuradas de forma predeterminada. Esto ayuda a garantizar que, si hay daños en una sola AZ, disponga de suficientes pods de DNS en AZ en buen estado para poder continuar con la gestión de las solicitudes de detección de servicios de otros pods que se comunican entre sí en el clúster. El complemento de EKS de CoreDNS tiene una configuración predeterminada para los pods de CoreDNS que garantiza que, si hay nodos en varias AZ disponibles, se distribuyan en las zonas de disponibilidad del clúster. Si lo desea, también puede reemplazar esta configuración predeterminada por su propia configuración personalizada.

Al instalar CoreDNS con Helm, puede actualizar el `replicaCount` en el archivo values.yaml para asegurarse de que disponga de suficientes réplicas en cada AZ. Además, para asegurarse de que estas réplicas estén distribuidas en las distintas AZ del entorno de clústeres, asegúrese de actualizar la propiedad `topologySpreadConstraints` en el mismo archivo `values.yaml`. En el siguiente fragmento de código, se ilustra cómo puede configurar CoreDNS para hacerlo.

CoredNS Helm values.yaml

```
replicaCount: 6
topologySpreadConstraints:
  - maxSkew: 1
    topologyKey: topology.kubernetes.io/zone
    whenUnsatisfiable: ScheduleAnyway
    labelSelector:
      matchLabels:
        k8s-app: kube-dns
```

Si hay daños en una AZ, puede absorber el aumento de carga de los pods de CoreDNS mediante un sistema de escalado automático para CoreDNS. La cantidad de instancias de DNS que necesite dependerá de la cantidad de cargas de trabajo que se ejecuten en el clúster. CoreDNS está vinculado a la CPU, lo que le permite escalar en función de la CPU mediante el Escalador automático de pods horizontales (HPA). A continuación, se muestra un ejemplo que puede modificar según sus necesidades.

```
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
```

Como alternativa, Amazon EKS puede administrar el escalado automático de la implementación de CoreDNS en la versión del complemento de EKS de CoreDNS. Este escalador automático de CoreDNS monitorea continuamente el estado del clúster, incluida la cantidad de nodos y núcleos de CPU. En función de esa información, el controlador ajusta dinámicamente el número de réplicas de la implementación de CoreDNS en un clúster de Amazon EKS.

Para activar la configuración de escalado automático en el complemento de EKS de CoreDNS, utilice la configuración siguiente:

```
{
  "autoScaling": {
    "enabled": true
  }
}
```

También puede usar el NodeLocal DNS o el escalador automático proporcional del clúster para escalar CoredNS. Para obtener más información, consulte Escalado horizontal de CoreDNS.

Colocación de pods interdependientes en la misma zona de disponibilidad

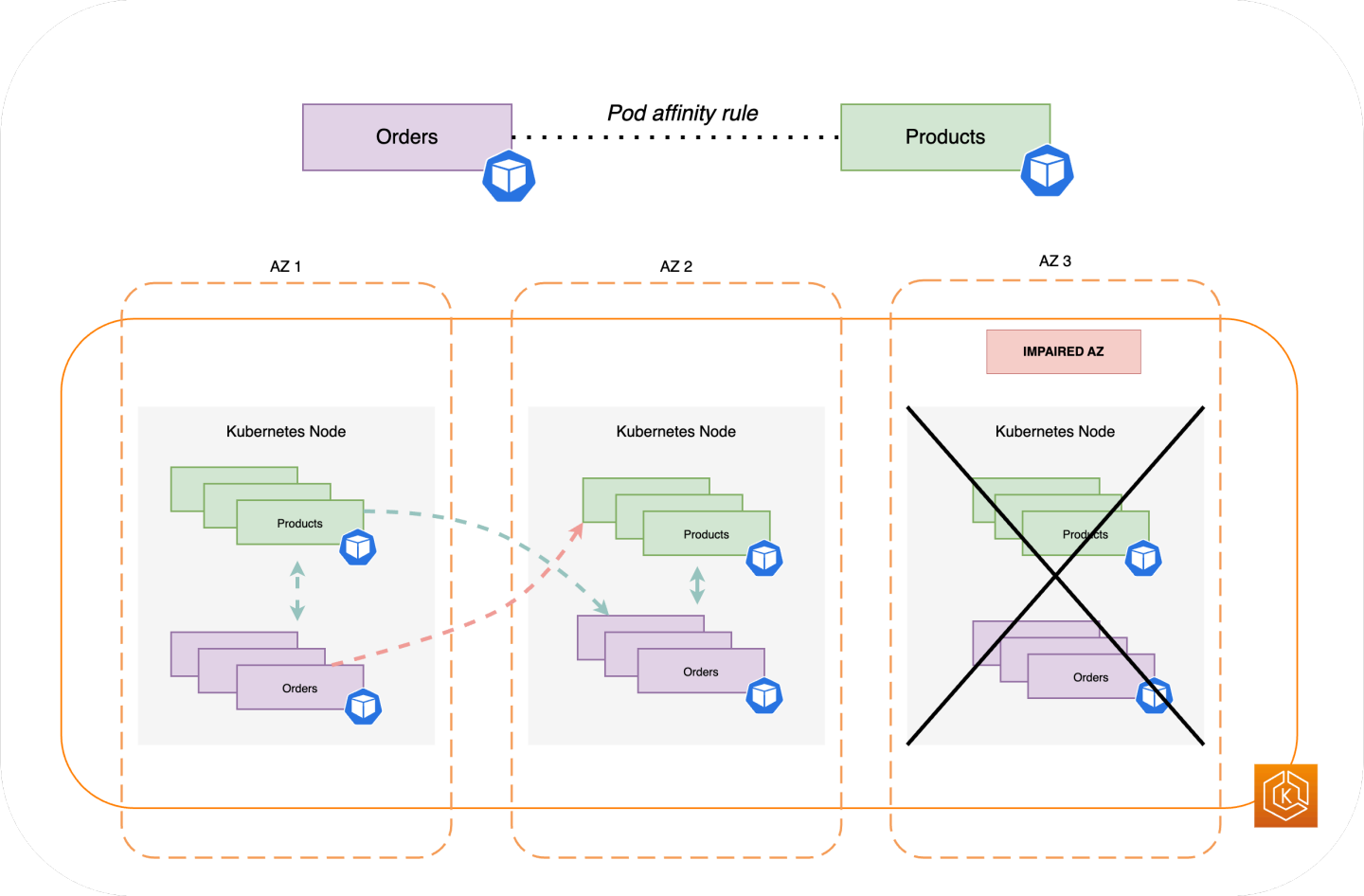
Por lo general, las aplicaciones tienen cargas de trabajo distintas que deben comunicarse entre sí para completar correctamente un proceso integral. Si estas distintas aplicaciones están distribuidas en diferentes AZ y no están ubicadas en la misma AZ, el proceso integral puede verse afectado si hay daños en una sola AZ. Por ejemplo, si la Aplicación A tiene varias réplicas en la AZ 1 y la AZ 2, pero la Aplicación B tiene todas sus réplicas en la AZ 3, la pérdida de la AZ 3 afectará al proceso integral entre las dos cargas de trabajo, Aplicación A y Aplicación B. Si combina las restricciones de distribución topológica con la afinidad de pods, puede mejorar la resiliencia de la aplicación mediante la distribución de pods entre todas las AZ. Además, se configura una relación entre determinados pods para garantizar su coubicación.

Con las reglas de afinidad de los pods, puede definir las relaciones entre las cargas de trabajo para influir en el comportamiento del programador de Kubernetes, de modo que coloque los pods en el mismo nodo de trabajo o en la misma zona de disponibilidad. También puede configurar qué tan estrictas deben ser las restricciones de programación.

```
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"
```

En el siguiente diagrama, se muestran varios pods que se han ubicado en el mismo nodo con reglas de afinidad de pods.



Comprobación de si el entorno de clústeres puede soportar la pérdida de una AZ

Después de cumplir los requisitos descritos en las secciones anteriores, el siguiente paso es comprobar que disponga de la capacidad de computación y cargas de trabajo suficientes para afrontar la pérdida de una AZ. Para ello, puede iniciar un cambio de zona en EKS de forma manual. Como alternativa, puede activar el cambio de zona automático y configurar ejecuciones de práctica, que también comprueban que las aplicaciones funcionen según lo esperado con una AZ menos en el entorno de clústeres.

Preguntas frecuentes

Por qué debo utilizar esta característica?

Al utilizar el cambio de zona o el cambio de zona automático de ARC en su clúster de Amazon EKS es más fácil mantener la disponibilidad de las aplicaciones de Kubernetes al automatizar el proceso de recuperación rápida que consiste en desviar el tráfico de red de una zona de disponibilidad

defectuosa dentro del clúster. Con ARC, puede evitar pasos largos y complicados que pueden conllevar un periodo de recuperación prolongado cuando se producen daños en las AZ.

¿Cómo funciona esta característica con otros servicios de AWS?

EKS se integra con ARC, que proporciona la interfaz principal en la que puede llevar a cabo las operaciones de recuperación en AWS. Para garantizar que el tráfico interno del clúster se aleje de manera apropiada de una AZ dañada, EKS hace modificaciones en la lista de puntos de conexión de red para los pods que se ejecutan en el plano de datos de Kubernetes. Si utiliza Elastic Load Balancing para dirigir el tráfico externo al clúster, puede registrar los equilibradores de carga con ARC e iniciar un cambio de zona en ellos para evitar que el tráfico fluya hacia la AZ dañada. El cambio de zona también funciona con grupos de Amazon EC2 Auto Scaling creados por los grupos de nodos administrados por EKS. Para evitar que una AZ dañada se utilice para lanzar nuevos nodos o pods de Kubernetes, EKS elimina la AZ dañada de los grupos de escalado automático.

En qué se diferencia esta característica de las protecciones predeterminadas de Kubernetes?

Esta característica funciona junto con varias protecciones integradas de Kubernetes que ayudan a la resiliencia de las aplicaciones de los clientes. Puede configurar sondeos de disponibilidad y actividad de un pod que decidan cuándo un pod debe recibir tráfico. Cuando se produce un error en estos sondeos, Kubernetes elimina estos pods como objetivos de un servicio y deja de enviarse el tráfico al pod. Si bien esto es útil, no es sencillo para los clientes configurar estas comprobaciones de estado de forma que se garantice que se produzca un error cuando una AZ se dañe. La característica de cambio de zona de ARC proporciona una red de seguridad adicional que lo ayuda a aislar por completo una AZ dañada cuando las protecciones nativas de Kubernetes no fueron suficientes. El cambio de zona también le proporciona una forma sencilla de evaluar la preparación operativa y la resiliencia de su arquitectura.

¿AWS puede iniciar un cambio de zona en mi nombre?

Sí, si desea utilizar el cambio de zona de ARC de forma totalmente automática, puede activar el cambio de zona automático de ARC. Con el cambio de zona automático, puede depender de AWS para supervisar el estado de las AZ del clúster de EKS e iniciar un cambio de zona cuando se detecten daños en la AZ de manera automática.

Qué ocurre si utilizo esta característica y mis nodos de trabajo y cargas de trabajo no están escaladas previamente?

Si no ha escalado previamente y depende del aprovisionamiento de nodos o pods adicionales durante un cambio de zona, corre el riesgo de que la recuperación se retrase. El proceso de agregar

nuevos nodos al plano de datos de EKS lleva algún tiempo, lo que puede afectar al rendimiento y la disponibilidad en tiempo real de las aplicaciones, especialmente si hay daños en una zona. Además, si se producen daños en una zona, es posible que se encuentre con una limitación de la capacidad de computación que impida agregar nuevos nodos necesarios a las AZ en buen estado.

Si sus cargas de trabajo no se han escalado previamente y no están distribuidas entre todas las AZ del clúster, los daños de una zona podrían afectar a la disponibilidad de una aplicación que solo se ejecuta en los nodos de trabajo de una AZ afectada. Para reducir el riesgo de una interrupción total de la disponibilidad de la aplicación, Amazon EKS dispone de un sistema de seguridad para que el tráfico se envíe a los puntos de conexión de los pods situados en una zona afectada, si esa carga de trabajo tiene todos sus puntos de conexión en una AZ en mal estado. Sin embargo, le recomendamos encarecidamente que escale previamente sus aplicaciones y las distribuya en todas las AZ para mantener la disponibilidad si se produce un problema en un zona.

¿Cómo funciona si ejecuto una aplicación con estado?

Si ejecuta una aplicación con estado, debe evaluar su tolerancia a errores en función del caso de uso y la arquitectura. Si tiene una arquitectura o un patrón, ya sea activo o en espera, es posible que haya instancias en las que el elemento activo se encuentre en una AZ dañada. En la aplicación, si el elemento en espera no se activa, es posible que tenga problemas con la aplicación. También es posible que tenga problemas cuando se lancen nuevos pods de Kubernetes en AZ en buen estado, ya que no se podrán conectar a los volúmenes persistentes enlazados a la AZ dañada.

Funciona esta característica con Karpenter?

Actualmente, Karpenter no es compatible con el cambio de zona y el cambio de zona automático de ARC en Amazon EKS. Si una AZ está dañada, para ajustar la configuración de NodePool de Karpenter correspondiente, puede eliminar la AZ en mal estado a fin de que los nuevos nodos de trabajo solo se lancen en las AZ en buen estado.

Funciona esta característica con EKS Fargate?

Esta característica no es compatible con EKS Fargate. De forma predeterminada, cuando EKS Fargate reconoce un evento de salud en una zona, los pods preferirán ejecutarse en las otras AZ.

Amazon EKS afectará el plano de control de Kubernetes administrado?

No, de manera predeterminada Amazon EKS ejecuta y escala el plano de control de Kubernetes en varias AZ para garantizar una alta disponibilidad. El cambio de zona y el cambio de zona automático de ARC solo actúan en el plano de datos de Kubernetes.

Hay algún costo asociado a esta nueva característica?

Puede utilizar el cambio de zona y el cambio de zona automático de ARC en su clúster de Amazon EKS sin ningún costo adicional. Sin embargo, continuará pagando por las instancias aprovisionadas y le recomendamos encarecidamente que escale previamente el plano de datos de Kubernetes antes de utilizar esta característica. Debe plantearse que haya un equilibrio entre el costo y la disponibilidad de la aplicación.

Recursos adicionales

- Cómo funciona un cambio de zona
- Prácticas recomendadas para los cambios de zona en ARC
- Recursos y acontecimientos compatibles con el cambio de zona y el cambio de zona automático
- Operación de cargas de trabajo resilientes en Amazon EKS
- Eliminación del retraso en el escalado de nodos de Kubernetes con la prioridad de los pods y el sobreaprovisionamiento
- Escalabilidad de los pods de CoredNS para un tráfico de DNS elevado

Activación del cambio de zona de EKS para evitar zonas de disponibilidad deterioradas

El Controlador de recuperación de aplicaciones de Amazon (ARC) le ayuda a administrar y coordinar la recuperación de sus aplicaciones en todas las zonas de disponibilidad (AZ) y funciona con muchos servicios, incluido Amazon EKS. Con la compatibilidad de EKS con el cambio de zona de ARC, puede alejar el tráfico de red dentro del clúster de una AZ afectada. También puede autorizar que AWS supervise el estado de sus zonas de disponibilidad y desvíe de forma temporal el tráfico de la red de una zona de disponibilidad en mal estado, en su nombre.

Cómo se utiliza el cambio de zona de EKS:

1. Habilite su clúster de EKS con el Controlador de recuperación de aplicaciones de Amazon (ARC). Esto se hace a nivel del clúster, mediante la consola de Amazon EKS, la AWS CLI, CloudFormation o eksctl.
2. Una vez activado, puede administrar los cambios de zona o los cambios de zona automáticos mediante la consola de ARC, la AWS CLI o las API de cambio de zona y cambio de zona automático.

Tenga en cuenta que después de registrar un clúster de EKS con ARC, aún debe configurar ARC. Por ejemplo, puede usar la consola de ARC para configurar el cambio de zona automático.

Para obtener información más detallada sobre cómo funciona el cambio de zona de EKS y cómo diseñar sus cargas de trabajo para administrar las zonas de disponibilidad dañadas, consulte the section called “Información sobre el cambio de zona”.

Consideraciones

- El modo automático de EKS no es compatible con el controlador de recuperación de aplicaciones de Amazon, el cambio de zona ni con el cambio automático de zona.
- Recomendamos esperar al menos 60 segundos entre las operaciones de cambio de zona para garantizar el procesamiento adecuado de cada solicitud.

  Cuando intenta realizar cambios de zona en sucesión rápida (dentro de un intervalo de 60 segundos entre ellos), el servicio de Amazon EKS puede no procesar correctamente todas las solicitudes de cambio. Esto se debe al mecanismo de sondeo actual que actualiza el estado de zona del clúster. Si necesita realizar varios cambios de zona, asegúrese de que haya suficiente tiempo entre las operaciones para que el sistema procese cada cambio.

¿Qué es el controlador de recuperación de aplicaciones de Amazon?

El controlador de recuperación de aplicaciones de Amazon (ARC) lo ayuda a prepararse y realizar operaciones de recuperación más rápidas para las aplicaciones que se ejecutan en AWS. El cambio de zona permite recuperarse rápidamente de los problemas de la zona de disponibilidad (AZ), ya que desvían temporalmente el tráfico de un recurso compatible de una AZ a otra AZ en buen estado en la región de AWS.

Obtenga más información sobre el Controlador de recuperación de aplicaciones de Amazon (ARC)

¿Qué es el cambio de zona?

El cambio de zona es una capacidad de ARC que permite mover el tráfico de un recurso, como un clúster de EKS o un equilibrador de carga elástico, fuera de una zona de disponibilidad a una región de AWS para mitigar un problema y recuperar la aplicación de manera rápida. Puede optar por cambiar el tráfico, por ejemplo, si una implementación incorrecta provoca problemas de latencia o porque la zona de disponibilidad está deteriorada. Un cambio de zona no requiere pasos de configuración avanzados.

Más información sobre el cambio de zona de ARC

¿Qué es el cambio de zona automático?

El cambio de zona automático es una función de ARC que se puede habilitar para autorizar que AWS traslade el tráfico de una zona de disponibilidad para recursos compatibles a zonas de disponibilidad en buen estado de la región AWS, en su nombre. AWS inicia un cambio automático cuando la telemetría interna indica que hay una alteración en una AZ de una región que podría afectar a los clientes. La telemetría interna incorpora métricas de varios orígenes, incluida la red de AWS y los servicios Amazon EC2 y Elastic Load Balancing.

AWS finaliza los cambios automáticos cuando los indicadores muestran que ya no hay ningún problema o potencial inconveniente.

Más información sobre el cambio de zona automático de ARC

¿Qué hace el EKS durante un cambio automático?

EKS actualiza las configuraciones de red para evitar dirigir el tráfico a las zonas de disponibilidad deterioradas. Además, si se utilizan grupos de nodos administrados, EKS solo lanzará nuevos nodos en las AZ en buen estado durante un cambio de zona. Cuando el turno caduque o se cancele, las configuraciones de red se restaurarán para incluir la AZ que anteriormente se había detectado como defectuosa.

Más información sobre el cambio de zona de Amazon EKS.

Registre un clúster de EKS con el Controlador de recuperación de aplicaciones de Amazon (ARC) (consola de AWS)

1. Busque el nombre y la región del clúster de EKS que desea registrar con el ARC.
2. Navegue hasta la consola de EKS en esa región y seleccione su clúster.
3. Seleccione la pestaña Información general en la página Información del clúster.
4. En el encabezado Cambio de zona, seleccione el botón Administrar.
5. Seleccione activar o desactivar el cambio de zona de EKS.

Ahora su clúster EKS está registrado con ARC.

Si desea que AWS detecte y evite las zonas de disponibilidad deterioradas, debe configurar el cambio de zona automático de ARC. Por ejemplo, puede hacer esto en la consola de ARC.

Siguientes pasos

- Aprenda cómo activar el cambio de zona automático.
- Obtenga información sobre cómo iniciar de forma manual un cambio de zona.

Información sobre el funcionamiento del control de acceso en Amazon EKS

Obtenga información sobre cómo administrar el acceso a su clúster de Amazon EKS. El uso de Amazon EKS requiere saber cómo Kubernetes y AWS Identity and Access Management (AWS IAM) gestionan el control de acceso.

Esta sección incluye:

the section called “Acceso a la API de Kubernetes”: obtenga información sobre cómo permitir que las aplicaciones o los usuarios se autentiquen en la API de Kubernetes. Puede usar entradas de acceso, el aws-auth ConfigMap o un proveedor de OIDC externo.

the section called “Acceso a recursos del clúster”: obtenga información sobre cómo configurar la Consola de administración de AWS para que se comunique con su clúster de Amazon EKS. Utilice la consola para ver los recursos de Kubernetes en el clúster, como espacios de nombres, nodos y pods.

the section called “Acceso al clúster con kubectl”: obtenga información sobre cómo configurar kubectl para que se comunique con su clúster de Amazon EKS. Use la CLI de AWS para crear un archivo kubeconfig.

the section called “Acceso de la carga de trabajo a AWS ”: obtenga información sobre cómo asociar una cuenta de servicio de Kubernetes con roles de AWS IAM. Puede usar los roles de IAM o de Pod Identity para las cuentas de servicio (IRSA).

Tareas comunes

- Conceda a los desarrolladores acceso a la API de Kubernetes. Visualice los recursos de Kubernetes en la Consola de administración de AWS.
  - Solución: utilice las entradas de acceso para asociar los permisos de RBAC de Kubernetes a los roles o usuarios de AWS IAM.
- Configure kubectl para que se comunique con un clúster de Amazon EKS mediante credenciales de AWS.
  - Solución: utilice la CLI de AWS para crear un archivo kubeconfig.
- Utilice un proveedor de identidad externo, como Ping Identity, para autenticar usuarios en la API de Kubernetes.

  - Solución: vincule un proveedor de OIDC externo.
- Conceda a las cargas de trabajo de su clúster de Kubernetes la capacidad de llamar a las API de AWS.
  - Solución: use Pod Identity para asociar un rol de AWS IAM a una cuenta de servicio de Kubernetes.

Introducción

- Obtenga información sobre cómo funcionan las cuentas de servicio de Kubernetes.
- Revise el modelo de control de acceso basado en roles (RBAC) de Kubernetes
- Para obtener más información sobre la gestión del acceso a los recursos de AWS, consulte la Guía del usuario de AWS IAM. Como alternativa, haga una formación introductoria gratuita sobre el uso de AWS IAM.

Consideraciones para el modo automático de EKS

El modo automático de EKS se integra con las entradas de acceso de EKS EKS y EKS Pod Identity.

- El modo automático de EKS utiliza entradas de acceso para conceder permisos de Kubernetes al plano de control de EKS. Por ejemplo, las políticas de acceso permiten al modo automático de EKS leer información sobre puntos de conexión y servicios de red.
  - No puede desactivar las entradas de acceso en un clúster de modo automático de EKS.
  - Si lo desea, puede habilitar `aws-auth ConfigMap`.
  - Las entradas de acceso para el modo automático de EKS se configuran automáticamente. Puede ver estas entradas de acceso, pero no puede modificarlas.
  - Si utiliza una NodeClass para crear un rol de IAM de nodo personalizado, deberá crear una entrada de acceso para el rol mediante la política de acceso AmazonEKSAutoNodePolicy.
- Si quiere conceder permisos de carga de trabajo para los servicios de AWS, utilice EKS Pod Identity.
  - No necesita instalar el agente de Pod Identity en los clusters de modo automático de EKS.

Concesión a los usuarios y roles de IAM de acceso a las API de Kubernetes

Su clúster tiene un punto de conexión de la API de Kubernetes. Kubectl usa esta API. Puede autenticarse en esta API mediante dos tipos de identidades:

- Una entidad principal de AWS Identity and Access Management (IAM) (rol o usuario): este tipo requiere autenticación ante IAM. Los usuarios pueden iniciar sesión en AWS como un usuario de IAM o con una identidad federada mediante las credenciales proporcionadas a través de una fuente de identidad. Los usuarios solo pueden iniciar sesión con una identidad federada si su administrador previamente configuró la federación de identidades mediante los roles de IAM. Cuando los usuarios acceden a AWS mediante la federación, están asumiendo un rol de forma indirecta. Cuando los usuarios utilizan este tipo de identidad, puede hacer lo siguiente:
  - Puede asignarles permisos de Kubernetes para que puedan trabajar con los objetos de Kubernetes en su clúster. Para obtener más información sobre cómo asignar permisos a las entidades principales de IAM para que puedan acceder a los objetos de Kubernetes en su clúster, consulte the section called “Entradas de los registros”.
  - Puede asignarles permisos de IAM para que puedan trabajar con su clúster de Amazon EKS y sus recursos mediante la API de Amazon EKS, AWS CLI, AWS CloudFormation, Consola de administración de AWS o `eksctl`. Para obtener más información, consulte Acciones definidas por Amazon Elastic Kubernetes Service en la Referencia de autorización de servicios.
  - Los nodos se unen al clúster asumiendo un rol de IAM. El acceso al clúster mediante las entidades principales de IAM está habilitado por el Autenticador de IAM de AWS para Kubernetes, que se ejecuta en el plano de control de Amazon EKS.
- Un usuario en su propio proveedor de OpenID Connect (OIDC): este tipo requiere la autenticación de su proveedor de OIDC. Para obtener más información acerca de cómo configurar su propio proveedor de OIDC con su clúster de Amazon EKS, consulte the section called “Vinculación del proveedor de OIDC”. Cuando los usuarios utilizan este tipo de identidad, puede hacer lo siguiente:
  - Puede asignarles permisos de Kubernetes para que puedan trabajar con los objetos de Kubernetes en su clúster.
  - No puede asignarles permisos de IAM para que puedan trabajar con su clúster de Amazon EKS y sus recursos mediante la API de Amazon EKS, la AWS CLI, la AWS CloudFormation, la Consola de administración de AWS o el `eksctl`.

Puede utilizar ambos tipos de identidades con su clúster. El método de autenticación de IAM no se puede deshabilitar. El método de autenticación de OIDC es opcional.

Asociación de las identidades de IAM con los permisos de Kubernetes

El Autenticador de AWS IAM para Kubernetes se instala en el plano de control del clúster. Permite a las entidades principales de AWS Identity and Access Management (IAM) (roles y usuarios) acceder a los recursos de Kubernetes en su clúster. Puede permitir que las entidades principales de IAM accedan a los objetos de Kubernetes en su clúster mediante uno de los siguientes métodos:

- Creación de entradas de acceso: si su clúster tiene la versión de la plataforma igual o posterior a la que se indica en la sección Requisitos previos de la versión de Kubernetes del clúster, le recomendamos que utilice esta opción.

  Utilice las entradas de acceso para administrar los permisos de Kubernetes de las entidades principales de IAM ajenos al clúster. Puede agregar y administrar el acceso al clúster mediante la API de EKS, la Interfaz de la línea de comandos de AWS, los SDK de AWS, AWS CloudFormation y la Consola de administración de AWS. Esto significa que puede administrar los usuarios con las mismas herramientas con las que creó el clúster.

  Para comenzar, siga Cambio del modo de autenticación para usar entradas de acceso y, a continuación, Migración de las entradas existentes de aws-auth ConfigMap a entradas de acceso.
- Agregar entradas al **`aws-auth` `ConfigMap`** : si la versión de la plataforma del clúster es anterior a la que aparece en la sección de Requisitos previos, debe utilizar esta opción. Si la versión de la plataforma de su clúster es igual o posterior a la versión de la plataforma que aparece en la sección Requisitos previos de la versión de Kubernetes de su clúster y ha agregado entradas al `ConfigMap`, le recomendamos que migre esas entradas para acceder a las ellas. Sin embargo, no puede migrar las entradas que Amazon EKS haya agregado al `ConfigMap`, como las entradas para los roles de IAM utilizadas con los grupos de nodos administrados o los perfiles de Fargate. Para obtener más información, consulte the section called “Acceso a la API de Kubernetes”.
  - Si tiene que usar la opción de `ConfigMap` de `aws-auth`, puede añadir entradas al `ConfigMap` mediante el comando `eksctl create iamidentitymapping`. Para obtener más información, consulte Administrar usuarios y roles de IAM en la documentación de `eksctl`.

Establecimiento de nodos de autenticación del clúster

Cada clúster tiene un modo de autenticación. El modo de autenticación determina los métodos que puede utilizar para permitir que las entidades principales de IAM accedan a los objetos de Kubernetes en su clúster. Existen tres modos de autenticación.

> **Important**
>
> Una vez que se habilita el método de entrada de acceso, no se puede deshabilitar.
> Si el método `ConfigMap` no está habilitado durante la creación del clúster, no se podrá habilitar más adelante. Todos los clústeres creados antes de la introducción de las entradas de acceso tienen el método `ConfigMap` activado.
> Si usa nodos híbridos con el clúster, debe usar los modos de autenticación de clúster `API` o `API_AND_CONFIG_MAP`.

El **`aws-auth` `ConfigMap`** dentro del clúster

Este es el modo de autenticación original de los clústeres de Amazon EKS. La entidad principal de IAM que creó el clúster es el usuario inicial que puede acceder al clúster con `kubectl`. El usuario inicial debe añadir otros usuarios a la lista en el `ConfigMap` de `aws-auth` y asignar los permisos que afecten a los demás usuarios del clúster. Estos otros usuarios no pueden administrar ni eliminar al usuario inicial, ya que no hay ninguna entrada en el `ConfigMap` que administrar.

Tanto el **`ConfigMap`** como las entradas de acceso

Con este modo de autenticación, puede utilizar ambos métodos para añadir las entidades principales de IAM al clúster. Tenga en cuenta que cada método almacena entradas independientes; por ejemplo, si agrega una entrada de acceso desde la AWSCLI, el `aws-auth` `ConfigMap` no se actualiza.

Solo entradas de acceso

Con este modo de autenticación, puede utilizar la API de EKS, la Interfaz de la línea de comandos de AWS, los SDK de AWS, AWS CloudFormation y la Consola de administración de AWS para administrar el acceso al clúster de las entidades principales de IAM.

Cada entrada de acceso tiene un tipo y puede utilizar la combinación de un enlace de acceso para limitar a la entidad principal a un espacio de nombres específico y una política de acceso

para establecer políticas de permisos preconfiguradas y reutilizables. Como alternativa, puede usar el tipo STANDARD y los grupos de RBAC de Kubernetes para asignar permisos personalizados.

Modo de autenticación	Métodos
Solo `ConfigMap` (`CONFIG_MAP`)	`aws-auth` `ConfigMap`
API de EKS y `ConfigMap` (`API_AND_CONFIG_MAP`)	entradas de acceso en la API de EKS, la Interfaz de línea de comandos de AWS, los SDK de AWS, AWS CloudFormation, la Consola de administración de AWS y `aws-auth` `ConfigMap`
Solo la API de EKS (`API`)	entradas de acceso en la API de EKS, la Interfaz de línea de comandos de AWS, los SDK de AWS, AWS CloudFormation y la Consola de administración de AWS

> **Note**
>
> El modo automático de Amazon EKS requiere entradas de acceso.

Concesión de acceso para los usuarios de IAM a las entradas de acceso de Kubernetes con EKS

Esta sección está diseñada para mostrarle cómo administrar el acceso de la entidad principal de IAM a los clústeres de Kubernetes en Amazon Elastic Kubernetes Service (EKS) utilizando entradas y políticas de acceso. Encontrará detalles sobre cómo cambiar los modos de autenticación, cómo migrar desde entradas de `aws-auth` ConfigMap heredadas, cómo crear, actualizar y eliminar entradas de acceso, cómo asociar políticas a entradas, cómo revisar los permisos de políticas predefinidas y los requisitos previos y las consideraciones clave para la administración segura del acceso.

Descripción general

Las entradas de acceso de EKS son la mejor forma de conceder acceso a la API de Kubernetes a los usuarios. Por ejemplo, puede utilizar entradas de acceso para conceder a los desarrolladores acceso para utilizar kubectl. Básicamente, una entrada de acceso de EKS asocia un conjunto de permisos de Kubernetes a una identidad de IAM, como un rol de IAM. Por ejemplo, un desarrollador puede asumir un rol de IAM y utilizarlo para autenticarse en un clúster de EKS.

Características

- Autenticación y autorización centralizadas: controla el acceso a los clústeres de Kubernetes directamente a través de las API de Amazon EKS, lo que elimina la necesidad de cambiar entre las API de AWS y de Kubernetes para obtener permisos de usuario.
- Administración detallada de los permisos: utiliza entradas y políticas de acceso para definir permisos detallados para las entidades principales de AWS IAM, incluida la capacidad de modificar o revocar los permisos de administrador del clúster a su creador.
- Integración con la herramienta de IaC: admite infraestructura como herramientas de código, como AWS CloudFormation, Terraform y AWS CDK, lo que permite definir las configuraciones de acceso durante la creación del clúster.
- Recuperación de errores de configuración: permite restaurar el acceso al clúster a través de la API de Amazon EKS sin acceso directo a la API de Kubernetes.
- Reducción de gastos y seguridad mejorada: centraliza las operaciones para reducir los gastos al tiempo que aprovecha las características de AWS IAM, como el registro de auditorías de CloudTrail y la autenticación multifactor.

Cómo adjuntar permisos

Puede asociar permisos de Kubernetes a entradas de acceso dos maneras:

- Utilice una política de acceso. Las políticas de acceso son plantillas de permisos de Kubernetes predefinidas mantenidas por AWS. Para obtener más información, consulte the section called “Revisión de las políticas de acceso”.
- Haga referencia a un grupo de Kubernetes. Si asocia una identidad de IAM a un grupo de Kubernetes, podrá crear recursos de Kubernetes que concedan permisos al grupo. Para obtener más información, consulte Utilización de la autorización de RBAC en la documentación de Kubernetes.

Consideraciones

Al habilitar las entradas de acceso de EKS en los clústeres existentes, se debe tener en cuenta lo siguiente:

- Comportamiento de los clústeres heredados: para los clústeres creados antes de la introducción de las entradas de acceso (aquellos con versiones de la plataforma iniciales anteriores a las especificadas en los requisitos de versión de la plataforma), EKS crea automáticamente una entrada de acceso que refleja los permisos preexistentes. Esta entrada incluye la identidad de IAM que creó originalmente el clúster y los permisos administrativos otorgados a esa identidad durante la creación del clúster.
- Gestión del **`aws-auth`** ConfigMap heredado: si su clúster se basa en el `aws-auth` ConfigMap heredado para la administración del acceso, solo se crea automáticamente la entrada de acceso del creador del clúster original al habilitar las entradas de acceso. Los roles o permisos adicionales agregados al ConfigMap (por ejemplo, roles de IAM personalizados para desarrolladores o servicios) no se migran automáticamente. Para solucionarlo, se deben crear manualmente las entradas de acceso correspondientes.

Introducción

1. Determine la identidad de IAM y la política de acceso que desea utilizar.
   - the section called “Revisión de las políticas de acceso”
2. Habilite las entradas de acceso de EKS en el clúster. Confirme que tiene una versión de la plataforma compatible.
   - the section called “Modo de autenticación”
3. Cree una entrada de acceso que asocie una identidad de IAM a un permiso de Kubernetes.
   - the section called “Creación de entradas de acceso”
4. Autentíquese en el clúster mediante la identidad de IAM.
   - the section called “Configurar AWS CLI”
   - the section called “Configure kubectl y eksctl”

Asociación de políticas de acceso a entradas de acceso

Puede asignar una o más políticas de acceso a las entradas de acceso del tipo `STANDARD`. Amazon EKS concede automáticamente a los demás tipos de entradas de acceso los permisos necesarios

para funcionar correctamente en el clúster. Las políticas de acceso de Amazon EKS incluyen permisos de Kubernetes, no permisos de IAM. Antes de asociar una política de acceso a una entrada de acceso, asegúrese de haberse familiarizado con los permisos de Kubernetes incluidos en cada política de acceso. Para obtener más información, consulte the section called “Revisión de las políticas de acceso”. Si ninguna de las políticas de acceso cumple con sus requisitos, no asocie una política de acceso a una entrada de acceso. En su lugar, especifique uno o más nombres de grupo para la entrada de acceso, y cree y administre objetos de control de acceso basados en roles de Kubernetes. Para obtener más información, consulte the section called “Creación de entradas de acceso”.

- Una entrada de acceso existente. Para crear uno, consulte the section called “Creación de entradas de acceso”.
- Un rol o usuario de AWS Identity and Access Management con los siguientes permisos: `ListAccessEntries`, `DescribeAccessEntry`, `UpdateAccessEntry`, `ListAccessPolicies`, `AssociateAccessPolicy` y `DisassociateAccessPolicy`. Para obtener más información, consulte Acciones definidas por Amazon Elastic Kubernetes Service en la Referencia de autorizaciones del servicio.

Antes de asociar políticas de acceso con entradas de acceso, tenga en cuenta los siguientes requisitos:

- Puede asociar varias políticas de acceso a cada entrada de acceso, pero solo puede asociar cada política a una entrada de acceso una vez. Si asocia varias políticas de acceso, la entidad principal de IAM de la entrada de acceso tendrá todos los permisos incluidos en todas las políticas de acceso asociadas.
- Puede limitar una política de acceso a todos los recursos de un clúster o especificar el nombre de uno o más espacios de nombres de Kubernetes. Puede utilizar caracteres comodín para el nombre de un espacio de nombres. Por ejemplo, si desea limitar una política de acceso a todos los espacios de nombres que comiencen con `dev-`, puede especificar `dev-*` como nombre de un espacio de nombres. Asegúrese de que los espacios de nombres existan en su clúster y de que la ortografía coincida con el nombre real del espacio de nombres del clúster. Amazon EKS no confirma la ortografía ni la existencia de los espacios de nombres del clúster.
- Puede cambiar el alcance del acceso de una política de acceso después de asociarla a una entrada de acceso. Si limitó la política de acceso a los espacios de nombres de Kubernetes, puede agregar y eliminar espacios de nombres para la asociación, según sea necesario.

- Si asocia una política de acceso a una entrada de acceso que también tenga nombres de grupo especificados, la entidad principal de IAM tendrá todos los permisos en todas las políticas de acceso asociadas. También tiene todos los permisos en cualquier objeto `Role` o `ClusterRole` de Kubernetes que se especifique en cualquier objeto `Role` y `RoleBinding` de Kubernetes que especifican los nombres de grupo.
- Si ejecuta el comando `kubectl auth can-i --list`, no verá ningún permiso de Kubernetes asignado por las políticas de acceso asociadas a una entrada de acceso para la entidad principal de IAM que esté utilizando cuando ejecute el comando. El comando solo muestra los permisos de Kubernetes si los concedió en los objetos `Role` o `ClusterRole` de Kubernetes que vinculó a los nombres de grupo o al nombre de usuario que especificó para una entrada de acceso.
- Si se hace pasar por un usuario o un grupo de Kubernetes al interactuar con los objetos de Kubernetes del clúster, por ejemplo, al utilizar el comando `kubectl` con `--as` *`username`* o `--as-group` *`group-name`* , forzará el uso de una autorización RBAC de Kubernetes. Como resultado, la entidad principal de IAM no tiene permisos asignados por ninguna política de acceso asociada a la entrada de acceso. Los únicos permisos de Kubernetes que tiene el usuario o grupo al que está suplantando la entidad principal de IAM son los permisos de Kubernetes que concedió en los objetos `Role` o `ClusterRole` de Kubernetes, los cuales vinculó a los nombres de grupo o al nombre de usuario. Para que su entidad principal de IAM tenga los permisos de las políticas de acceso asociadas, no suplante un usuario o un grupo de Kubernetes. La entidad principal de IAM también seguirá teniendo todos los permisos que le haya concedido en los objetos `Role` o `ClusterRole` de Kubernetes que usted haya vinculado con los nombres de grupo o el nombre de usuario que especificó para la entrada de acceso. Para obtener más información, consulte User impersonation en la documentación de Kubernetes.

Puede asociar una política de acceso a una entrada de acceso mediante la Consola de administración de AWS o la AWS CLI.

Consola de administración de AWS

1. Abra la consola de Amazon EKS.
2. Elija el nombre del clúster que tenga una entrada de acceso a la que desea asociar una política de acceso.
3. Elija la pestaña Acceso.
4. Si el tipo de entrada de acceso es Estándar, puede asociar o desasociar las políticas de acceso de Amazon EKS. Si el tipo de entrada de acceso no es Estándar, entonces esta opción no está disponible.

5. Elija Asociar política de acceso.
6. En Nombre de la política, seleccione la política con los permisos que desea que tenga la entidad principal de IAM. Para ver los permisos incluidos en cada política, consulte the section called "Revisión de las políticas de acceso".
7. En Alcance del acceso, elija un alcance del acceso. Si elige Clúster, los permisos de la política de acceso se otorgan a la entidad principal de IAM para los recursos de todos los espacios de nombres de Kubernetes. Si elige espacio de nombres de Kubernetes, luego puede elegir Agregar nuevo espacio de nombres. En el campo Espacio de nombres que aparece, puede ingresar el nombre de un espacio de nombres de Kubernetes en el clúster. Si quiere que la entidad principal de IAM tenga los permisos en varios espacios de nombres, puede introducir varios espacios de nombres.
8. Seleccione Añadir política de acceso.

AWS CLI

1. La versión `2.12.3` o posterior, o bien, la versión `1.27.160` o posterior de la AWS interfaz de la línea de comandos (AWS CLI) instalada y configurada en su dispositivo o AWS CloudShell. Para comprobar su versión actual, utilice `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente. Para actualizarla, consulte Instalación de la CLI de AWS en su directorio principal en la Guía del usuario de AWS CloudShell.
2. Ver las políticas de acceso disponibles.

```
aws eks list-access-policies --output table
```

Un ejemplo de salida sería el siguiente.

```
-------------------------------------------------------------------------------------------------
|                                        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          ||
|+-------------------------------------------------------------------
+------------------------------+|
```

Para ver los permisos incluidos en cada política, consulte the section called “Revisión de las políticas de acceso”.

3. Ver las entradas de acceso existentes. Reemplace *my-cluster* por el nombre de su clúster.

```
aws eks list-access-entries --cluster-name my-cluster
```

Un ejemplo de salida sería el siguiente.

```
{
    "accessEntries": [
        "arn:aws:iam::111122223333:role/my-role",
        "arn:aws:iam::111122223333:user/my-user"
    ]
}
```

4. Asocie una política de acceso a una entrada de acceso. El siguiente ejemplo asocia la política de acceso de `AmazonEKSViewPolicy` a una entrada de acceso. Siempre que el rol de IAM *my-role* intente acceder a los objetos de Kubernetes del clúster, Amazon EKS autorizará al rol a usar los permisos en la política para acceder únicamente a los objetos de Kubernetes en los espacios de nombres *my-namespace1* y *my-namespace2* de Kubernetes. Reemplace *my-cluster* por el nombre de su clúster, *111122223333* por el ID de su cuenta de AWS y *my-role* por el nombre

del rol de IAM para el que desea que Amazon EKS autorice el acceso a los objetos del clúster de Kubernetes.

```
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
```

Si desea que la entidad principal de IAM tenga los permisos en todo el clúster, reemplace `type=namespace,namespaces=my-namespace1,my-namespace2` con `type=cluster`. Si desea asociar varias políticas de acceso a la entrada de acceso, ejecute el comando varias veces, cada una con una política de acceso única. Cada política de acceso asociada tiene su propio alcance.

> **Note**
>
> Si más adelante desea cambiar el alcance de una política de acceso asociada, vuelva a ejecutar el comando anterior con el nuevo alcance. Por ejemplo, si quisiera eliminar *my-namespace2*, debería volver a ejecutar el comando usando solamente `type=namespace,namespaces=my-namespace1` . Si quisiera cambiar el alcance de `namespace` a `cluster`, debería volver a ejecutar el comando usando `type=cluster` y eliminar `type=namespace,namespaces=my-namespace1,my-namespace2` .

5. Determine qué políticas de acceso están asociadas a una entrada de acceso.

```
aws eks list-associated-access-policies --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role
```

Un ejemplo de salida sería el siguiente.

```
{
    "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"
        }
    ]
}
```

En el ejemplo anterior, la entidad principal de IAM para esta entrada de acceso tiene permisos de visualización en todos los espacios de nombres del clúster y permisos de administrador en dos espacios de nombres de Kubernetes.

6. Desasocie una política de acceso de una entrada de acceso. En este ejemplo, la política de `AmazonEKSAdminPolicy` está disociada de una entrada de acceso. Sin embargo, la entidad principal de IAM conserva los permisos de la política de acceso de `AmazonEKSViewPolicy` para los objetos en los espacios de nombres *`my-namespace1`* y *`my-namespace2`*, ya que esa política de acceso no está disociada de la entrada de acceso.

```
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
```

Para ver una lista de las políticas de acceso disponibles, consulte the section called “Revisión de las políticas de acceso”.

Migración de las entradas existentes de `aws-auth ConfigMap` a entradas de acceso

Si ha agregado entradas al `ConfigMap` de `aws-auth` en su clúster, le recomendamos que cree entradas de acceso para las entradas existentes en su `ConfigMap` de `aws-auth`. Después de crear las entradas de acceso, puede eliminarlas de su `ConfigMap`. No puede asociar políticas de acceso a las entradas del `ConfigMap` de `aws-auth`. Si desea asociar políticas de acceso a sus entidades principales de IAM, cree entradas de acceso.

> **Important**
>
> - Cuando un clúster está en modo de autenticación de `API_AND_CONFIGMAP` y hay una asignación para el mismo rol de IAM tanto en `aws-auth ConfigMap` como en las entradas de acceso, el rol utilizará la asignación de la entrada de acceso para la autenticación. Las entradas de acceso tienen prioridad sobre las entradas de `ConfigMap` de la misma entidad principal de IAM.
> - Antes de eliminar las entradas `aws-auth ConfigMap` existentes que Amazon EKS creó para un grupo de nodos administrado o un perfil de Fargate en su clúster, verifique si las entradas de acceso correctas para esos recursos específicos existen en su clúster de Amazon EKS. Si elimina las entradas que Amazon EKS creó en el `ConfigMap` sin tener las entradas de acceso equivalentes, el clúster no funcionará correctamente.

Requisitos previos

- Familiaridad con las entradas de acceso y las políticas de acceso. Para obtener más información, consulte the section called “Entradas de los registros” y the section called “Asociación de políticas de acceso”.
- Un clúster existente con una versión de la plataforma que es igual o posterior a las versiones enumeradas en los requisitos previos del tema the section called “Entradas de los registros”.
- La versión `0.215.0` o posterior de la herramienta de línea de comandos de `eksctl` instalada en su dispositivo o AWS CloudShell. Para instalar o actualizar `eksctl`, consulte la sección de Instalación en la documentación de `eksctl`.
- Permisos de Kubernetes para modificar el `ConfigMap aws-auth` en el espacio de nombres `kube-system`.
- Un rol o usuario de AWS Identity and Access Management con los siguientes permisos: `CreateAccessEntry` y `ListAccessEntries`. Para obtener más información, consulte

Acciones definidas por Amazon Elastic Kubernetes Service en la Referencia de autorizaciones del servicio.

`eksctl`

1. Visualización de las entradas existentes en su `aws-auth ConfigMap`. Reemplace *`my-cluster`* por el nombre de su clúster.

```
eksctl get iamidentitymapping --cluster my-cluster
```

Un ejemplo de salida sería el siguiente.

```
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 “Creación de entradas de acceso” para cualquiera de las entradas del `ConfigMap` que haya creado y que se mostraron en el resultado anterior. Al crear las entradas de acceso, asegúrese de especificar los mismos valores para `ARN`, `USERNAME`, `GROUPS` y `ACCOUNT` que aparecen en el resultado. En el resultado de ejemplo, crearía entradas de acceso para todas las entradas, excepto las dos últimas, ya que Amazon EKS creó esas entradas para un perfil de Fargate y un grupo de nodos administrados.
3. Elimine las entradas del `ConfigMap` para cualquier entrada de acceso que haya creado. Si no elimina la entrada del `ConfigMap`, la configuración de la entrada de acceso para el ARN de entidad principal de IAM anula la entrada del `ConfigMap`. Reemplace *`111122223333`* por el ID de su cuenta de AWS y *`EKS-my-cluster-my-namespace-Viewers`* por el nombre del rol en la

entrada de su `ConfigMap`. Si la entrada que va a eliminar es para un usuario de IAM y no para un rol de IAM, reemplace el `role` por el `user` y *`EKS-my-cluster-my-namespace-Viewers`* por el nombre de usuario.

```
eksctl delete iamidentitymapping --arn arn:aws:iam::111122223333:role/EKS-my-cluster-my-namespace-Viewers --cluster my-cluster
```

Revisión de los permisos de la política de acceso

Las políticas de acceso incluyen `rules` que contienen `verbs` (permisos) y `resources` de Kubernetes. Las políticas de acceso no incluyen los permisos ni los recursos de IAM. Al igual que los objetos `Role` y `ClusterRole` de Kubernetes, las políticas de acceso solo incluyen `rules` `allow`. No puede modificar el contenido de una política de acceso. No puede crear sus propias políticas de acceso. Si los permisos de las políticas de acceso no satisfacen sus necesidades, cree objetos RBAC de Kubernetes y especifique los nombres de grupo para las entradas de acceso. Para obtener más información, consulte the section called “Creación de entradas de acceso”. Los permisos contenidos en las políticas de acceso son similares a los permisos de los roles del clúster orientados a los usuarios de Kubernetes. Para obtener más información, consulte User-facing roles en la documentación de Kubernetes.

Enumeración de todas las políticas

Utilice cualquiera de las políticas de acceso que se enumeran en esta página o recupere una lista de todas las políticas de acceso disponibles mediante la CLI de AWS:

```
aws eks list-access-policies
```

El resultado esperado debería ser similar al siguiente (abreviado para mayor brevedad):

```
{
    "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
    ]
}
```

AmazonEKSAdminPolicy

Esta política de acceso incluye permisos que otorgan a una entidad principal de IAM la mayoría de los permisos a los recursos. Cuando se asocia a una entrada de acceso, su alcance de acceso suele ser uno o más espacios de nombres de Kubernetes. Si desea que una entidad principal de IAM tenga acceso de administrador a todos los recursos de su clúster, asocie la política de acceso de the section called “AmazonEKSClusterAdminPolicy” a su entrada de acceso.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminPolicy`

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
`apps`	`daemonsets` , `deployments` , `deployments/rollback` , `deployments/scale` , `replicasets` , `replicasets/scale`, `statefulsets` , `statefulsets/scale`	`create`, `delete`, `deletecollection` , `patch`, `update`
`apps`	`controllerrevisions` , `daemonsets` , `daemonset`	`get`, `list`, `watch`

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
	s/status , deployments , deployments/scale , deployments/status , replicasets , replicasets/scale , replicasets/status , statefulsets , statefulsets/scale , statefulsets/status	
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

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
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

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
rbac.authorization .k8s.io	rolebindings , roles	create, delete, deletecol lection , get, list, patch, update, watch
	configmaps , endpoints , persistentvolumecl aims , persisten tvolumeclaims/stat us , pods, replicati oncontrollers , replicationcontrol lers/scale , serviceac counts , services, services/status	get,list, watch
	pods/attach , pods/exec , pods/portforward , pods/proxy , secrets, services/proxy	get, list, watch
	configmaps , events, persistentvolumecl aims , replicati oncontrollers , replicationcontrol lers/scale , secrets, serviceaccounts , services, services/ proxy	create, delete, deletecol lection , patch, update
	pods, pods/attach , pods/exec , pods/port forward , pods/proxy	create, delete, deletecol lection , patch, update

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
	`serviceaccounts`	`impersonate`
	`bindings`, `events`, `limitranges` , `namespaces/status` , `pods/log`, `pods/status` , `replicationcontrollers/status` , `resourcequotas` , `resourcequotas/status`	`get`, `list`, `watch`
	`namespaces`	`get`,`list`, `watch`

AmazonEKSClusterAdminPolicy

Esta política de acceso incluye permisos que permiten a un administrador de la entidad principal de IAM acceder a un clúster. Cuando se asocia a una entrada de acceso, su alcance de acceso suele ser el clúster, en lugar de un espacio de nombres de Kubernetes. Si desea que una entidad principal de IAM tenga un alcance administrativo más limitado, considere la posibilidad de asociar la política de acceso de the section called “AmazonEKSAdminPolicy” a su entrada de acceso.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy`

Grupos de API de Kubernetes	nonResourceURLs de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
*		*	*
	*		*

AmazonEKSAdminViewPolicy

Esta política de acceso incluye permisos que permiten a una entidad principal de IAM acceder para enumerar/visualizar todos los recursos en un clúster. Tenga en cuenta que esto incluye Kubernetes Secrets.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminViewPolicy`

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
*	*	`get`, `list`, `watch`

AmazonEKSEditPolicy

Esta política de acceso incluye permisos que permiten a una entidad principal de IAM editar la mayoría de los recursos de Kubernetes.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSEditPolicy`

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
`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` , `replicase`	`get`, `list`, `watch`

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
	ts/status , statefuls ets , statefulsets/ scale , statefulsets/ status	
autoscaling	horizontalpodautos calers , horizonta lpodautoscalers/st atus	get, list, watch
autoscaling	horizontalpodautos calers	create, delete, deletecol lection , patch, update
batch	cronjobs, jobs	create, delete, deletecol lection , patch, update
batch	cronjobs, cronjobs/ status , jobs, jobs/stat us	get, list, watch
discovery.k8s.io	endpointslices	get, list, watch
extensions	daemonsets , deploymen ts , deployments/ rollback , deploymen ts/scale , ingresses , networkpolicies , replicasets , replicase ts/scale , replicati oncontrollers/scale	create, delete, deletecol lection , patch, update

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
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

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
	pods, pods/attach , pods/exec , pods/port forward , 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

AmazonEKSViewPolicy

Esta política de acceso incluye permisos que permiten a una entidad principal de IAM ver la mayoría de los recursos de Kubernetes.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSViewPolicy`

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
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

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
	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 , resourcequotas/status	get, list, watch
	namespaces	get, list, watch

AmazonEKSSecretReaderPolicy

Esta política de acceso incluye permisos que permiten a una entidad principal de IAM leer secretos de Kubernetes.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSSecretReaderPolicy`

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
	`secrets`	`get`, `list`, `watch`

AmazonEKSAutoNodePolicy

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSAutoNodePolicy`

Esta política incluye los siguientes permisos que permiten a los componentes de Amazon EKS realizar las siguientes tareas:

- `kube-proxy`: supervise los puntos de conexión y los servicios de red, y administre los eventos relacionados. Esto habilita la funcionalidad de proxy de red a nivel de clúster.
- `ipamd`: administre los recursos de red de AWS VPC y las interfaces de red de contenedores (CNI). Esto permite al daemon de administración de direcciones IP manejar la red de pods.
- `coredns`: acceda a recursos de detección de servicios, como puntos de conexión y servicios. Esto habilita la funcionalidad de proxy de red a nivel de clúster.
- `ebs-csi-driver`: trabaje con recursos relacionados con el almacenamiento para los volúmenes de Amazon EBS. Esto permite el aprovisionamiento dinámico y la conexión de volúmenes persistentes.
- `neuron`: supervise los nodos y los pods de los dispositivos de AWS Neuron. Esto permite administrar los aceleradores de AWS Inferentia y Trainium.
- `node-monitoring-agent`: acceda a los diagnósticos y eventos de los nodos. Esto permite la supervisión del estado del clúster y la recopilación de diagnósticos.

Cada componente utiliza una cuenta de servicio dedicada y está restringido únicamente a los permisos necesarios para su función específica.

Si especifica manualmente un rol de IAM de nodo en una NodeClass, tiene que crear una entrada de acceso que asocie el nuevo rol de IAM de nodo a esta política de acceso.

AmazonEKSBlockStoragePolicy

> **Note**
>
> Esta política está destinada exclusivamente a los roles vinculados a servicios de AWS y no se puede utilizar con roles administrados por el cliente.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSBlockStoragePolicy`

Esta política incluye permisos que permiten a Amazon EKS administrar la elección de líderes y los recursos de coordinación para las operaciones de almacenamiento:

- `coordination.k8s.io`: cree y administre objetos de arrendamiento para la elección del líder. Esto permite a los componentes de almacenamiento de EKS coordinar sus actividades en todo el clúster mediante un mecanismo de elección de líderes.

La política está limitada a recursos específicos de arrendamiento utilizados por los componentes de almacenamiento de EKS para evitar accesos conflictivos a otros recursos de coordinación en el clúster.

Amazon EKS crea automáticamente una entrada de acceso con esta política de acceso para el rol de IAM del clúster cuando el modo automático está habilitado, garantizando que los permisos necesarios estén configurados correctamente para que la capacidad de almacenamiento en bloques funcione adecuadamente.

AmazonEKSLoadBalancingPolicy

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSLoadBalancingPolicy`

Esta política incluye permisos que permiten a Amazon EKS administrar recursos de elección de líderes para el equilibrio de carga:

- `coordination.k8s.io`: cree y administre objetos de arrendamiento para la elección del líder. Esto permite a los componentes de equilibrio de carga de EKS coordinar las actividades entre varias réplicas mediante la elección de un líder.

El alcance de la política se aplica específicamente a los recursos de arrendamiento de equilibrio de carga para garantizar una coordinación adecuada y, al mismo tiempo, impedir el acceso a otros recursos de arrendamiento en el clúster.

Amazon EKS crea automáticamente una entrada de acceso con esta política de acceso para el rol de IAM del clúster cuando se habilita el modo automático, lo que garantiza que se disponga de los permisos necesarios para que la capacidad de red funcione correctamente.

AmazonEKSNetworkingPolicy

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSNetworkingPolicy`

Esta política incluye permisos que permiten a Amazon EKS administrar recursos de elección de líderes para las redes:

- `coordination.k8s.io`: cree y administre objetos de arrendamiento para la elección del líder. Esto permite a los componentes de red de EKS coordinar las actividades de asignación de direcciones IP mediante la elección de un líder.

El alcance de la política se aplica específicamente a los recursos de arrendamiento de red para garantizar una coordinación adecuada y, al mismo tiempo, impedir el acceso a otros recursos de arrendamiento del clúster.

Amazon EKS crea automáticamente una entrada de acceso con esta política de acceso para el rol de IAM del clúster cuando se habilita el modo automático, lo que garantiza que se disponga de los permisos necesarios para que la capacidad de red funcione correctamente.

AmazonEKSComputePolicy

> **Note**
>
> Esta política está destinada exclusivamente a los roles vinculados a servicios de AWS y no se puede utilizar con roles administrados por el cliente.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSComputePolicy`

Esta política incluye permisos que permiten a Amazon EKS administrar recursos de elección de líderes para operaciones de computación:

- `coordination.k8s.io`: cree y administre objetos de arrendamiento para la elección del líder. Esto permite a los componentes de computación de EKS coordinar las actividades de escalado de nodos mediante la elección de un líder.

El alcance de la política se aplica específicamente a los recursos de arrendamiento de administración de computación, al tiempo que se permite el acceso de lectura básico (`get`,`watch`) a todos los recursos de arrendamiento del clúster.

Amazon EKS crea automáticamente una entrada de acceso con esta política de acceso para el rol de IAM del clúster cuando se habilita el modo automático, lo que garantiza que se disponga de los permisos necesarios para que la capacidad de red funcione correctamente.

AmazonEKSBlockStorageClusterPolicy

> **Note**
>
> Esta política está destinada exclusivamente a los roles vinculados a servicios de AWS y no se puede utilizar con roles administrados por el cliente.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSBlockStorageClusterPolicy`

Esta política concede los permisos necesarios para la capacidad de almacenamiento en bloque del modo automático de Amazon EKS. Permite una administración eficaz de los recursos de almacenamiento en bloque dentro de los clústeres de Amazon EKS. La política incluye los siguientes permisos:

Administración de controladores de CSI:

- Cree, lea, actualice y elimine controladores CSI, específicamente para el almacenamiento en bloques.

Administración de los volúmenes:

- Enumerar, vigilar, crear, actualizar, aplicar parches y eliminar volúmenes persistentes.
- Enumerar, vigilar y actualizar las reclamaciones de volumen persistentes.
- Aplique parches a los estados de reclamación de volúmenes persistentes.

Interacción entre nodos y pods:

- Lea la información sobre nodos y pods.
- Administre los eventos relacionados con las operaciones de almacenamiento.

Clases y atributos de almacenamiento:

- Lea las clases de almacenamiento y los nodos CSI.
- Lea las clases de atributos de volumen.

Conexiones de volúmenes:

- Enumerar, vigilar y modificar las conexiones de volúmenes y sus estados.

Operaciones de instantáneas:

- Administre las instantáneas de volúmenes, el contenido de las instantáneas y las clases de instantáneas.
- Gestione operaciones para instantáneas de grupos de volúmenes y recursos relacionados.

Esta política está diseñada para admitir la administración integral del almacenamiento en bloques dentro de los clústeres de Amazon EKS que se ejecutan en modo automático. Combina permisos para diversas operaciones, como el aprovisionamiento, la conexión, el cambio de tamaño y la creación de instantáneas de volúmenes de almacenamiento en bloque.

Amazon EKS crea automáticamente una entrada de acceso con esta política de acceso para el rol de IAM del clúster cuando el modo automático está habilitado, garantizando que los permisos necesarios estén configurados correctamente para que la capacidad de almacenamiento en bloques funcione adecuadamente.

AmazonEKSComputeClusterPolicy

> **Note**
>
> Esta política está destinada exclusivamente a los roles vinculados a servicios de AWS y no se puede utilizar con roles administrados por el cliente.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSComputeClusterPolicy`

Esta política concede los permisos necesarios para la capacidad de administración de computación del modo automático de Amazon EKS. Permite orquestar y escalar eficazmente los recursos de computación dentro de los clústeres de Amazon EKS. La política incluye los siguientes permisos:

Administración de nodos:

- Cree, lea, actualice, elimine y administre el estado de NodePools y NodeClaims.
- Administre NodeClasses, lo que incluye la creación, modificación y eliminación.

Programación y administración de recursos:

- Acceso de lectura a pods, nodos, volúmenes persistentes, reclamaciones de volúmenes persistentes, controladores de replicación y espacios de nombres.
- Acceso de lectura a clases de almacenamiento, nodos de CSI y conexiones de volúmenes.
- Enumere y observe implementaciones, conjuntos de daemon, conjuntos de réplicas y conjuntos con estado.
- Lea los presupuestos de disrupción de pods.

Control de eventos:

- Cree, lea y administre los eventos de clústeres.

Desaprovisionamiento de nodos y expulsión de nodos:

- Actualice, aplique parches y elimine nodos.
- Cree expulsiones de pods y elimínelos cuando sea necesario.

Administración de definiciones personalizadas de recursos (CRD):

- Creación de nuevas CRD.
- Administre CRD específicas relacionadas con la administración de nodos (NodeClasses, NodePools, NodeClaims y NodeDiagnostics).

Esta política se ha diseñado para admitir una administración de computación completa en los clústeres de Amazon EKS que se ejecutan en modo automático. Combina permisos para diversas operaciones, como el aprovisionamiento de nodos, la programación, el escalado y la optimización de recursos.

Amazon EKS crea automáticamente una entrada de acceso con esta política de acceso para el rol de IAM del clúster cuando se habilita el modo automático, lo que garantiza que se disponga de los permisos necesarios para que la capacidad de administración de computación funcione correctamente.

AmazonEKSLoadBalancingClusterPolicy

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSLoadBalancingClusterPolicy`

Esta política concede los permisos necesarios para la capacidad de equilibrio de carga del modo automático de Amazon EKS. Permite administrar y configurar eficazmente los recursos de equilibrio de carga dentro de los clústeres de Amazon EKS. La política incluye los siguientes permisos:

Administración de eventos y recursos:

- Crear eventos y aplicarles parches.
- Acceso de lectura a pods, nodos, puntos de conexión y espacios de nombres.
- Actualice los estados de los pods.

Administración de servicios e ingresos:

- Administración completa de los servicios y sus estados.
- Control completo de los ingresos y sus estados.
- Acceso de lectura a segmentos de puntos de conexión y clases de ingreso.

Vinculaciones de grupos de destino:

- Cree y modifique vinculaciones de grupos de destino y sus estados.
- Acceso de lectura a los parámetros de las clases de ingreso.

Administración de definiciones personalizadas de recursos (CRD):

- Cree y lea todas las CRD.
- Administración específica de las CRD de targetgroupbindings.eks.amazonaws.com e ingressclassparams.eks.amazonaws.com.

Configuración de webhook:

- Cree y lea las configuraciones de webhook que mutan y validan.
- Administre la configuración de eks-load-balancing-webhook.

Esta política está diseñada para admitir la administración integral del equilibrio de carga en los clústeres de Amazon EKS que se ejecutan en modo automático. Combina permisos para diversas operaciones, como la exposición de servicios, el enrutamiento de ingreso y la integración con servicios de equilibrio de carga de AWS.

Amazon EKS crea automáticamente una entrada de acceso con esta política de acceso para el rol de IAM del clúster cuando se habilita el modo automático, lo que garantiza que se disponga de los permisos necesarios para que la capacidad de equilibrio de carga funcione correctamente.

AmazonEKSNetworkingClusterPolicy

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSNetworkingClusterPolicy`

AmazonEKSNetworkingClusterPolicy

Esta política concede los permisos necesarios para la capacidad de conexión en red del modo automático de Amazon EKS. Permite administrar y configurar eficazmente los recursos de red dentro de los clústeres de Amazon EKS. La política incluye los siguientes permisos:

Administración de nodos y pods:

- Acceso de lectura a las clases de nodos y sus estados.
- Acceso de lectura a las NodeClaims y sus estados.
- Acceso de lectura a los pods.

Administración de nodos de CNI:

- Permisos para CNINodes y sus estados, que incluyen crear, leer, actualizar, eliminar y aplicar parches.

Administración de definiciones personalizadas de recursos (CRD):

- Cree y lea todas las CRD.
- Administración específica (actualizar, parchear, eliminar) del CRD cninodes.eks.amazonaws.com.

Evento de administración:

- Crear eventos y aplicarles parches.

Esta política está diseñada para admitir la administración integral de redes dentro de los clústeres de Amazon EKS que se ejecutan en modo automático. Combina permisos para varias operaciones, como la configuración de redes de nodos, la administración de la interfaz de red de contenedores (CNI) y la administración de recursos personalizados relacionados.

La política permite a los componentes de red interactuar con recursos relacionados con nodos, administrar configuraciones de nodos específicas de CNI y administrar recursos personalizados críticos para las operaciones de red en el clúster.

Amazon EKS crea automáticamente una entrada de acceso con esta política de acceso para el rol de IAM del clúster cuando se habilita el modo automático, lo que garantiza que se disponga de los permisos necesarios para que la capacidad de red funcione correctamente.

AmazonEKSHybridPolicy

> **Note**
>
> Esta política está destinada exclusivamente a los roles vinculados a servicios de AWS y no se puede utilizar con roles administrados por el cliente.

Esta política de acceso incluye permisos que conceden a EKS acceso a los nodos de un clúster. Cuando se asocia a una entrada de acceso, su alcance de acceso suele ser el clúster, en lugar de un espacio de nombres de Kubernetes. Los Nodos híbridos de Amazon EKS utilizan esta política.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSHybridPolicy`

Grupos de API de Kubernetes	nonResourceURLs de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
*		nodes	list

AmazonEKSClusterInsightsPolicy

> **Note**
>
> Esta política está destinada exclusivamente a los roles vinculados a servicios de AWS y no se puede utilizar con roles administrados por el cliente.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSClusterInsightsPolicy`

Esta política otorga permisos de solo lectura a Información del clúster de Amazon EKS. La política incluye los siguientes permisos:

Acceso a los nodos: - Enumeración y visualización de los nodos del clúster - Lectura de la información de estado de los nodos

Acceso al DaemonSet: - Acceso de lectura a la configuración de kube-proxy

El servicio de EKS de Información del clúster administra automáticamente esta política. Para obtener más información, consulte the section called “Información sobre clústeres”.

AWSBackupFullAccessPolicyForBackup

ARN – `arn:aws:eks::aws:cluster-access-policy/AWSBackupFullAccessPolicyForBackup`

AWSBackupFullAccessPolicyForBackup

Esta política otorga los permisos necesarios para que AWS Backup administre y cree copias de seguridad del clúster de EKS. Esta política incluye los permisos siguientes:

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
*	*	list, get

AWSBackupFullAccessPolicyForRestore

ARN – arn:aws:eks::aws:cluster-access-policy/AWSBackupFullAccessPolicyForRestore

AWSBackupFullAccessPolicyForRestore

Esta política otorga los permisos necesarios para que AWS Backup administre y restaure copias de seguridad del clúster de EKS. Esta política incluye los permisos siguientes:

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
*	*	list, get, create

AmazonEKSACKPolicy

ARN – arn:aws:eks::aws:cluster-access-policy/AmazonEKSACKPolicy

Esta política otorga los permisos necesarios para que la capacidad Controladores de AWS para Kubernetes (ACK) administre los recursos de AWS desde Kubernetes. La política incluye los siguientes permisos:

Administración de recursos personalizados de ACK:

- Acceso total a todos los recursos personalizados del servicio de ACK en más de 50 servicios de AWS, incluidos S3, RDS, DynamoDB, Lambda, EC2, etc.
- Cree, lea, actualice y elimine definiciones de recursos personalizados de ACK.

Acceso al espacio de nombres:

- Acceso de lectura a los espacios de nombres para organizar los recursos.

Elección de líder:

- Cree y lea asignaciones de coordinación para la elección de líder.
- Actualice y elimine las asignaciones específicas de controladores de servicios de ACK.

Evento de administración:

- Cree eventos y aplíqueles parches para las operaciones de ACK.

Esta política está diseñada para admitir la administración integral de recursos de AWS a través de las API de Kubernetes. Amazon EKS crea automáticamente una entrada de acceso con esta política de acceso para el rol de IAM de la capacidad que se proporciona cuando se crea la capacidad de ACK.

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
	`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.service`	*	*

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
s.k8s.aws , cloudtrail.services.k8s.aws , cloudwatch.services.k8s.aws , cloudwatchlogs.services.k8s.aws , codeartifact.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.servi		

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
ces.k8s.aws , lambda.services.k8s.aws , memorydb.services.k8s.aws , mq.services.k8s.aws , networkfirewall.services.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 ,		

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
`wafv2.services.k8s.aws`		
`coordination.k8s.io`	`leases`	`create`, `get`, `list`, `watch`
`coordination.k8s.io`	`leases` (solo asignaciones de controladores de servicios de ACK específicos)	`delete`, `update`, `patch`
	`events`	`create`, `patch`
`apiextensions.k8s.io`	`customresourcedefinitions`	`*`

AmazonEKSArgoCDClusterPolicy

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSArgoCDClusterPolicy`

Esta política otorga los permisos de clúster necesarios para que la capacidad de Argo CD pueda detectar recursos y administrar objetos del ámbito del clúster. La política incluye los siguientes permisos:

Administración de espacios de nombres:

- Cree, lea, actualice y elimine espacios de nombres para la administración de los espacios de nombres de las aplicaciones.

Administración de definiciones de recursos personalizados:

- Administre CRD específicos de Argo CD (Aplicaciones, AppProjects, ApplicationSets).

Detección de API:

- Lea el acceso a los puntos de conexión de la API de Kubernetes para la detección de recursos.

Esta política está diseñada para admitir las operaciones de clústeres de Argo CD, lo que incluye la administración del espacio de nombres y la instalación de CRD. Amazon EKS crea automáticamente una entrada de acceso con esta política de acceso para el rol de IAM de la capacidad que se proporciona cuando se crea la capacidad de Argo CD.

Grupos de API de Kubernetes	nonResourceURLs de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
		`namespaces`	`create`, `get`, `update`, `patch`, `delete`
`apiextensions.k8s.io`		`customresourcedefinitions`	`create`
`apiextensions.k8s.io`		`customresourcedefinitions` (solo CRD de Argo CD)	`get`, `update`, `patch`, `delete`
	`/api`, `/api/*`, `/apis`, `/apis/*`		`get`

AmazonEKSArgoCDPolicy

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSArgoCDPolicy`

Esta política otorga los permisos de espacio de nombres necesarios para que la capacidad de Argo CD pueda implementar y administrar aplicaciones. La política incluye los siguientes permisos:

Administración de secretos:

- Acceso completo a los secretos para las credenciales de Git y los secretos de clústeres.

Acceso a ConfigMap:

- Lea el acceso a ConfigMaps para enviar advertencias si los clientes intentan utilizar ConfigMaps de Argo CD no compatibles.

Evento de administración:

- Lea y cree eventos para el seguimiento del ciclo de vida de la aplicación.

Administración de recursos de Argo CD:

- Obtenga acceso completo a aplicaciones, ApplicationSets y AppProjects.
- Administre los finalizadores y el estado de los recursos de Argo CD.

Esta política está diseñada para admitir las operaciones de espacios de nombres de Argo CD, lo que incluye la implementación y la administración de aplicaciones. Amazon EKS crea automáticamente una entrada de acceso con esta política de acceso para el rol de IAM de la capacidad que se proporciona cuando se crea la capacidad de Argo CD, en el ámbito del espacio de nombres de Argo CD.

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
	`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/finali`	`*`

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
	`zers` , `appprojects/status`	

AmazonEKSKROPolicy

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSKROPolicy`

Esta política otorga los permisos necesarios para que la capacidad de kro (Kube Resource Orchestrator) cree y administre las API de Kubernetes personalizadas. La política incluye los siguientes permisos:

Administración de recursos de kro:

- Acceso completo a todos los recursos de kro, lo que incluye ResourceGraphDefinitions y las instancias de recursos personalizados.

Administración de definiciones de recursos personalizados:

- Cree, lea, actualice y elimine CRD para API personalizadas definidas por ResourceGraphDefinitions.

Elección de líder:

- Cree y lea asignaciones de coordinación para la elección de líder.
- Actualice y elimine la asignación de controladores de kro.

Evento de administración:

- Cree eventos y aplíqueles parches para las operaciones de kro.

Esta política está diseñada para admitir la composición integral de recursos y la administración de API personalizadas a través de kro. Amazon EKS crea automáticamente una entrada de acceso con esta política de acceso para el rol de IAM de la capacidad que se proporciona cuando se crea la capacidad de kro.

Grupos de API de Kubernetes	Recursos de Kubernetes	Verbos de Kubernetes (permisos)
`kro.run`	`*`	`*`
`apiextensions.k8s.io`	`customresourcedefinitions`	`*`
`coordination.k8s.io`	`leases`	`create`, `get`, `list`, `watch`
`coordination.k8s.io`	`leases` (solo asignación de controladores de kro)	`delete`, `update`, `patch`
	`events`	`create`, `patch`

Actualizaciones de la política de acceso

Vea detalles sobre las actualizaciones de las políticas de acceso desde su introducción. Para obtener alertas automáticas sobre cambios en esta página, suscríbase a la fuente RSS en Historial de documentos.

Cambio	Descripción	Fecha
Adición de políticas para capacidades de EKS	Publicación de `AmazonEKSACKPolicy`, `AmazonEKSArgoCDClusterPolicy`, `AmazonEKSArgoCDPolicy` y `AmazonEKSKROPolicy` para administrar capacidades de EKS	22 de noviembre de 2025
Añada `AmazonEKSSecretReaderPolicy`	Adición de una nueva política para el acceso de solo lectura a los secretos	6 de noviembre de 2025

Cambio	Descripción	Fecha
Agregue una política para Información del clúster de EKS	Publicar `AmazonEKSClusterInsightsPolicy`	2 de diciembre de 2024
Agregar políticas para Amazon EKS Hybrid	Publicar `AmazonEKSHybridPolicy`	2 de diciembre de 2024
Agregar políticas para el modo automático de Amazon EKS	Estas políticas de acceso conceden al rol de IAM del clúster y al rol de IAM del nodo permiso para llamar a las API de Kubernetes. AWS las utiliza para automatizar tareas rutinarias para los recursos de almacenamiento, computación y redes.	2 de diciembre de 2024
Añada `AmazonEKSAdminViewPolicy`	Agregue una nueva política para ampliar el acceso a las vistas, incluidos recursos como Secretos.	23 de abril de 2024
Se introdujeron políticas de acceso.	Amazon EKS introdujo políticas de acceso.	29 de mayo de 2023

Cambio del modo de autenticación para utilizar las entradas de acceso

Para empezar a utilizar las entradas de acceso, cambie el modo de autenticación del clúster a los modos `API_AND_CONFIG_MAP` o `API`. Esto añade la API para las entradas de acceso.

Consola de AWS

1. Abra la consola de Amazon EKS.
2. Elija el nombre del clúster en el que desea crear una entrada de acceso.
3. Elija la pestaña Acceso.

4. En Modo de autenticación se muestra el modo de autenticación actual del clúster. Si el modo indica API de EKS, ya puede agregar entradas de acceso y puede omitir los pasos restantes.
5. Elija Administrar acceso.
6. En Modo de autenticación de clústeres, seleccione un modo con la API de EKS. Tenga en cuenta que no puede volver a cambiar el modo de autenticación a un modo que elimine la API de EKS y las entradas de acceso.
7. Seleccione Save changes (Guardar cambios). Amazon EKS comienza a actualizar el clúster, el estado del clúster cambia a Actualizando y el cambio se registra en la pestaña Historial de actualizaciones.
8. Espere a que el estado del clúster vuelva a ser Activo. Cuando el clúster esté activo, puede seguir los pasos que se indican en the section called “Creación de entradas de acceso” para agregar acceso al clúster para las entidades principales de IAM.

AWS CLI

1. Instale AWS CLI, tal y como se describe en Instalación en la Guía del usuario de la interfaz de la línea de comandos de AWS.
2. Ejecute el siguiente comando. Reemplace *my-cluster* por el nombre de su clúster. Si desea deshabilitar permanentemente el método `ConfigMap`, reemplace`API_AND_CONFIG_MAP` por `API`.

   Amazon EKS comienza a actualizar el clúster, el estado del clúster cambia a ACTUALIZANDO y el cambio se registra en aws eks list-updates.

   ```
   aws eks update-cluster-config --name my-cluster --access-config
    authenticationMode=API_AND_CONFIG_MAP
   ```
3. Espere a que el estado del clúster vuelva a ser Activo. Cuando el clúster esté activo, puede seguir los pasos que se indican en the section called “Creación de entradas de acceso” para agregar acceso al clúster para las entidades principales de IAM.

Versión de la plataforma requerida

Para usar entradas de acceso, el clúster debe tener una versión de la plataforma igual o posterior a la que se indica en la siguiente tabla, o una versión de Kubernetes posterior a las versiones que se muestran en la tabla. Si su versión de Kubernetes no aparece en la lista, todas las versiones de la plataforma admiten entradas de acceso.

Versión de Kubernetes	Versión de la plataforma
No se muestra	Todos los compatibles
`1.30`	`eks.2`
`1.29`	`eks.1`
`1.28`	`eks.6`

Para obtener más información, consulte Versiones de la plataforma.

Creación de entradas de acceso

Antes de crear entradas de acceso, tenga en cuenta lo siguiente:

- Un modo de autenticación configurado correctamente. Consulte the section called “Modo de autenticación”.
- Una entrada de acceso incluye el Nombre de recurso de Amazon (ARN) de una sola entidad principal de IAM existente. No se puede incluir una entidad principal de IAM en más de una entrada de acceso. Consideraciones adicionales para el ARN que especifique:
  - Las prácticas recomendadas de IAM sugieren acceder al clúster mediante roles de IAM que tengan credenciales a corto plazo, en lugar de usuarios de IAM que tengan credenciales a largo plazo. Para obtener más información, consulte Solicitar que los usuarios humanos utilicen la federación con un proveedor de identidades para acceder a AWS mediante credenciales temporales en la Guía del usuario de IAM.
  - Si el ARN es para un rol de IAM, puede incluir una ruta. Los ARN de las entradas del `ConfigMap` de `aws-auth` no pueden incluir una ruta. Por ejemplo, su ARN puede ser `arn:aws:iam::<111122223333>:role/<development/apps/my-role>` o `arn:aws:iam::<111122223333>:role/<my-role>`.
  - Si el tipo de entrada de acceso es distinto a `STANDARD` (consulte la siguiente consideración sobre los tipos), el ARN debe estar en la misma cuenta de AWS que el clúster. Si el tipo es `STANDARD`, el ARN puede estar en la misma cuenta de AWS o en una cuenta diferente a la cuenta en la que se encuentra el clúster.
  - Después de crear la entrada de acceso, no se puede cambiar la entidad principal de IAM.

  - Si alguna vez elimina la entidad principal de IAM con este ARN, la entrada de acceso no se elimina automáticamente. Le recomendamos que elimine la entrada de acceso con un ARN para la entidad principal de IAM que desea eliminar. Si no elimina la entrada de acceso y vuelve a crear la entidad principal de IAM, la entrada de acceso no funcionará aunque tenga el mismo ARN. Esto se debe a que, aunque el ARN es el mismo para la entidad principal de IAM recreada, el `roleID` o `userID` (puede verlo con el comando `aws sts get-caller-identity` de la CLI de AWS) es diferente para la entidad principal de IAM recreada que para la entidad principal de IAM original. Aunque no vea el `roleID` o el `userID` de la entidad principal de IAM para una entrada de acceso, Amazon EKS lo almacena junto con la entrada de acceso.
- Cada entrada de acceso tiene un tipo. El tipo de entrada de acceso depende del tipo de recurso con el que está asociada y no define los permisos. Si no especifica ningún tipo, Amazon EKS establece automáticamente el tipo en `STANDARD`.
  - `EC2_LINUX`: para un rol de IAM utilizado con nodos autoadministrados de Linux o Bottlerocket
  - `EC2_WINDOWS`: para un rol de IAM utilizado con nodos autoadministrados de Windows
  - `FARGATE_LINUX`: para un rol de IAM utilizado con AWS Fargate (Fargate)
  - `HYBRID_LINUX`: para un rol de IAM utilizado con nodos híbridos
  - `STANDARD`: tipo predeterminado si no se especifica ninguno
  - `EC2`: para las clases de nodos personalizados del modo automático de EKS. Para obtener más información, consulte the section called “Creación de una entrada de acceso a una clase de nodos”.
  - Después de crear la entrada de acceso, no se puede cambiar el tipo.
- No es necesario crear una entrada de acceso para un rol de IAM que se utiliza para un grupo de nodos administrados o un perfil de Fargate. EKS creará entradas de acceso (si están habilitadas) o actualizará las asignaciones de configuración de autenticación (si las entradas de acceso no están disponibles).
- Si el tipo de entrada de acceso es `STANDARD`, puede especificar un nombre de usuario para la entrada de acceso. Si no especifica un valor para el nombre de usuario, Amazon EKS establece uno de los siguientes valores en función del tipo de entrada de acceso y de si la entidad principal de IAM que especificó es un rol de IAM o un usuario de IAM. A menos que tenga un motivo específico para especificar su propio nombre de usuario, le recomendamos que no especifique ninguno y deje que Amazon EKS lo genere automáticamente. Si especifica su propio nombre de usuario:
  - No puede empezar con `system:`, `eks:`, `aws:`, `amazon:` o `iam:`.

- Si el nombre de usuario corresponde a un rol de IAM, le recomendamos que añada `{{SessionName}}` o `{{SessionNameRaw}}` al final de este. Si añade `{{SessionName}}` o `{{SessionNameRaw}}` a su nombre de usuario, este debe incluir dos puntos antes de {{SessionName}}. Cuando se asume este rol, el nombre de la sesión de AWS STS especificada al asumir el rol se transfiere automáticamente al clúster y aparece en los registros de CloudTrail. Por ejemplo, no puede tener un nombre de usuario como `john{{SessionName}}`. El nombre de usuario tendría que ser `:john{{SessionName}}` o `jo:hn{{SessionName}}`. Los dos puntos deben estar antes de `{{SessionName}}`. El nombre de usuario generado por Amazon EKS en la siguiente tabla incluye un ARN. Como un ARN incluye dos puntos, cumple con este requisito. Los dos puntos no son obligatorios si no incluye `{{SessionName}}` en su nombre de usuario. Tenga en cuenta que, en `{{SessionName}}`, el carácter especial “@” se sustituye por “-” en el nombre de la sesión. `{{SessionNameRaw}}` mantiene todos los caracteres especiales en el nombre de la sesión.

Tipo de entidad principal de IAM	Tipo	Valor de nombre de usuario que Amazon EKS establece automáticamente
Usuario	`STANDARD`	El ARN del usuario. Ejemplo: `arn:aws:iam::<111122223333>:user/<my-user>`

Tipo de entidad principal de IAM	Tipo	Valor de nombre de usuario que Amazon EKS establece automáticamente
Rol	`STANDARD`	El ARN de STS del rol cuando se asume. Amazon EKS agrega `{{SessionName}}` al rol. Ejemplo: `arn:aws:sts::<111122223333>:assumed-role/<my-role>/{{SessionName}}` Si el ARN del rol que especificó contenía una ruta, Amazon EKS la elimina del nombre de usuario generado.
Rol	`EC2_LINUX` o `EC2_Windows`	`system:node:{{EC2PrivateDNSName}}`
Rol	`FARGATE_LINUX`	`system:node:{{SessionName}}`
Rol	`HYBRID_LINUX`	`system:node:{{SessionName}}`

Puede cambiar el nombre de usuario después de crear la entrada de acceso.

- Si el tipo de entrada de acceso es `STANDARD` y desea utilizar la autorización RBAC de Kubernetes, puede agregar uno o más nombres de grupo a la entrada de acceso. Después de crear una entrada de acceso, puede añadir y eliminar nombres de grupos. Para que la entidad principal de IAM tenga acceso a los objetos de Kubernetes del clúster, debe crear y administrar los objetos de autorización basados en roles (RBAC) de Kubernetes. Cree objetos `RoleBinding` o `ClusterRoleBinding` de Kubernetes en el clúster que especifique el nombre del grupo

como `subject` para `kind: Group`. Kubernetes autoriza el acceso de la entidad principal de IAM a cualquier objeto del clúster que haya especificado en un objeto `Role` o `ClusterRole` de Kubernetes que también haya especificado en el `roleRef` del enlace. Si especifica los nombres de grupo, recomendamos que esté familiarizado con los objetos de Autorizaciones basados en roles (RBAC) de Kubernetes. Para obtener más información, consulte Utilización de la autorización de RBAC en la documentación de Kubernetes.

> **Important**
>
> Amazon EKS no confirma que ninguno de los objetos de RBAC de Kubernetes en el clúster incluya alguno de los nombres de grupo que especifique. Por ejemplo, si crea una entrada de acceso para un grupo que no existe actualmente, EKS creará el grupo en lugar de arrojar un error.

En lugar de autorizar a Kubernetes para que la entidad principal de IAM acceda a los objetos de Kubernetes de su clúster, o además de ello, puede asociar las políticas de acceso de Amazon EKS a una entrada de acceso. Amazon EKS autoriza a las entidades principales de IAM a acceder a los objetos de Kubernetes del clúster con los permisos de la política de acceso. Puede limitar los permisos de una política de acceso a los espacios de nombres de Kubernetes que especifique. El uso de políticas de acceso no requiere que administre los objetos de RBAC de Kubernetes. Para obtener más información, consulte the section called “Asociación de políticas de acceso”.

- Si crea una entrada de acceso con el tipo `EC2_LINUX` o `EC2_Windows`, la entidad principal de IAM que crea la entrada de acceso debe tener el permiso `iam:PassRole`. Para obtener más información, consulte Concesión de permisos a un usuario para transferir un rol a un servicio de AWS en la Guía del usuario de IAM.
- Al igual que el comportamiento de IAM estándar, la creación y las actualizaciones de las entradas de acceso son eventualmente uniformes y pueden tardar varios segundos en hacerse efectivas una vez que la llamada inicial a la API se haya completado con éxito. Debe diseñar sus aplicaciones teniendo en cuenta estos posibles retrasos. Le recomendamos que no incluya las creaciones o las actualizaciones de las entradas de acceso en las rutas de código de gran importancia y alta disponibilidad de su aplicación. En su lugar, realice los cambios de en otra rutina de inicialización o configuración que ejecute con menos frecuencia. Además, asegúrese de verificar que los cambios se han propagado antes de que los flujos de trabajo de producción dependan de ellos.

- Las entradas de acceso no admiten roles vinculados a servicios. No puede crear entradas de acceso en las que el ARN principal sea un rol vinculado al servicio. Puede identificar los roles vinculados al servicio por su ARN, que está en el formato `arn:aws:iam::*:role/aws-service-role/*`.

Puede crear una entrada de acceso mediante la Consola de administración de AWS o la CLI de AWS.

Consola de administración de AWS

1. Abra la consola de Amazon EKS.
2. Elija el nombre del clúster en el que desea crear una entrada de acceso.
3. Elija la pestaña Acceso.
4. Elija Crear entrada de acceso.
5. En Entidad principal de IAM, seleccione un usuario o rol de IAM existente. Las prácticas recomendadas de IAM sugieren acceder al clúster mediante roles de IAM que tengan credenciales a corto plazo, en lugar de usuarios de IAM que tengan credenciales a largo plazo. Para obtener más información, consulte Solicitar que los usuarios humanos utilicen la federación con un proveedor de identidades para acceder a AWS mediante credenciales temporales en la Guía del usuario de IAM.
6. En Tipo, si la entrada de acceso es para el rol de nodo utilizado para los nodos Amazon EC2 autoadministrados, seleccione EC2 Linux o EC2 Windows. De lo contrario, acepte el valor predeterminado (Estándar).
7. Si el Tipo que ha elegido es Estándar y desea especificar un Nombre de usuario, introdúzcalo.
8. Si el Tipo que ha elegido es Estándar y desea utilizar la autorización RBAC de Kubernetes para la entidad principal de IAM, especifique uno o más nombres para los Grupos. Si no especifica ningún nombre de grupo y desea utilizar la autorización de Amazon EKS, puede asociar una política de acceso en un paso posterior o después de crear la entrada de acceso.
9. (Opcional) En Etiquetas, asigne etiquetas a la entrada de acceso. Por ejemplo, para facilitar la búsqueda de todos los recursos con la misma etiqueta.
10. Elija Siguiente.
11. En la página Agregar política de acceso, si el tipo que ha elegido es Estándar y quiere que Amazon EKS autorice a la entidad principal de IAM a tener permisos para los objetos de Kubernetes de su clúster, complete los siguientes pasos. En caso contrario, elija Siguiente.

a. En Nombre de la política, elija una política de acceso. No puede ver los permisos de las políticas de acceso, pero incluyen permisos similares a los de los objetos `ClusterRole` orientados al usuario de Kubernetes. Para obtener más información, consulte User-facing roles en la documentación de Kubernetes.

b. Seleccione una de las siguientes opciones:

- Clúster: elija esta opción si desea que Amazon EKS autorice a la entidad principal de IAM a tener los permisos de la política de acceso para todos los objetos de Kubernetes de su clúster.
- Espacio de nombres de Kubernetes: elija esta opción si desea que Amazon EKS autorice a la entidad principal de IAM a tener los permisos de la política de acceso para todos los objetos de Kubernetes en un espacio de nombres específico de Kubernetes en su clúster. En Espacio de nombres, ingrese el nombre del espacio de nombres de Kubernetes en el clúster. Si quiere añadir espacios de nombres adicionales, seleccione Añadir nuevo espacio de nombres e ingrese el nombre del espacio de nombres.

c. Si desea añadir políticas adicionales, seleccione Añadir política. Puede establecer el ámbito de cada política de forma diferente, pero puede añadir cada política solo una vez.

d. Elija Siguiente.

12 Revise la configuración de su entrada de acceso. Si algo parece incorrecto, seleccione Anterior para volver a repasar los pasos y corregir el error. Si la configuración es correcta, seleccione Crear.

AWS CLI

1. Instale la CLI de AWS, tal y como se describe en Instalación en la Guía del usuario de la interfaz de la línea de comandos de AWS.
2. Puede utilizar cualquiera de los siguientes ejemplos para crear entradas de acceso:

- Cree una entrada de acceso para un grupo de nodos autoadministrados de Amazon EC2 Linux. Reemplace *`my-cluster`* por el nombre del clúster, *`111122223333`* por el ID de la cuenta de AWS y *`EKS-my-cluster-self-managed-ng-1`* por el nombre del rol de IAM del nodo. Si el grupo de nodos es un grupo de nodos de Windows, sustituya *`EC2_Linux`* por `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
```

No puede utilizar la opción `--kubernetes-groups` cuando especifica un tipo que no sea `STANDARD`. No puede asociar una política de acceso a esta entrada de acceso porque su tipo es un valor distinto a `STANDARD`.

- Cree una entrada de acceso que permita a un rol de IAM, no utilizado por un grupo de nodos autoadministrados de Amazon EC2, autorizar a Kubernetes el acceso a su clúster. Reemplace *`my-cluster`* por el nombre de su clúster, *`111122223333`* por el ID de su cuenta de AWS y *`my-role`* por el nombre de su rol de IAM. Reemplace *`Espectadores`* por el nombre de un grupo que haya especificado en un objeto `RoleBinding` o `ClusterRoleBinding` de Kubernetes de su clúster.

```
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
```

- Cree una entrada de acceso que permita a un usuario de IAM autenticarse en su clúster. Este ejemplo se proporciona porque es posible, aunque las prácticas recomendadas de IAM sugieren acceder a su clúster mediante roles de IAM que tengan credenciales a corto plazo, en lugar de usuarios de IAM que tengan credenciales a largo plazo. Para obtener más información, consulte Solicitar que los usuarios humanos utilicen la federación con un proveedor de identidades para acceder a AWS mediante credenciales temporales en la Guía del usuario de IAM.

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:user/my-user --type STANDARD --username my-user
```

Si desea que este usuario tenga más acceso a su clúster que los permisos en los roles de detección de la API de Kubernetes, debe asociar una política de acceso a la entrada de acceso, ya que la opción `--kubernetes-groups` no se utiliza. Para obtener más información, consulte the section called “Asociación de políticas de acceso” y API discovery roles en la documentación de Kubernetes.

Actualización de entradas de acceso

Puede actualizar una entrada de acceso mediante la Consola de administración de AWS o la AWS CLI.

Consola de administración de AWS

1. Abra la consola de Amazon EKS.
2. Elija el nombre del clúster en el que desea crear una entrada de acceso.
3. Elija la pestaña Acceso.
4. Elija la entrada de acceso que desea actualizar.
5. Elija Editar.
6. En Nombre de usuario, puede cambiar el valor existente.
7. En Grupos, puede eliminar los nombres de los grupos existentes o añadir nuevos nombres de grupos. Si existen los siguientes nombres de grupos, no los elimine: system:nodes o system:bootstrappers. Si elimina estos grupos, puede provocar que el clúster no funcione correctamente. Si no especifica ningún nombre de grupo y desea utilizar la autorización de Amazon EKS, asocie una política de acceso en un paso posterior.
8. En Etiquetas, puede asignar etiquetas a la entrada de acceso. Por ejemplo, para facilitar la búsqueda de todos los recursos con la misma etiqueta. También puede eliminar las etiquetas existentes.
9. Elija Guardar cambios.
10. Si desea asociar una política de acceso a la entrada, consulte the section called “Asociación de políticas de acceso”.

AWS CLI

1. Instale la AWS CLI, tal y como se describe en Instalación en la Guía del usuario de la interfaz de la línea de comandos de AWS.
2. Para actualizar una entrada de acceso Sustituya *my-cluster* por el nombre de su clúster, *111122223333* por el ID de su cuenta de AWS y *EKS-my-cluster-my-namespace-Viewers* por el nombre de un rol de IAM.

```
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
```

No puede usar la opción de `--kubernetes-groups` si el tipo de entrada de acceso es un valor distinto a `STANDARD`. Tampoco puede asociar una política de acceso a una entrada de acceso de un tipo distinto a `STANDARD`.

Eliminación de entradas de acceso

Si descubre que ha eliminado una entrada de acceso por error, siempre podrá volver a crearla. Si la entrada de acceso que va a eliminar está asociada a alguna política de acceso, las asociaciones se eliminarán automáticamente. No es necesario desasociar las políticas de acceso de una entrada de acceso antes de eliminarla.

Puede eliminar una entrada de acceso de mediante la Consola de administración de AWS o la CLI de AWS.

Consola de administración de AWS

1. Abra la consola de Amazon EKS.
2. Elija el nombre del clúster del que desea eliminar una entrada de acceso.
3. Elija la pestaña Acceso.
4. En la lista de Entradas de acceso, elija la entrada de acceso que desea eliminar.
5. Elija Eliminar.
6. En el cuadro de diálogo de confirmación, elija Eliminar.

AWS CLI

1. Instale la CLI de AWS, tal y como se describe en Instalación la Guía del usuario de la interfaz de la línea de comandos de AWS.
2. Para eliminar una entrada de acceso, reemplace *my-cluster* por el nombre de su clúster, *111122223333* por el ID de su cuenta de AWS y *my-role* por el nombre del rol de IAM que ya no quiere que tenga acceso a su clúster.

```
aws eks delete-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role
```

Establecimiento de un nombre de usuario personalizado para las entradas de acceso de EKS

Al crear entradas de acceso para Amazon EKS, puede utilizar el nombre de usuario generado automáticamente o especificar un nombre de usuario personalizado. En esta página se explican ambas opciones y se le guía por la configuración de un nombre de usuario personalizado.

Descripción general

El nombre de usuario de una entrada de acceso se utiliza para identificar la entidad principal de IAM en las pistas de auditoría y los registros de Kubernetes. De forma predeterminada, Amazon EKS genera un nombre de usuario en función del ARN de la identidad de IAM, pero puede especificar un nombre de usuario personalizado si es necesario.

Generación de nombres de usuario predeterminados

Si no especifica ningún valor para el nombre de usuario, Amazon EKS genera automáticamente un nombre de usuario en función de la identidad de IAM:

- Para usuarios de IAM:
  - EKS establece el nombre de usuario de Kubernetes en el ARN del usuario de IAM
  - Ejemplo:

    ```
    {arn-aws}iam::<111122223333>:user/<my-user>
    ```
- Para roles de IAM:
  - EKS establece el nombre de usuario de Kubernetes en función del ARN del rol de IAM
  - El ARN de STS del rol cuando se asume. Amazon EKS agrega `{{SessionName}}` al rol. Si el ARN del rol que especificó contenía una ruta, Amazon EKS la elimina del nombre de usuario generado.
  - Ejemplo:

    ```
    {arn-aws}sts::<111122223333>:assumed-role/<my-role>/{{SessionName}}
    ```

A menos que tenga un motivo concreto para especificar su propio nombre de usuario, le recomendamos que no especifique ninguno y deje que Amazon EKS lo genere automáticamente.

Establecimiento de un nombre de usuario personalizado

Al crear una entrada de acceso, puede especificar un nombre de usuario personalizado mediante el parámetro `--username`:

```
aws eks create-access-entry --cluster-name <cluster-name> --principal-arn <iam-
identity-arn> --type STANDARD --username <custom-username>
```

Requisitos para los nombres de usuario personalizados

Si especifica un nombre de usuario personalizado:

- El nombre de usuario no puede empezar por `system:`, `eks:`, `aws:`, `amazon:` ni `iam:`.
- Si el nombre de usuario corresponde a un rol de IAM, le recomendamos que añada `{{SessionName}}` o `{{SessionNameRaw}}` al final de este.
  - Si añade `{{SessionName}}` o `{{SessionNameRaw}}` a su nombre de usuario, este debe incluir dos puntos antes de {{SessionName}}.

Creación de una entrada de acceso para un usuario o rol de IAM mediante una política de acceso y la AWS CLI

Cree entradas de acceso a Amazon EKS que utilicen políticas de acceso de EKS administradas de AWS para conceder permisos estandarizados a las identidades de IAM para acceder a los clústeres de Kubernetes y administrarlos.

Descripción general

Las entradas de acceso en Amazon EKS definen cómo las identidades de IAM (usuarios y roles) pueden acceder a sus clústeres de Kubernetes e interactuar con ellos. Al crear entradas de acceso con las políticas de acceso de EKS, puede hacer lo siguiente:

- Conceder permisos a roles o usuarios de IAM específicos para acceder a su clúster de EKS
- Controlar los permisos mediante políticas de acceso de EKS administradas de AWS que proporcionan conjuntos de permisos estandarizados y predefinidos
- Limitar los permisos a espacios de nombres específicos o a todo el clúster
- Simplificar la administración del acceso sin modificar `aws-auth` ConfigMap ni crear recursos de RBAC de Kubernetes
- Utilizar un enfoque integrado de AWS para el control de acceso de Kubernetes que trate los casos de uso más comunes y, al mismo tiempo, mantenga las prácticas recomendadas de seguridad

Este enfoque se recomienda para la mayoría de los casos de uso porque proporciona permisos estandarizados y administrados de AWS sin requerir ninguna configuración manual de RBAC de Kubernetes. Las políticas de acceso de EKS eliminan la necesidad de configurar manualmente los recursos de RBAC de Kubernetes y ofrecen conjuntos de permisos predefinidos que tratan los casos de uso más comunes.

Requisitos previos

- El modo de autenticación del clúster debe estar configurado para habilitar las entradas de acceso. Para obtener más información, consulte the section called “Modo de autenticación”.
- Instale y configure la AWS CLI, tal y como se describe en Instalación en la Guía del usuario de la Interfaz de la línea de comandos de AWS.

Paso 1: definición de la entrada de acceso

1. Busque el ARN de la identidad de IAM (por ejemplo, un usuario o rol) al que desea conceder permisos.
   - Cada identidad de IAM solo puede tener una entrada de acceso a EKS.
2. Determine si desea que los permisos de la política de acceso de Amazon EKS se apliquen solo a un espacio de nombres de Kubernetes específico o a todo el clúster.
   - Si desea limitar los permisos a un espacio de nombres específico, anote el nombre del espacio de nombres.
3. Seleccione la política de acceso de EKS que desee para la identidad de IAM. Esta política concede permisos en el clúster. Anote el ARN de la política.
   - Para ver una lista de políticas, consulte las políticas de acceso disponibles.
4. Determine si el nombre de usuario generado automáticamente es adecuado para la entrada de acceso o si necesita especificar un nombre de usuario manualmente.
   - AWS genera automáticamente este valor en función de la identidad de IAM. Puede configurar un nombre de usuario personalizado. Esto es visible en los registros de Kubernetes.
   - Para obtener más información, consulte the section called “Establecimiento de un nombre de usuario personalizado”.

Paso 2: creación de una entrada de acceso

Después de planificar la entrada de acceso, utilice la AWS CLI para crearla.

El siguiente ejemplo trata la mayoría de los casos de uso. Consulte la referencia de CLI para ver todas las opciones de configuración.

Adjuntará la política de acceso en el siguiente paso.

```
aws eks create-access-entry --cluster-name <cluster-name> --principal-arn <iam-
identity-arn> --type STANDARD
```

Paso 3: asociación de la política de acceso

El comando varía en función de si desea que la política se limite a un espacio de nombres de Kubernetes específico.

Necesita el ARN de la política de acceso. Consulte las políticas de acceso disponibles.

Creación de una política sin ámbito de espacio de nombres

```
aws eks associate-access-policy --cluster-name <cluster-name> --principal-arn <iam-
identity-arn> --policy-arn <access-policy-arn>
```

Creación con ámbito de espacio de nombres

```
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>
```

Pasos a seguir a continuación

- Creación de kubeconfig para poder usar kubectl con una identidad de IAM

Creación de una entrada de acceso mediante grupos de Kubernetes con la AWS CLI

Cree entradas de acceso de Amazon EKS que utilicen grupos de Kubernetes para la autorización y que requieran una configuración de RBAC manual.

> **Note**
>
> Para la mayoría de los casos de uso, recomendamos utilizar las políticas de acceso de EKS en lugar del enfoque de grupos de Kubernetes que se describe en esta página. Las políticas de acceso de EKS proporcionan una forma más sencilla e integrada con AWS de administrar el acceso sin requerir una configuración de RBAC manual. Utilice el enfoque de grupos de Kubernetes solo cuando necesite un control más detallado que el que ofrecen las políticas de acceso de EKS.

Descripción general

Las entradas de acceso definen cómo las identidades de IAM (usuarios y roles) acceden a sus clústeres de Kubernetes. El enfoque de grupos de Kubernetes concede a los usuarios o roles de IAM permiso para acceder al clúster de EKS a través de grupos de RBAC estándar de Kubernetes. Este método requiere crear y administrar recursos de RBAC de Kubernetes (Roles, RoleBindings, ClusterRoles y ClusterRoleBindings) y se recomienda cuando se necesitan conjuntos de permisos altamente personalizados, requisitos de autorización complejos o si se desea mantener patrones de control de acceso consistentes en los entornos híbridos de Kubernetes.

En este tema no se trata la creación de entradas de acceso para las identidades de IAM utilizadas para que las instancias de Amazon EC2 se unan a los clústeres de EKS.

Requisitos previos

- El modo de autenticación del clúster debe estar configurado para habilitar las entradas de acceso. Para obtener más información, consulte the section called “Modo de autenticación”.
- Instale y configure la AWS CLI, tal y como se describe en Instalación en la Guía del usuario de la Interfaz de la línea de comandos de AWS.
- Se recomienda haberse familiarizado con RBAC de Kubernetes. Para obtener más información, consulte Utilización de la autorización de RBAC en la documentación de Kubernetes.

Paso 1: definición de la entrada de acceso

1. Busque el ARN de la identidad de IAM (por ejemplo, un usuario o rol) al que desea conceder permisos.
   - Cada identidad de IAM solo puede tener una entrada de acceso a EKS.
2. Defina los grupos de Kubernetes que desea asociar a esta identidad de IAM.
   - Deberás crear o usar los recursos `Role`/`ClusterRole` y `RoleBinding`/`ClusterRoleBinding` de Kubernetes existentes que hagan referencia a estos grupos.
3. Determine si el nombre de usuario generado automáticamente es adecuado para la entrada de acceso o si necesita especificar un nombre de usuario manualmente.
   - AWS genera automáticamente este valor en función de la identidad de IAM. Puede configurar un nombre de usuario personalizado. Esto es visible en los registros de Kubernetes.
   - Para obtener más información, consulte the section called “Establecimiento de un nombre de usuario personalizado”.

Paso 2: creación de una entrada de acceso con grupos de Kubernetes

Tras planificar la entrada de acceso, utilice la AWS CLI para crearla con los grupos de Kubernetes correspondientes.

```
aws eks create-access-entry --cluster-name <cluster-name> --principal-arn <iam-
identity-arn> --type STANDARD --kubernetes-groups <groups>
```

Reemplace:

- `<cluster-name>` por el nombre del clúster de EKS
- `<iam-identity-arn>` por el ARN del rol o usuario de IAM
- `<groups>` por una lista de grupos de Kubernetes separados por comas (por ejemplo, “system:developers,system:readers”)

Consulte la referencia de CLI para ver todas las opciones de configuración.

Paso 3: configuración de RBAC de Kubernetes

Para que la entidad principal de IAM tenga acceso a los objetos de Kubernetes del clúster, debe crear y administrar los objetos de control de acceso basados en roles (RBAC) de Kubernetes:

1. Cree objetos `Role` o `ClusterRole` de Kubernetes que definan los permisos.
2. Cree objetos `RoleBinding` o `ClusterRoleBinding` de Kubernetes en el clúster que especifique el nombre del grupo como `subject` para `kind: Group`.

Para obtener información detallada sobre la configuración de grupos y permisos en Kubernetes, consulte Using RBAC Authorization en la documentación de Kubernetes.

Pasos a seguir a continuación

- Creación de kubeconfig para poder usar kubectl con una identidad de IAM

Concesión de acceso a Kubernetes con un ConfigMap para los usuarios de IAM

> **Important**
>
> `aws-auth ConfigMap` se ha quedado obsoleto. Para conocer el método recomendado a fin de administrar el acceso a las API de Kubernetes, consulte the section called “Entradas de los registros”.

El acceso al clúster mediante las entidades principales de IAM está habilitado por el AWSAutenticador de IAM para Kubernetes, que se ejecuta en el plano de control de Amazon EKS. El autenticador obtiene la información de la configuración del `ConfigMap` de `aws-auth`. Para todos las configuraciones del `ConfigMap` de `aws-auth`, consulte el formato de configuración completa en GitHub.

Agregar las entidades principales de IAM al clúster de Amazon EKS

Cuando se crea un clúster de Amazon EKS, la entidad principal de IAM que crea el clúster recibe permisos de `system:masters` de forma automática en la configuración del Control de acceso basado en roles (RBAC) del clúster en el plano de control de Amazon EKS. Esta entidad principal no aparece en ninguna configuración visible, así que asegúrese de realizar un seguimiento de la entidad principal que creó el clúster originalmente. Para conceder a entidades principales adicionales de IAM la capacidad de interactuar con el clúster, edite el `aws-auth ConfigMap` dentro de Kubernetes y cree un `rolebinding` o `clusterrolebinding` de Kubernetes con el nombre de un `group` que especifique en el `aws-auth ConfigMap`.

> **Note**
>
> Para obtener más información sobre la configuración del control de acceso basado en roles (RBAC) de Kubernetes, consulte Using RBAC Authorization en la documentación de Kubernetes.

1. Determine qué credenciales de `kubectl` utiliza para obtener acceso al clúster. En la computadora, puede ver qué credenciales de `kubectl` utiliza con el siguiente comando. Si no utiliza la ruta predeterminada, reemplace *`~/.kube/config`* por la ruta al archivo `kubeconfig`.

```
cat ~/.kube/config
```

Un ejemplo de salida sería el siguiente.

```
[...]
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
[...]
```

En la salida de ejemplo anterior, se configuran las credenciales de un usuario llamado *admin* para un clúster con el nombre *my-cluster*. Si este es el usuario que creó el clúster, entonces ya tiene acceso a él. Si no es el usuario el que creó el clúster, deberá completar los pasos restantes para habilitar el acceso al clúster para otras entidades principales de IAM. Según las prácticas recomendadas de IAM, se recomienda conceder permisos a los roles en lugar de a los usuarios. Puede ver qué otras entidades principales tienen acceso actualmente al clúster con el siguiente comando:

```
kubectl describe -n kube-system configmap/aws-auth
```

Un ejemplo de salida sería el siguiente.

```
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>
```

El ejemplo anterior es un valor predeterminado `aws-auth` `ConfigMap`. Solo el rol de la instancia de nodos tiene acceso al clúster.

2. Asegúrese de tener `roles` y `rolebindings` de Kubernetes o `clusterroles` y `clusterrolebindings` existentes a los que pueda asignar entidades principales de IAM. Para obtener más información sobre estos recursos, consulte Utilización de la autorización de RBAC en la documentación de Kubernetes.

   a. Vea sus Kubernetes existentes `roles` o `clusterroles`. Los `Roles` están asignados a un `namespace`, pero los `clusterroles` se ajustan al clúster.

   ```
   kubectl get roles -A
   ```

   ```
   kubectl get clusterroles
   ```

   b. Consulte los detalles de cualquier `role` o `clusterrole` devuelto en la salida anterior y confirme que tiene los permisos (`rules`) que desea que las entidades principales de IAM tengan en el clúster.

   Reemplace *`role-name`* con un nombre de `role` devuelto en el resultado del comando anterior. Reemplace *`kube-system`* con el espacio de nombres del `role`.

   ```
   kubectl describe role role-name -n kube-system
   ```

   Reemplace *`cluster-role-name`* con un nombre de `clusterrole` devuelto en el resultado del comando anterior.

   ```
   kubectl describe clusterrole cluster-role-name
   ```

   c. Vea sus Kubernetes existentes `rolebindings` o `clusterrolebindings`. Los `Rolebindings` están asignados a un `namespace`, pero los `clusterrolebindings` se ajustan al clúster.

```
kubectl get rolebindings -A
```

```
kubectl get clusterrolebindings
```

d. Vea los detalles de cualquier `rolebinding` o `clusterrolebinding` y confirme que tiene un `role` o `clusterrole` del paso anterior enumerado como `roleRef` y un nombre de grupo enumerado para `subjects`.

Reemplace *`role-binding-name`* con un nombre `rolebinding` devuelto en el resultado del comando anterior. Reemplace *`kube-system`* con `namespace` del `rolebinding`.

```
kubectl describe rolebinding role-binding-name -n kube-system
```

Un ejemplo de salida sería el siguiente.

```
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
```

Reemplace *`cluster-role-binding-name`* con un nombre `clusterrolebinding` devuelto en el resultado del comando anterior.

```
kubectl describe clusterrolebinding cluster-role-binding-name
```

Un ejemplo de salida sería el siguiente.

```
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. Edite el `ConfigMap` de `aws-auth`. Puede utilizar una herramienta como `eksctl` para actualizar el `ConfigMap` o puede actualizarlo manualmente editándolo.

> **Important**
>
> Recomendamos utilizar `eksctl`, u otra herramienta, para editar el `ConfigMap`. Para obtener información acerca de otras herramientas que puede utilizar, consulte Utilice herramientas para realizar cambios en el aws-authConfigMap en las guías de prácticas recomendadas de Amazon EKS. Un formato incorrecto de `aws-auth` `ConfigMap` puede provocar que pierda el acceso a su clúster.

- Vea los pasos para editar el configmap con eksctl.
- Vea los pasos para editar el configmap manualmente.

Edición de Configmap con Eksctl

1. Necesita la versión `0.215.0` o posterior de la herramienta de línea de comandos de `eksctl` instalada en su dispositivo o AWS CloudShell. Para instalar o actualizar `eksctl`, consulte la sección de Instalación en la documentación de `eksctl`.
2. Vea las asignaciones actuales en la `ConfigMap`. Reemplace *`my-cluster`* por el nombre de su clúster. Reemplace *`region-code`* por la región de AWS en la que se encuentra el clúster.

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

Un ejemplo de salida sería el siguiente.

```
ARN
            USERNAME                                GROUPS
 ACCOUNT
arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
```

3. Agregue una asignación para un rol. Reemplace *my-role* con el nombre de su rol. Reemplace *eks-console-dashboard-full-access-group* por el nombre del grupo especificado en su objeto `RoleBinding` o `ClusterRoleBinding` de Kubernetes. Reemplace *111122223333* por el ID de su cuenta. Puede reemplazar *admin* (administrador) con cualquier nombre que elija.

```
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**
>
> El ARN del rol no puede incluir una ruta como `role/my-team/developers/my-role`. El formato del ARN debe ser `arn:aws:iam::111122223333:role/my-role` . En este ejemplo, se necesita eliminar `my-team/developers/`.

Un ejemplo de salida sería el siguiente.

```
[...]
2022-05-09 14:51:20 [#]  adding identity "{arn-aws}iam::111122223333:role/my-role" to
 auth ConfigMap
```

4. Agregue una asignación para un usuario. Según las prácticas recomendadas de IAM, se recomienda conceder permisos a los roles en lugar de a los usuarios. Reemplace *my-user* por el nombre de usuario. Reemplace *eks-console-dashboard-restricted-access-group* por el nombre del grupo especificado en su objeto `RoleBinding` o `ClusterRoleBinding` de Kubernetes. Reemplace *111122223333* por el ID de su cuenta. Puede reemplazar *my-user* (mi usuario) con cualquier nombre que elija.

```
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
```

Un ejemplo de salida sería el siguiente.

```
[...]
2022-05-09 14:53:48 [#]  adding identity "arn:aws:iam::111122223333:user/my-user" to
 auth ConfigMap
```

5. Vea las asignaciones en el `ConfigMap` de nuevo.

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

Un ejemplo de salida sería el siguiente.

```
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
```

Edición de Configmap manualmente

1. Abra el icono `ConfigMap` para editar.

```
kubectl edit -n kube-system configmap/aws-auth
```

> **Note**
>
> Si recibe un error que indica "`Error from server (NotFound): configmaps "aws-auth" not found`", utilice el procedimiento de Aplicación de ConfigMap de aws-auth a un clúster para aplicar el `ConfigMap` estándar.

2. Agregue las entidades principales de IAM al `ConfigMap`. Un grupo de IAM no es una entidad principal de IAM, por lo que no se puede agregar al `ConfigMap`.
   - A fin de agregar un rol de IAM (por ejemplo, para usuarios federados): agregue los detalles del rol a la sección `mapRoles` del `ConfigMap`, en `data`. Agregue esta sección si no existe todavía en el archivo. Cada entrada admite los siguientes parámetros:
     - rolearn: ARN del rol de IAM que se va a agregar. Este valor no puede incluir una ruta. Por ejemplo, no puede especificar un ARN como `arn:aws:iam::111122223333:role/my-team/developers/role-name` . El ARN se debe `arn:aws:iam::111122223333:role/role-name` en su lugar.
     - username: nombre del usuario de Kubernetes al que se mapea el rol de IAM.
     - groups (grupos): el grupo o la lista de grupos de Kubernetes a los que asignar el rol. El grupo puede ser un grupo predeterminado o un grupo especificado en un `clusterrolebinding` o `rolebinding`. Para obtener más información, consulte Default roles and role bindings en la documentación de Kubernetes.
   - Para agregar un usuario de IAM: Según las prácticas recomendadas de IAM, se recomienda conceder permisos a los roles en lugar de a los usuarios. Agregue los detalles del usuario a la sección `mapUsers` del `ConfigMap`, en `data`. Agregue esta sección si no existe todavía en el archivo. Cada entrada admite los siguientes parámetros:
     - userarn: ARN del usuario de IAM que se va a agregar.
     - username: nombre de usuario dentro de Kubernetes al que se mapea el usuario de IAM.
     - groups (grupos): el grupo o la lista de grupos de Kubernetes a los que asignar el usuario. El grupo puede ser un grupo predeterminado o un grupo especificado en un `clusterrolebinding` o `rolebinding`. Para obtener más información, consulte Default roles and role bindings en la documentación de Kubernetes.
3. Por ejemplo, el siguiente bloque YAML contiene:
   - Una sección de `mapRoles` que asigna la instancia de nodos de IAM a grupos de Kubernetes para que los nodos puedan registrarse en el clúster y el rol de IAM `my-console-viewer-role` que se asigna a un grupo de Kubernetes que puede ver todos los recursos de Kubernetes

para todos los clústeres. Para obtener una lista de los permisos de grupo de IAM y Kubernetes necesarios para el rol de IAM de `my-console-viewer-role`, consulte the section called “Permisos necesarios”.

- Una sección de `mapUsers` que asigna el usuario de IAM `admin` desde la cuenta de valor predeterminado de AWS al grupo Kubernetes `system:masters` y el usuario `my-user` de otra cuenta AWS asignada a un grupo de Kubernetes que puede ver los recursos de Kubernetes para un espacio de nombres específico. Para obtener una lista de los permisos de grupo de IAM y Kubernetes necesarios para el usuario de IAM de `my-user`, consulte the section called “Permisos necesarios”.

  Agregue o quite líneas según sea necesario y reemplace todos los valores de ejemplo con sus propios valores.

```
# 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. Guarde el archivo y salga del editor de texto.

Aplique el **ConfigMap** de **aws-auth** en su clúster

El ConfigMap de aws-auth se crea y aplica de forma automática al clúster cuando crea un grupo de nodos administrados o cuando crea un grupo de nodos mediante eksctl. En un principio, se crea para permitir que los nodos se unan al clúster, pero también se utiliza este ConfigMap para agregar acceso de control de acceso basado en roles (RBAC) a las entidades principales de IAM. Si ha lanzado nodos autoadministrados y no ha aplicado el ConfigMap de aws-auth al clúster, puede hacerlo con el siguiente procedimiento.

1. Verifique si ya ha aplicado el ConfigMap de aws-auth.

   ```
   kubectl describe configmap -n kube-system aws-auth
   ```

   Si recibe un error con estado "Error from server (NotFound): configmaps "aws-auth" not found", continúe con los siguientes pasos para aplicar el ConfigMap estándar.

2. Descargue, edite y aplique el mapa de configuración del autenticador de AWS.

   a. Descargue el mapa de configuración.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/aws-auth-cm.yaml
   ```

   b. En el archivo aws-auth-cm.yaml, establezca el rolearn para el Nombre de recurso de Amazon (ARN) o el rol de IAM asociado con sus nodos. Puede hacerlo con un editor de texto o puede reemplazar *my-node-instance-role* y ejecutar el siguiente comando:

   ```
   sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-instance-role|' aws-auth-cm.yaml
   ```

   No modifique ninguna otra línea de este archivo.

   > **Important**
   >
   > El ARN de rol no puede incluir una ruta como role/my-team/developers/my-role. El formato del ARN debe ser arn:aws:iam::*111122223333*:role/*my-role* . En este ejemplo, se debe eliminar my-team/developers/.

Puede inspeccionar las salidas de la pila de AWS CloudFormation para los grupos de nodos y buscar los siguientes valores:

- InstanceRoleARN: para grupos de nodos que se crearon con `eksctl`
- NodeInstanceRole: para grupos de nodos que se crearon con plantillas de Amazon EKS incluidas en AWS CloudFormation en la Consola de administración de AWS

c. Aplique la configuración. Este comando puede tardar varios minutos en finalizar.

```
kubectl apply -f aws-auth-cm.yaml
```

> **Note**
>
> Si recibe cualquier error de tipo de recurso o autorización, consulte the section called "Acceso denegado o no autorizado (`kubectl`)" en el tema de solución de problemas.

3. Observe el estado de los nodos y espere a que aparezca el estado `Ready`.

```
kubectl get nodes --watch
```

Ingrese `Ctrl+C` para obtener un símbolo del intérprete de comandos.

Concesión de acceso a Kubernetes con un proveedor de OIDC externo para los usuarios

Amazon EKS admite la utilización de proveedores de identidad OpenID Connect (OIDC) como método para autenticar usuarios en su clúster. Los proveedores de identidad de OIDC se pueden utilizar con AWS Identity and Access Management (IAM) o como una alternativa de IAM. Para obtener más información acerca del uso de IAM, consulte the section called "Acceso a la API de Kubernetes". Después de configurar la autenticación de su clúster, puede crear`roles` y `clusterroles` de Kubernetes para asignar permisos a los roles y, a continuación, vincular los roles a las identidades con `rolebindings` y `clusterrolebindings` de Kubernetes. Para obtener más información, consulte Utilización de la autorización de RBAC en la documentación de Kubernetes.

- Puede asociar un proveedor de identidad de OIDC al clúster.

- Kubernetes no proporciona un proveedor de identidad de OIDC. Puede utilizar un proveedor de identidad de OIDC público existente o puede ejecutar su propio proveedor de identidad. Para obtener una lista de proveedores certificados, consulte Certificación de OpenID en la página web de OpenID.
- La URL del emisor del proveedor de identidad de OIDC debe ser accesible de manera pública para que Amazon EKS pueda descubrir las claves de firma. Amazon EKS no admite proveedores de identidad de OIDC con certificados autofirmados.
- No es posible desactivar el autenticador de IAM en el clúster, ya que aún es necesario para unir los nodos al clúster.
- Un clúster de Amazon EKS aún debe crearse mediante una entidad principal de AWS IAM, en lugar de un usuario de proveedor de identidad de OIDC. Esto se debe a que el creador del clúster interactúa con las API de Amazon EKS, en lugar de con las API de Kubernetes.
- Los usuarios autenticados por el proveedor de identidad de OIDC aparecen en el registro de auditoría del clúster si los Registros de CloudWatch están activados para el plano de control. Para obtener más información, consulte the section called “Habilitación o deshabilitación de los registros del plano de control”.
- No puede iniciar sesión en la Consola de administración de AWS con una cuenta de un proveedor de OIDC. Solo puede the section called “Acceso a recursos del clúster” iniciando sesión en la Consola de administración de AWS con una cuenta de AWS Identity and Access Management (IAM).

Asociar un proveedor de identidad de OIDC

Antes de asociar un proveedor de identidad de OIDC al clúster, necesita la siguiente información de su proveedor:

URL del emisor

La URL del proveedor de identidad de OIDC que permite al servidor de API descubrir claves de firma públicas para verificar tokens. La URL debe comenzar con `https://` y debe corresponder a la afirmación `iss` en los tokens de ID de OIDC del proveedor. De acuerdo con el estándar de OIDC, los componentes de ruta están permitidos, pero los parámetros de consulta no lo están. Normalmente, la URL consta de solo un nombre de host, como `https://server.example.org` o `https://example.com`. Esta URL debe apuntar al siguiente nivel a `.well-known/openid-configuration` y debe ser de acceso público a través de Internet.

ID de cliente (también conocido como público)

El ID de la aplicación cliente que realiza solicitudes de autenticación al proveedor de identidad de OIDC.

Puede asociar un proveedor de identidad mediante `eksctl` o la Consola de administración de AWS.

Asociación de un proveedor de identidad mediante eksctl

1. Cree un archivo denominado `associate-identity-provider.yaml` con el siguiente contenido. Sustituya los valores de ejemplo por sus propios valores. Los valores de la sección `identityProviders` se obtienen de su proveedor de identidad de OIDC. Los valores solo son necesarios para la configuración de `name`, `type`, `issuerUrl` y `clientId` en `identityProviders`.

```
---
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**
>
> No especifique `system:` o cualquier parte de esa cadena para `groupsPrefix` o `usernamePrefix`.

2. Cree el proveedor.

```
eksctl associate identityprovider -f associate-identity-provider.yaml
```

3. Para utilizar `kubectl` a fin de que trabaje con su clúster y proveedor de identidad de OIDC, consulte Using kubectl en la documentación de Kubernetes.

Puede asociar un proveedor de identidad mediante la consola de AWS

1. Abra la consola de Amazon EKS.
2. Seleccione el clúster y, a continuación, seleccione la pestaña Acceso.
3. En la sección Proveedores de identidad OIDC, seleccione *Asociar proveedor de identidad*.
4. En la página Associate OIDC Identity Provider (Asociar proveedor de identidad de OIDC), ingrese o seleccione las siguientes opciones y, a continuación, seleccione Associate (Asociar).
   - En Name (Nombre), ingrese un nombre único para el proveedor.
   - En Issuer URL (URL del emisor), ingrese la URL del proveedor. Debe ser posible el acceso a esta URL a través de Internet.
   - En ID de cliente, ingrese el ID de cliente del proveedor de identidad de OIDC (también conocido como audiencia).
   - En Username claim (Afirmación de nombre de usuario), ingrese la afirmación que desea utilizar como nombre de usuario.
   - En Afirmación de grupo, ingrese la afirmación que desea utilizar como grupo del usuario.
   - (Opcional) Seleccione Opciones avanzadas, e ingrese o seleccione la siguiente información.
     - Username prefix (Prefijo de nombre de usuario): ingrese un prefijo para anteponer a las afirmaciones de nombre de usuario. El prefijo se antepone a las afirmaciones de nombre de usuario para evitar conflictos con nombres existentes. Si no proporciona un valor y el nombre de usuario es un valor distinto de `email`, el prefijo se establece de forma predeterminada en el valor de Issuer URL (URL del emisor). Puede utilizar el valor `-` para desactivar todos los prefijos. No especifique `system:` o cualquier parte de esa cadena.

- Groups prefix (Prefijo de grupos): ingrese un prefijo para anteponer a las afirmaciones de grupos. El prefijo se antepone a las afirmaciones de grupo para evitar conflictos con nombres existentes (por ejemplo, `system: groups`). Por ejemplo, el valor `oidc:` crea nombres de grupo como `oidc:engineering` y `oidc:infra`. No especifique `system:` o cualquier parte de esa cadena.
- Required claims (Afirmaciones requeridas): seleccione Add claim (Agregar afirmación) e ingrese uno o más pares de valor de clave que describan las afirmaciones requeridas en el token del ID de cliente. Los pares describen las contestaciones requeridas en el token del ID. Si se establece, se verifica que cada afirmación esté presente en el token del ID con un valor coincidente.

  a. Para utilizar `kubectl` a fin de que trabaje con su clúster y proveedor de identidad de OIDC, consulte Using kubectl en la documentación de Kubernetes.

Política de IAM de ejemplo

Si desea evitar que un proveedor de identidad de OIDC se asocie a un clúster, cree y asocie la siguiente política de IAM a las cuentas de IAM de sus administradores de Amazon EKS. Para obtener más información, consulte Cómo crear políticas de IAM y Cómo agregar permisos de identidades de IAM en la Guía del usuario de IAM y Acciones en la Referencia de autorización de servicios.

```
{
    "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": "*"
        }
    ]
}
```

La siguiente política de ejemplo permite la asociación de proveedores de identidad de OIDC si el `clientID` es `kubernetes` y la `issuerUrl` es https://cognito-idp.us-west-2amazonaws.com/* .

```
{
    "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": "*"
        }
    ]
}
```

Desasociar un proveedor de identidad de OIDC del clúster

Si desasocia un proveedor de identidad de OIDC del clúster, los usuarios incluidos en el proveedor ya no podrán acceder al clúster. Sin embargo, sigue teniendo acceso al clúster con las entidades principales de IAM.

1. Abra la consola de Amazon EKS.
2. En la sección OIDC Identity Providers (Proveedores de identidad de OIDC), seleccione Disassociate (Desasociar), ingrese el nombre del proveedor de identidad y, a continuación, seleccione `Disassociate`.

Visualización de los recursos de Kubernetes en la Consola de administración de AWS

Puede ver los recursos de Kubernetes implementados en su clúster con la Consola de administración de AWS. No puede ver los recursos de Kubernetes con la AWS CLI o eksctl. Para ver los recursos de Kubernetes mediante una herramienta de línea de comandos, utilice kubectl.

> **Note**
>
> Para ver la pestaña Recursos y la sección Nodos de la pestaña Computación de la Consola de administración de AWS, la entidad principal de IAM que utilice debe tener permisos de IAM y Kubernetes específicos. Para obtener más información, consulte the section called "Permisos necesarios".

1. Abra la consola de Amazon EKS.
2. En la lista Clusters (Clústeres), seleccione el clúster que contiene los recursos de Kubernetes que desea ver.
3. Seleccione la pestaña Recursos.
4. Seleccione un grupo de Tipo de recurso del que desea ver los recursos, como Cargas de trabajo. Aparece una lista de los tipos de recursos de ese grupo.
5. Seleccione un tipo de recurso, como Deployments (Implementaciones), en el grupo Cargas de trabajo. Puede ver una descripción del tipo de recurso, un enlace a la documentación de Kubernetes para obtener más información sobre el tipo de recurso y una lista de los recursos

de ese tipo que se implementan en el clúster. Si la lista está vacía, no hay recursos de ese tipo implementado en el clúster.

6. Seleccione un recurso para ver más información acerca de una instantánea. Pruebe los siguientes ejemplos:
   - Seleccione el grupo Cargas de trabajo, seleccione el tipo de recurso de Implementaciones y seleccione el recurso coredns. Al seleccionar un recurso, se encuentra en Vista estructurada, de forma predeterminada. Para algunos tipos de recursos, verá una sección de Pods en Vista estructurada. En esta sección se muestran los pods administrados por la carga de trabajo. Puede seleccionar cualquier pod de la lista para ver información acerca del pod. No todos los tipos de recursos muestran información en Structured View (Vista estructurada). Si selecciona Raw view (Vista sin procesar) en la esquina superior derecha de la página del recurso, verá la respuesta JSON completa de la API de Kubernetes para el recurso.
   - Seleccione el grupo Cluster (Clústeres) y, a continuación, seleccione el tipo de recursos Nodes (Nodos). Aparece una lista de todos los nodos del clúster. Los nodos pueden ser cualquier tipo de nodo de Amazon EKS. Esta es la misma lista que ve en la sección Nodos al seleccionar la pestaña Informática de su clúster. Seleccione un recurso de nodo de la lista. En Vista estructurada, también ve una sección de Pods. En esta sección se muestran todos los pods que se ejecutan en el nodo.

Permisos necesarios

Para ver la pestaña Recursos y la sección Nodos de la pestaña Computación de la Consola de administración de AWS, la entidad principal de IAM que utilice debe tener permisos mínimos de IAM y Kubernetes específicos. Complete los siguientes pasos para asignar los permisos necesarios a las entidades principales de IAM.

1. Asegúrese de que la `eks:AccessKubernetesApi` y otros permisos de IAM necesarios para ver los recursos de Kubernetes estén asignados a la entidad principal de IAM que esté utilizando. Para obtener más información acerca de cómo editar los permisos para una entidad principal de IAM, consulte Control del acceso para las entidades principales de IAM en la Guía del usuario de IAM. Para obtener más información acerca de cómo editar los permisos de un rol, consulte Modificación de una política de permisos de rol (consola) en la Guía del usuario de IAM.

   En la siguiente política de ejemplo se incluyen los permisos necesarios para que una entidad principal vea los recursos de Kubernetes de todos los clústeres de su cuenta. Reemplace *111122223333* por su ID de cuenta de 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/*"
        }
    ]
}
```

Para ver los nodos de los clústeres conectados, el rol de IAM del conector de Amazon EKS debería poder suplantar a la entidad principal en el clúster. Esto permite que Amazon EKS Connector asigne la entidad principal a un usuario de Kubernetes.

2. Cree un `rolebinding` o `clusterrolebinding` de Kubernetes vinculado a un `role` o `clusterrole` de Kubernetes que tenga los permisos necesarios para ver los recursos de Kubernetes. Para conocer más sobre roles y vinculaciones de roles de Kubernetes, consulte la Utilización de la autorización de RBAC en la documentación de Kubernetes. Puede aplicar uno de los siguientes manifiestos al clúster que crea un `role` y `rolebinding` o un `clusterrole` y `clusterrolebinding` con los permisos Kubernetes necesarios:

Visualización de recursos de Kubernetes en todos los espacios de nombres

- El nombre del grupo en el archivo es `eks-console-dashboard-full-access-group`. Aplique el manifiesto al clúster con el siguiente comando:

```
kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-full-access.yaml
```

Visualización de recursos de Kubernetes en un espacio de nombres específico

- El espacio de nombres de este archivo es `default`. El nombre del grupo en el archivo es `eks-console-dashboard-restricted-access-group`. Aplique el manifiesto al clúster con el comando siguiente:

```
kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-restricted-access.yaml
```

Si necesita cambiar el nombre del grupo de Kubernetes, el espacio de nombres, los permisos o cualquier otra configuración del archivo, descargue el archivo y edítelo antes de aplicarlo al clúster:

a. Descargue el archivo con uno de los siguientes comandos:

```
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. Edite el archivo según sea necesario.

c. Aplique el manifiesto al clúster con uno de los siguientes comandos:

```
kubectl apply -f eks-console-full-access.yaml
```

```
kubectl apply -f eks-console-restricted-access.yaml
```

3. Asigne la entidad principal de IAM al usuario de Kubernetes o al grupo en el `ConfigMap` de `aws-auth`. Puede utilizar una herramienta como `eksctl` para actualizar el `ConfigMap` o puede actualizarlo manualmente editándolo.

> **Important**
>
> Recomendamos utilizar `eksctl`, u otra herramienta, para editar el `ConfigMap`. Para obtener información acerca de otras herramientas que puede utilizar, consulte Utilice herramientas para realizar cambios en el aws-authConfigMap en las guías de prácticas recomendadas de Amazon EKS. Un formato incorrecto de `aws-auth` `ConfigMap` puede provocar que pierda el acceso a su clúster.

Edición con eksctl

1. Necesita tener la versión `0.215.0` o posterior de la herramienta de línea de comandos `eksctl` instalada en su dispositivo o AWS CloudShell. Para instalar o actualizar `eksctl`, consulte la sección de Instalación en la documentación de `eksctl`.
2. Vea las asignaciones actuales en la `ConfigMap`. Reemplace *`my-cluster`* por el nombre de su clúster. Reemplace *`region-code`* por la región de AWS en la que se encuentra el clúster.

   ```
   eksctl get iamidentitymapping --cluster my-cluster --region=region-code
   ```

   Un ejemplo de salida sería el siguiente.

   ```
   ARN
               USERNAME                                GROUPS
    ACCOUNT
   arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
   NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
    system:bootstrappers,system:nodes
   ```

3. Agregue una asignación para un rol. En este ejemplo se supone que ha adjuntado los permisos de IAM en el primer paso a un rol denominado *`my-console-viewer-role`*. Reemplace *`111122223333`* por el ID de su cuenta.

   ```
   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**
>
> El ARN del rol no puede incluir una ruta como `role/my-team/developers/my-role`. El formato del ARN debe ser `arn:aws:iam::111122223333:role/my-role` . En este ejemplo, se necesita eliminar `my-team/developers/`.

Un ejemplo de salida sería el siguiente.

```
[...]
2022-05-09 14:51:20 [#]  adding identity "arn:aws:iam::111122223333:role/my-console-
viewer-role" to auth ConfigMap
```

4. Agregue una asignación para un usuario. Según las prácticas recomendadas de IAM, se recomienda conceder permisos a los roles en lugar de a los usuarios. En este ejemplo se supone que ha adjuntado los permisos de IAM en el primer paso a un usuario denominado *my-user*. Reemplace *111122223333* por el ID de su cuenta.

```
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
```

Un ejemplo de salida sería el siguiente.

```
[...]
2022-05-09 14:53:48 [#]  adding identity "arn:aws:iam::111122223333:user/my-user" to
 auth ConfigMap
```

5. Vea las asignaciones en el `ConfigMap` de nuevo.

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

Un ejemplo de salida sería el siguiente.

```
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
```

Edite ConfigMap de forma manual

Para obtener más información sobre cómo agregar usuarios al `ConfigMap` de `aws-auth`, consulte the section called “Agregar las entidades principales de IAM al clúster de Amazon EKS”.

1. Abra el `ConfigMap` de `aws-auth` para editar.

```
kubectl edit -n kube-system configmap/aws-auth
```

2. Agregue las asignaciones a la `aws-auth` `ConfigMap`, pero no reemplace ninguna de las asignaciones existentes. En el siguiente ejemplo se agregan asignaciones entre entidades principales de IAM con permisos agregados en el primer paso y los grupos de Kubernetes creados en el paso anterior:
   - El rol *`my-console-viewer-role`* y el `eks-console-dashboard-full-access-group`.
   - El usuario *`my-user`* y el `eks-console-dashboard-restricted-access-group`.

     En estos ejemplos se supone que ha adjuntado los permisos de IAM en el primer paso a un rol denominado *`my-console-viewer-role`* y un usuario llamado *`my-user`*. Reemplace *`111122223333`* por su ID de cuenta de AWS.

```
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**
>
> El ARN del rol no puede incluir una ruta como `role/my-team/developers/my-console-viewer-role`. El formato del ARN debe ser `arn:aws:iam::111122223333:role/my-console-viewer-role` . En este ejemplo, se debe eliminar `my-team/developers/`.

3. Guarde el archivo y salga del editor de texto.

Conexión de kubectl a un clúster de EKS mediante la creación de un archivo kubeconfig

En este tema, creará un archivo `kubeconfig` para su clúster (o actualizará uno existente).

La herramienta de línea de comandos `kubectl` usa la información de configuración en los archivos `kubeconfig` para comunicarse con el servidor de API de un clúster. Para obtener más información, consulte Organizing Cluster Access Using kubeconfig Files en la documentación de Kubernetes.

Amazon EKS usa el comando `aws eks get-token` con `kubectl` para la autenticación del clúster. De forma predeterminada, la CLI de AWS utiliza las mismas credenciales que se devuelven con el siguiente comando:

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

- Un clúster existente de Amazon EKS. Para implementar uno, consulte Introducción.
- La herramienta de línea de comandos de `kubectl` está instalada en su dispositivo o AWS CloudShell. La versión puede ser la misma o hasta una versión secundaria anterior o posterior a la versión de Kubernetes de su clúster. Por ejemplo, si la versión del clúster es `1.29`, puede usar la versión `1.28`, `1.29` o `1.30` de `kubectl` con él. Para instalar o actualizar `kubectl`, consulte the section called “Configure kubectl y eksctl”.

- La versión `2.12.3` o posterior, o bien, la versión `1.27.160` o posterior de la AWS interfaz de la línea de comandos (AWS CLI) instalada y configurada en su dispositivo o AWS CloudShell. Para comprobar su versión actual, utilice `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente. Para actualizarla, consulte Instalación de la CLI de AWS en su directorio principal en la Guía del usuario de AWS CloudShell.
- Un rol o usuario de IAM con permisos para utilizar la acción de API `eks:DescribeCluster` en el clúster que especifique. Para obtener más información, consulte the section called “Políticas basadas en identidades”. Si utiliza una identidad de su propio proveedor de OpenID Connect para acceder al clúster, consulte Using kubectl en la documentación de Kubernetes para crear o actualizar el archivo de `kube config`.

Crear el archivo **`kubeconfig`** de forma automática

- La versión `2.12.3` o posterior, o bien, la versión `1.27.160` o posterior de la Interfaz de la línea de comandos de AWS (AWS CLI) instalada y configurada en su dispositivo o AWS CloudShell. Para comprobar su versión actual, utilice `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente. Para actualizarla, consulte Instalación de la CLI de AWS en su directorio principal en la Guía del usuario de AWS CloudShell.
- Permiso para usar la acción de API `eks:DescribeCluster` en el clúster que especifique. Para obtener más información, consulte the section called “Políticas basadas en identidades”.

1. Creación o actualización de un archivo de `kubeconfig` para el clúster. Reemplace *`region-code`* por la región de AWS donde creó el clúster y *`my-cluster`* por el nombre de su clúster.

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

De forma predeterminada, el archivo de configuración resultante se crea en la ruta de `kubeconfig` predeterminada (`.kube`) en el directorio de inicio o en combinación con un archivo `config` existente en dicha ubicación. Puede especificar otra ruta con la opción `--kubeconfig`.

Puede especificar un ARN de rol de IAM con la opción `--role-arn` para utilizar en la autenticación al emitir comandos `kubectl`. De lo contrario, se utilizará la entidad principal de IAM de la CLI predeterminada o la cadena de credencial del SDK de AWS. Puede ver su identidad de la CLI o el SDK predeterminados de AWS ejecutando el comando `aws sts get-caller-identity`.

Para ver todas las opciones disponibles, ejecute el comando `aws eks update-kubeconfig help` o consulte update-kubeconfig en la Referencia de los comandos de la CLI de AWS.

2. Pruebe la configuración.

```
kubectl get svc
```

Un ejemplo de salida sería el siguiente.

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

Si recibe cualquier error de tipo de recurso o autorización, consulte the section called “Acceso denegado o no autorizado (`kubectl`)” en el tema de solución de problemas.

Concesión de acceso a las cargas de trabajo de Kubernetes a AWS mediante las cuentas de servicio de Kubernetes

A Kubernetes service account provides an identity for processes that run in a Pod. For more information see Administración de cuentas de servicio 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 “Credenciales con IRSA” or the section called “Pod Identity”.

Tokens de cuenta de servicio

La característica BoundServiceAccountTokenVolume está habilitada de forma predeterminada en las versiones de Kubernetes. Esta función mejora la seguridad de los tokens de cuenta de servicio al permitir que las cargas de trabajo se ejecuten en Kubernetes para solicitar tokens web JSON que estén vinculados a la audiencia, la hora y la clave. Los tokens de cuenta de servicio tienen una caducidad de una hora. En versiones anteriores de Kubernetes, los tokens no tenían caducidad. Esto significa que los clientes que confían en estos tokens deben actualizar los tokens en una hora. Los siguientes ejemplos SDK de cliente de Kubernetes actualizan los tokens automáticamente dentro del plazo requerido:

- Versión de Go `0.15.7` y posteriores
- Versión de Python `12.0.0` y posteriores
- Versión de Java `9.0.0` y posterior
- Versión de JavaScript `0.10.3` y posterior
- Rama de Ruby `master`
- Versión de Haskell `0.3.0.0`
- Versión `7.0.5` y posteriores de C#

Si su carga de trabajo utiliza una versión de cliente anterior, debe actualizarla. Para permitir una migración fluida de los clientes a los tokens de cuenta de servicio con límite de tiempo más nuevos, Kubernetes agrega un periodo de vencimiento extendido al token de cuenta de servicio durante la hora predeterminada. Para los clústeres de Amazon EKS, el período de caducidad extendido es de 90 días. El servidor de API de Kubernetes de los clústeres de Amazon EKS rechaza solicitudes con tokens de más de 90 días de antigüedad. Le recomendamos que compruebe sus aplicaciones y sus dependencias para asegurarse de que los SDK de cliente de Kubernetes son iguales o posteriores a las versiones indicadas anteriormente.

Cuando el servidor API recibe solicitudes con tokens de más de una hora de antigüedad, anota el evento de registro de auditoría de la API con `annotations.authentication.k8s.io/stale-token`. El valor de la anotación es similar al siguiente ejemplo:

```
subject: system:serviceaccount:common:fluent-bit, seconds after warning threshold:
 4185802.
```

Si el clúster tiene un registro de plano de control habilitado, entonces las anotaciones se encuentran en los registros de auditoría. Puede utilizar la siguiente consulta de Información de registros de CloudWatch para identificar todos los pods del clúster de Amazon EKS que utilizan tokens obsoletos:

```
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
```

El `subject` hace referencia a la cuenta de servicio que utilizó el pod. El `elapsedtime` indica el tiempo transcurrido (en segundos) tras leer el último token. Las solicitudes al servidor API se denegan cuando el `elapsedtime` supera los 90 días (7 776 000 segundos). Debe actualizar de forma proactiva el SKD del cliente de Kubernetes de las aplicaciones para utilizar una de las versiones enumeradas anteriormente que actualiza automáticamente el token. Si el token de cuenta de servicio utilizado dura casi 90 días y no tiene tiempo suficiente para actualizar las versiones del SDK de cliente antes de que el token caduque, puede terminar los pods existentes y crear otros nuevos. Esto da como resultado la reactivación del token de la cuenta de servicio, lo que le da 90 días adicionales para actualizar los SDK de la versión de cliente.

Si el pod forma parte de una implementación, la forma sugerida de finalizar los pods y mantener la alta disponibilidad es efectuar un despliegue con el siguiente comando. Reemplace *`my-deployment`* con el nombre de la implementación.

```
kubectl rollout restart deployment/my-deployment
```

Complementos de clúster

Los siguientes complementos de clúster se han actualizado para utilizar los SDK del cliente de Kubernetes que reactivan automáticamente los tokens de cuentas de servicio. Recomendamos asegurarse de que las versiones enumeradas, o versiones posteriores, estén instaladas en su clúster.

- Versión `1.8.0` y posteriores de los complementos auxiliares de métricas y CNI de Amazon VPC para Kubernetes. Para comprobar su versión actual o actualizarla, consulte the section called “CNI de Amazon VPC” y cni-metrics-helper.
- Versión de CoreDNS `1.8.4` y posteriores. Para comprobar su versión actual o actualizarla, consulte the section called “CoreDNS”.

- Versión del Controlador del equilibrador de carga de AWS `2.0.0` y posteriores. Para comprobar su versión actual o actualizarla, consulte the section called " AWS Controlador del equilibrador de carga de ".
- Una versión actual de `kube-proxy`. Para comprobar su versión actual o actualizarla, consulte the section called "kube-proxy".
- AWS para Fluent Bit versión `2.25.0` o posterior. Para actualizar la versión actual, consulte Releases (Lanzamientos) en GitHub.
- Versión de imagen de Fluentd 1.14.6-1.2 o posterior y versión del complemento de filtro de Fluentd para metadatos de Kubernetes 2.11.1 o posterior.

Concesión de permisos a AWS Identity and Access Management a cargas de trabajo en clústeres de Amazon Elastic Kubernetes Service

Amazon EKS ofrece dos formas de conceder a AWS Identity and Access Management permisos a las cargas de trabajo que se ejecutan en los clústeres de Amazon EKS: los roles de IAM para cuentas de servicio y las identidades de Pod de EKS.

Roles de IAM para cuentas de servicio

Los roles de IAM para cuentas de servicio (IRSA) configuran las aplicaciones de Kubernetes que se ejecutan en AWS con permisos de IAM detallados para acceder a otros recursos de AWS, como los buckets de Amazon S3, las tablas de Amazon DynamoDB y más. Puede ejecutar varias aplicaciones juntas en el mismo clúster de Amazon EKS y asegurarse de que cada aplicación tenga solo el conjunto mínimo de permisos que necesita. IRSA se creó para admitir varias opciones de implementación de Kubernetes compatibles con AWS como Amazon EKS, Amazon EKS Anywhere, Red Hat OpenShift Service en AWS y clústeres de Kubernetes autoadministrados en instancias de Amazon EC2. Por lo tanto, IRSA se creó utilizando un servicio de AWS básico como IAM y no dependía directamente del servicio Amazon EKS ni de la API de EKS. Para obtener más información, consulte the section called "Credenciales con IRSA".

Pod Identities de EKS

Pod Identity de EKS ofrece a los administradores de clústeres un flujo de trabajo simplificado para autenticar las aplicaciones con el fin de acceder a otros recursos de AWS, como los buckets de Amazon S3, las tablas de Amazon DynamoDB y más. Pod Identity de EKS está dedicada solo para EKS y, como resultado, simplifica la forma en que los administradores de clústeres pueden configurar las aplicaciones de Kubernetes para obtener permisos de IAM. Estos

permisos ahora se pueden configurar fácilmente con menos pasos: directamente a través de Consola de administración de AWS, la API de EKS y la AWS CLI, no es necesario llevar a cabo ninguna acción dentro del clúster ni en ningún objeto de Kubernetes. Los administradores de clústeres no necesitan cambiar entre los servicios de EKS y de IAM, ni utilizar operaciones de IAM privilegiadas para configurar los permisos que requieren sus aplicaciones. Los roles de IAM ahora se pueden usar en varios clústeres sin necesidad de actualizar la política de confianza de roles al crear nuevos clústeres. Las credenciales de IAM proporcionadas por Pod Identity de EKS incluyen etiquetas de sesión de rol, con atributos como el nombre del clúster, el espacio de nombres y el nombre de la cuenta de servicio. Las etiquetas de sesión de rol permiten a los administradores crear un único rol que puede funcionar en todas las cuentas de servicio, ya que permiten el acceso a los recursos de AWS en función de las etiquetas coincidentes. Para obtener más información, consulte the section called “Pod Identity”.

Comparación de Pod Identity de EKS e IRSA

A un nivel alto, tanto Pod Identity de EKS como IRSA permiten conceder permisos de IAM a las aplicaciones que se ejecutan en clústeres de Kubernetes. Sin embargo, son fundamentalmente diferentes en cuanto a la forma de configurarlos, los límites admitidos y las características habilitadas. A continuación, comparamos algunos de los aspectos clave de ambas soluciones.

> **Note**
>
> AWS recomienda usar EKS Pod Identities para conceder acceso a los recursos de AWS de sus pods siempre que sea posible. Para obtener más información, consulte the section called “Pod Identity”.

Atributo	Pod Identity de EKS	IRSA
Extensibilidad de roles	Debe configurar cada rol una vez para establecer una relación de confianza con el recién introducido `pods.eks.amazonaws.com` de director de servicio de Amazon EKS. Tras este único paso, no necesitar	Debe actualizar la política de confianza del rol de IAM con el nuevo punto de conexión del proveedor de OIDC de clústeres de EKS cada vez que desee utilizar el rol en un clúster nuevo.

Atributo	Pod Identity de EKS	IRSA
	á actualizar la política de confianza del rol cada vez que lo utilice en un clúster nuevo.	
Escalabilidad de los clústeres	Pod Identity de EKS no requiere que los usuarios configuren un proveedor de IAM OIDC, por lo que este límite no se aplica.	Cada clúster de EKS tiene una URL de emisor de OpenID Connect (OIDC) asociada. Para usar IRSA, es necesario crear un proveedor de OpenID Connect único para cada clúster de EKS de IAM. IAM tiene un límite global predeterminado de 100 proveedores de OIDC para cada cuenta de AWS. Si planea tener más de 100 clústeres de EKS para cada cuenta de AWS con IRSA, alcanzará el límite de proveedores de OIDC de IAM.

Atributo	Pod Identity de EKS	IRSA
Escalabilidad de roles	Pod Identity de EKS no exige que los usuarios definan la relación de confianza entre el rol de IAM y la cuenta de servicio en la política de confianza, por lo que este límite no se aplica.	En IRSA, usted define la relación de confianza entre un rol de IAM y una cuenta de servicio en la política de confianza del rol. De forma predeterminada, el tamaño de la política de confianza es `2048`. Esto significa que normalmente se pueden definir 4 relaciones de confianza en una sola política de confianza. Si bien puede aumentar el límite de duración de la política de confianza, normalmente está limitado a un máximo de 8 relaciones de confianza dentro de una sola política de confianza.
Uso de cuotas de la API de STS	EKS Pod Identity simplific a la entrega de credencia les de AWS a sus pods y no requiere que su código realice llamadas directamente al AWS Security Token Service (STS). El servicio de EKS gestiona la asunción de roles y entrega las credenciales a las aplicaciones escritas con el SDK de AWS en sus pods sin que estos se comunique n con AWS STS ni utilicen la cuota de la API de STS.	En IRSA, las aplicaciones escritas con el SDK de AWS de sus pods utilizan tokens para llamar a la API de `AssumeRoleWithWebI dentity` de AWS Security Token Service (STS). Según la lógica del código del SDK de AWS, es posible que el código realice llamadas innecesarias a AWS STS y reciba errores de limitación.

Atributo	Pod Identity de EKS	IRSA
Reutilización de roles	Las credenciales temporales de AWS STS proporcionadas por Pod Identity de EKS incluyen etiquetas de sesión de rol, como el nombre del clúster, el espacio de nombres y el nombre de la cuenta de servicio. Las etiquetas de sesión de rol permiten a los administradores crear un único rol de IAM que se puede usar con varias cuentas de servicio, con diferentes permisos efectivos, ya que permiten el acceso a los recursos de AWS basados en las etiquetas adjuntas a ellas. Esto también se conoce como control de acceso basado en atributos (ABAC). Para obtener más información, consulte the section called “Concesión de acceso a los pods”.	No se admiten etiquetas de sesión de AWS STS. Puede reutilizar un rol entre clústeres, pero cada pod recibe todos los permisos del rol.
Entornos compatibles	Pod Identity de EKS solo está disponible en Amazon EKS.	Se puede usar IRSA como Amazon EKS, Amazon EKS Anywhere, Red Hat OpenShift Service en AWS y clústeres de Kubernetes autoadministrados en instancias de Amazon EC2.

Atributo	Pod Identity de EKS	IRSA
Versiones compatibles de EKS	Todas las versiones del clúster EKS compatibles. Para saber las versiones de la plataforma específicas, consulte the section called “Versiones del clúster de Pod Identity de EKS”.	Todas las versiones del clúster EKS compatibles.

Roles de IAM para cuentas de servicio

Las aplicaciones de los contenedores de un pod pueden usar un SDK de AWS o AWS CLI para llevar a cabo solicitudes de API a servicios de AWS mediante permisos de AWS Identity and Access Management (IAM). Las aplicaciones deben firmar sus solicitudes de API AWS con credenciales de AWS. Los roles de IAM para cuentas de servicio (IRSA) ofrecen la posibilidad de administrar las credenciales para las aplicaciones, de un modo similar a cómo los perfiles de instancia de Amazon EC2 proporcionan credenciales a instancias de Amazon EC2. En lugar de crear y distribuir las credenciales de AWS a los contenedores o de utilizar el rol de la instancia de Amazon EC2, puede asociar el rol de IAM con una cuenta de servicio de Kubernetes y configurar los pods para usar la cuenta de servicio. No puede usar roles de IAM para cuentas de servicio con clústeres locales para Amazon EKS en AWS Outposts.

Los roles de IAM para cuentas de servicio ofrecen los siguientes beneficios:

- Privilegio mínimo: puede limitar los permisos de IAM a una cuenta de servicio y solo los pods que utilizan esa cuenta de servicio tienen acceso a esos permisos. Esta característica también elimina la necesidad de soluciones de terceros como `kiam` o `kube2iam`.
- Aislamiento de credenciales: Cuando se restringe el acceso al Servicio de metadatos de instancias (IMDS) de Amazon EC2, los contenedores de un pod solo pueden recuperar las credenciales para el rol de IAM asociado a la cuenta de servicio que usa el contenedor. Un contenedor nunca tiene acceso a credenciales que utilizan otros contenedores de otros pods. Si no se restringe el IMDS, los contenedores del pod también tienen acceso al rol de IAM del nodo de Amazon EKS y es posible que los contenedores puedan acceder a las credenciales de los roles de IAM de otros pods del mismo nodo. Para obtener más información, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.

> **Note**
>
> Los pods configurados con `hostNetwork: true` siempre tendrán acceso al IMDS, pero los SDK y la CLI de AWS utilizarán credenciales IRSA cuando estén habilitados.

- Auditabilidad: el acceso y el registro de eventos se encuentra disponible a través de AWS CloudTrail para garantizar una auditoría retrospectiva.

> **Important**
>
> Los contenedores no son un límite de seguridad, y el uso de roles de IAM para las cuentas de servicio no cambia esta situación. Los pods asignados al mismo nodo compartirán un kernel y posiblemente otros recursos, según la configuración del pod. Aunque los pods que se ejecutan en nodos separados estarán aislados en la capa de procesamiento, hay aplicaciones de nodos que tienen permisos adicionales en la API de Kubernetes más allá del ámbito de una instancia individual. Algunos ejemplos son `kubelet`, `kube-proxy`, controladores de almacenamiento CSI o sus propias aplicaciones de Kubernetes.

Siga estos procedimientos para habilitar los roles de IAM para cuentas de servicio:

1. Cree un proveedor de OIDC de IAM para el clúster: solo debe completar este procedimiento una vez para cada clúster.

   > **Note**
   >
   > Si habilitó el punto de conexión de VPC de EKS, no se podrá acceder al punto de conexión del servicio OIDC de EKS desde dentro de esa VPC. Por lo tanto, no funcionarán operaciones tales como crear un proveedor de OIDC con `eksctl` en la VPC, y provocarán que se agote el tiempo de espera al intentar solicitar `https://oidc.eks.region.amazonaws.com`. A continuación se muestra un ejemplo de mensaje de error:
   >
   > ```
   > server cant find oidc.eks.region.amazonaws.com: NXDOMAIN
   > ```

Para completar este paso, puede ejecutar el comando fuera de la VPC; por ejemplo, en AWS CloudShell o en un equipo conectado a Internet. Como alternativa, puede crear un solucionador condicional de horizonte dividido en la VPC, como Route 53 Resolver, para usar un solucionador diferente para la URL del emisor de OIDC y no usar el DNS de la VPC para ello. Para ver un ejemplo de reenvío condicional en CoreDNS, consulte Amazon EKS feature request en GitHub.

2. Asigne roles de IAM a cuentas de servicio de Kubernetes: complete este procedimiento para cada conjunto único de permisos que desee que una aplicación tenga.
3. Configure los pods para que usen una cuenta de servicio de Kubernetes: complete este procedimiento para cada pod que necesite acceder a los servicios de AWS.
4. Utilice ISRA con el AWS SDK: confirme que la carga de trabajo utilice un AWS SDK de una versión compatible y que utilice la cadena de credenciales predeterminada.

Información general de IAM, Kubernetes y OpenID Connect (OIDC)

En 2014, AWS Identity and Access Management agregó compatibilidad con identidades federadas mediante OpenID Connect (OIDC). Esta característica le permite autenticar llamadas a la API AWS con proveedores de identidad compatibles y recibir un token web JSON (JWT) de OIDC válido. Puede transferir este token a la operación API de AWS STS `AssumeRoleWithWebIdentity` y recibir credenciales temporales del rol de IAM. Puede utilizar estas credenciales para interactuar con cualquier servicio de AWS, como Amazon S3 y DynamoDB.

Cada token JWT está firmado por un par de claves de firma. Las claves se envían al proveedor de OIDC administrado por Amazon EKS y la clave privada cambia cada 7 días. Amazon EKS conserva las claves públicas hasta que caduquen. Si conecta clientes OIDC externos, tenga en cuenta que debe actualizar las claves de firma antes de que caduque la clave pública. Aprenda a obtener las claves de firma para validar los tokens de OIDC.

Kubernetes tiene cuentas de servicio de uso largo como su propio sistema de identidad interno. Los pods pueden autenticarse con el servidor de la API de Kubernetes mediante un token montado automáticamente (que era un JWT no OIDC) que solo el servidor de la API de Kubernetes podía validar. Estos tokens de cuenta de servicio heredados no caducan, y rotar la clave de firma es un proceso difícil. En la versión `1.12` de Kubernetes, se agregó compatibilidad para una nueva característica de `ProjectedServiceAccountToken`. Esta característica es un token web JSON

de OIDC que también contiene la identidad de la cuenta de servicio y permite una audiencia configurable.

Amazon EKS aloja un punto de conexión de detección de OIDC público por clúster que contiene las claves de firma para los tokens web JSON `ProjectedServiceAccountToken` a fin de que los sistemas externos como IAM puedan validar y aceptar los tokens de OIDC que emite Kubernetes.

Crear un proveedor de OIDC de IAM para su clúster

Su clúster tiene una URL de emisor de OpenID Connect (OIDC) asociada. Para utilizar roles de AWS Identity and Access Management (IAM) para cuentas de servicio, debe existir un proveedor de OIDC de IAM para la URL del emisor de OIDC de su clúster.

Requisitos previos

- Un clúster existente de Amazon EKS. Para implementar uno, consulte Introducción.
- La versión `2.12.3` o posterior, o bien, la versión `1.27.160` o posterior de la AWS interfaz de la línea de comandos (AWS CLI) instalada y configurada en su dispositivo o AWS CloudShell. Para comprobar su versión actual, utilice `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente. Para actualizarla, consulte Instalación de la CLI de AWS en su directorio principal en la Guía del usuario de AWS CloudShell.
- La herramienta de línea de comandos de `kubectl` está instalada en su dispositivo o AWS CloudShell. La versión puede ser la misma o hasta una versión secundaria anterior o posterior a la versión de Kubernetes de su clúster. Por ejemplo, si la versión del clúster es `1.29`, puede usar la versión `1.28`, `1.29` o `1.30` de `kubectl` con él. Para instalar o actualizar `kubectl`, consulte the section called “Configure kubectl y eksctl”.
- Un archivo `config` de `kubectl` existente que contenga la configuración del clúster. Para crear un archivo `config` de `kubectl`, consulte the section called “Acceso al clúster con kubectl”.

Puede crear un proveedor de OIDC de IAM para el clúster mediante `eksctl` o la Consola de administración de AWS.

Creación de un proveedor de OIDC (eksctl)

1. La versión `0.215.0` o posterior de la herramienta de línea de comandos de `eksctl` instalada en su dispositivo o AWS CloudShell. Para instalar o actualizar `eksctl`, consulte la sección de Instalación en la documentación de `eksctl`.
2. Determine el ID de emisor de OIDC correspondiente a su clúster.

   Recupere el ID de emisor de OIDC de su clúster y almacénelo en una variable. Reemplace `<my-cluster>` por su propio valor.

   ```
   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. Determine si ya hay un proveedor de OIDC de IAM con el ID del proveedor de su clúster en su cuenta.

   ```
   aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
   ```

   Si el comando anterior devuelve la salida, significa que ya tiene un proveedor de OIDC de IAM para su clúster y puede ir al siguiente paso. Si no se devuelve ninguna salida, debe crear un proveedor de OIDC de IAM para el clúster.
4. Cree un proveedor de identidad de OIDC de IAM para su clúster con el siguiente comando.

   ```
   eksctl utils associate-iam-oidc-provider --cluster $cluster_name --approve
   ```

   > **Note**
   >
   > Si habilitó el punto de conexión de VPC de EKS, no se podrá acceder al punto de conexión del servicio OIDC de EKS desde dentro de esa VPC. Por lo tanto, operaciones como la creación de un proveedor de OIDC con `eksctl` en la VPC no se ejecutarán correctamente y generarán un agotamiento del tiempo de espera. A continuación se muestra un ejemplo de mensaje de error:
   >
   > ```
   > ** server cant find oidc.eks.<region-code>.amazonaws.com: NXDOMAIN
   > ```

Para completar este paso, puede ejecutar el comando fuera de la VPC; por ejemplo, en AWS CloudShell o en un equipo conectado a Internet. Como alternativa, puede crear un solucionador condicional de horizonte dividido en la VPC, como Route 53 Resolver, para usar un solucionador diferente para la URL del emisor de OIDC y no usar el DNS de la VPC para ello. Para ver un ejemplo de reenvío condicional en CoreDNS, consulte Amazon EKS feature request en GitHub.

Creación de un proveedor de OIDC (Consola de AWS)

1. Abra la consola de Amazon EKS.
2. En el panel de la izquierda, seleccione Clústeres y, a continuación, seleccione el nombre de su clúster en la página Clusters (Clústeres).
3. En la sección de Details (Detalles) en la pestaña Overview (Resumen), anote el valor de la OpenID Connect provider URL (URL del proveedor de OpenID Connect).
4. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
5. En el panel de navegación, elija Identity Providers (Proveedores de identidad) en Access management (Administración de acceso). Si aparece un Proveedor que coincide con la URL de su clúster, ya cuenta con un proveedor para su clúster. Si no aparece un proveedor en la lista que coincida con la URL del clúster, debe crear uno.
6. Para crear un proveedor, elija Add Provider (Agregar proveedor).
7. En Tipo de proveedor, seleccione OpenID Connect.
8. En URL de proveedor, ingrese la URL del proveedor de OIDC correspondiente al clúster.
9. En Audiencia, ingrese `sts.amazonaws.com`.
10. (Opcional) Agregue cualquier etiqueta, por ejemplo, una para identificar qué clúster corresponde a este proveedor.
11. Elija Agregar proveedor.

Siguiente paso: the section called “Asignación del rol de IAM”

Asignación de roles de IAM a cuentas de servicio de Kubernetes

En este tema, se explica cómo configurar una cuenta de servicio de Kubernetes para asumir un rol de AWS Identity and Access Management (IAM). Los pods que estén configurados para usar la cuenta de servicio pueden acceder a cualquier servicio de AWS al que el rol tenga permisos para acceder.

Requisitos previos

- Un clúster existente. Si no tiene uno, puede crearlo mediante una de las siguientes guías de Introducción.
- Cree un proveedor de OpenID Connect (OIDC) de IAM para su clúster. Para saber si ya tiene un proveedor o cómo crear uno, consulte the section called “Proveedor de OIDC de IAM”.
- La versión `2.12.3` o posterior, o bien, la versión `1.27.160` o posterior de la AWS interfaz de la línea de comandos (AWS CLI) instalada y configurada en su dispositivo o AWS CloudShell. Para comprobar su versión actual, utilice `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente. Para actualizarla, consulte Instalación de la CLI de AWS en su directorio principal en la Guía del usuario de AWS CloudShell.
- La herramienta de línea de comandos de `kubectl` está instalada en su dispositivo o AWS CloudShell. La versión puede ser la misma o hasta una versión secundaria anterior o posterior a la versión de Kubernetes de su clúster. Por ejemplo, si la versión del clúster es `1.29`, puede usar la versión `1.28`, `1.29` o `1.30` de `kubectl` con él. Para instalar o actualizar `kubectl`, consulte the section called “Configure kubectl y eksctl”.
- Un archivo `config` de `kubectl` existente que contenga la configuración del clúster. Para crear un archivo `config` de `kubectl`, consulte the section called “Acceso al clúster con kubectl”.

Paso 1: creación de una política de IAM

Si desea asociar una política de IAM existente a su rol de IAM, vaya al siguiente paso.

1. Cree una política de IAM. Puede crear su propia política o copiar una política administrada de AWS que ya conceda algunos de los permisos que necesita y personalizarla según sus requisitos específicos. Para obtener más información, consulte Creación de políticas de IAM en la Guía del usuario de IAM.
2. Cree un archivo que incluya los permisos para los servicios de AWS a los que quiere que accedan sus pods. Para obtener una lista de todas las acciones para todos los servicios de AWS, consulte la Referencia de autorizaciones de servicio.

Puede ejecutar el siguiente comando para crear un archivo de política de ejemplo que permita el acceso de solo lectura a un bucket de Amazon S3. Opcionalmente, puede almacenar información de configuración o un script de arranque en este bucket, y los contenedores de su pod pueden leer el archivo desde el bucket y cargarlo en su aplicación. Si desea crear esta política de ejemplo, copie el siguiente contenido en su dispositivo. Sustituya *my-pod-secrets-bucket* por el nombre de su bucket y ejecute el comando.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
```

3. Creación de la política de IAM.

```
aws iam create-policy --policy-name my-policy --policy-document file://my-policy.json
```

Paso 2: creación y asociación de un rol de IAM

Cree un rol de IAM y asócielo a una cuenta de servicio de Kubernetes. Puede utilizar `eksctl` o la AWS CLI.

Crear y asociar un rol (eksctl)

Este comando `eksctl` crea una cuenta de servicio de Kubernetes en el espacio de nombres especificado, crea un rol de IAM (si no existe) con el nombre especificado, adjunta un ARN de política de IAM existente al rol y anota la cuenta de servicio con el ARN del rol de IAM. Asegúrese de reemplazar los valores de los marcadores de posición de muestra en este comando con sus valores específicos. Para instalar o actualizar `eksctl`, consulte la sección de Instalación en la documentación de `eksctl`.

```
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**
>
> Si el rol o la cuenta de servicio ya existen, podría producirse un error al ejecutar el comando anterior. `eksctl` tiene diferentes opciones que puede proporcionar en esas situaciones. Para obtener más información, ejecute `eksctl create iamserviceaccount --help`.

Crear y asociar un rol (AWS CLI)

Si tiene una cuenta de servicio de Kubernetes existente que desea que asuma un rol de IAM, puede omitir este paso.

1. Cree una cuenta de servicio de Kubernetes. Copie los siguientes contenidos en su dispositivo. Reemplace *`my-service-account`* por el nombre que desee y *`default`* por un espacio de nombres diferente, si es necesario. Si cambia *`default`*, el espacio de nombres debe existir previamente.

   ```
   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. Establezca su ID de cuenta de AWS en una variable de entorno con el siguiente comando.

   ```
   account_id=$(aws sts get-caller-identity --query "Account" --output text)
   ```

3. Establezca el proveedor de identidades de OIDC en una variable de entorno con el siguiente comando. Reemplace *`my-cluster`* por el nombre de su clúster.

   ```
   oidc_provider=$(aws eks describe-cluster --name my-cluster --region $AWS_REGION --
   query "cluster.identity.oidc.issuer" --output text | sed -e "s/^https:\/\///")
   ```

4. Establezca variables para el espacio de nombres y el nombre de la cuenta de servicio. Reemplace *`my-service-account`* por la cuenta de servicio de Kubernetes que desea que asuma el rol. Reemplace *`default`* por el espacio de nombres de la cuenta de servicio.

```
export namespace=default
export service_account=my-service-account
```

5. Ejecute el siguiente comando para crear un archivo de política de confianza para el rol de IAM. Si quiere permitir que todas las cuentas de servicio de un espacio de nombres utilicen el rol, copie el siguiente contenido en su dispositivo. Reemplace *StringEquals* por `StringLike` y reemplace *$service_account* por `*`. Puede agregar varias entradas en las condiciones `StringEquals` y `StringLike` para permitir que varias cuentas de servicio o espacios de nombres asuman el rol. Para permitir que los roles de una cuenta de AWS diferente a la de su clúster asuman el rol, consulte the section called “IAM entre cuentas” para obtener más información.

```
{
  "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. Creación del rol. Reemplace *my-role* con un nombre para el rol de IAM, y *my-role-description* con una descripción de su rol.

```
aws iam create-role --role-name my-role --assume-role-policy-document file://trust-
relationship.json --description "my-role-description"
```

7. Adjunte la política de IAM al rol. Reemplace *my-role* por el nombre de su rol de IAM y *my-policy* por el nombre de una política existente que haya creado.

```
aws iam attach-role-policy --role-name my-role --policy-arn=arn:aws:iam::
$account_id:policy/my-policy
```

8. Anote su cuenta de servicio con el nombre de recurso de Amazon (ARN) del rol de IAM que desea que asuma la cuenta de servicio. Reemplace *my-role* por el nombre de su rol de IAM existente. Supongamos que permitió que un rol de una cuenta de AWS diferente a la de su clúster asumiera el rol en un paso anterior. A continuación, asegúrese de especificar la cuenta de AWS y el rol de la otra cuenta. Para obtener más información, consulte the section called “IAM entre cuentas”.

```
kubectl annotate serviceaccount -n $namespace $service_account eks.amazonaws.com/
role-arn=arn:aws:iam::$account_id:role/my-role
```

9. (Opcional) Configure el punto de conexión de AWS Security Token Service para una cuenta de servicio. AWS recomienda el uso de un punto de conexión regional de AWS STS en lugar del punto de conexión global. Esto reduce la latencia, proporciona redundancia integrada y aumenta la validez de los tokens de sesión.

Paso 3: confirmación de la configuración

1. Confirme que la política de confianza del rol de IAM se haya configurado correctamente.

```
aws iam get-role --role-name my-role --query Role.AssumeRolePolicyDocument
```

Un ejemplo de salida sería el siguiente.

```
{
    "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. Confirme que la política que adjuntó a su rol en un paso anterior se encuentre adjunta al rol.

```
aws iam list-attached-role-policies --role-name my-role --query
 "AttachedPolicies[].PolicyArn" --output text
```

   Un ejemplo de salida sería el siguiente.

```
arn:aws:iam::111122223333:policy/my-policy
```

3. Establezca una variable para almacenar el nombre de recurso de Amazon (ARN) de la política que quiera utilizar. Reemplace *my-policy* por el nombre de la política para la que desea confirmar los permisos.

```
export policy_arn=arn:aws:iam::111122223333:policy/my-policy
```

4. Vea la versión predeterminada de la política.

```
aws iam get-policy --policy-arn $policy_arn
```

   Un ejemplo de salida sería el siguiente.

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "EXAMPLEBIOWGLDEXAMPLE",
        "Arn": "arn:aws:iam::111122223333:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        [...]
```

```
    }
}
```

5. Vea el contenido de la política para asegurarse de que incluye todos los permisos que su pod necesita. Si es necesario, reemplace *1* en el siguiente comando por la versión devuelta en la salida anterior.

```
aws iam get-policy-version --policy-arn $policy_arn --version-id v1
```

Un ejemplo de salida sería el siguiente.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
```

Si creó la política de ejemplo en un paso anterior, el resultado es el mismo. Si creó una política diferente, el contenido de *example* es diferente.

6. Confirme que la cuenta de servicio de Kubernetes se anota con el rol.

```
kubectl describe serviceaccount my-service-account -n default
```

Un ejemplo de salida sería el siguiente.

```
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
[...]
```

Siguientes pasos

- the section called “Asignación a un pod”

Configuración de pods para usar una cuenta de servicio de Kubernetes

Si un pod necesita acceder a los servicios de AWS, debe configurarlo para que use una cuenta de servicio de Kubernetes. La cuenta de servicio debe estar asociada a un rol de AWS Identity and Access Management (IAM) que tenga permisos para acceder a los servicios de AWS.

- Un clúster existente. Si no tiene uno, puede crearlo mediante una de las guías de Introducción.
- Cree un proveedor de OpenID Connect (OIDC) de IAM para su clúster. Para saber si ya tiene un proveedor o cómo crear uno, consulte the section called “Proveedor de OIDC de IAM”.
- Una cuenta de servicio de Kubernetes asociada a un rol de IAM. La cuenta de servicio debe estar anotada con el nombre de recurso de Amazon (ARN) del rol de IAM. El rol debe tener una política de IAM asociada que contenga los permisos que desee que tengan sus pods para usar servicios de AWS. Para obtener más información acerca de cómo crear y configurar la cuenta de servicio y el rol, consulte the section called “Asignación del rol de IAM”.
- La versión `2.12.3` o posterior, o bien, la versión `1.27.160` o posterior de la AWS interfaz de la línea de comandos (AWS CLI) instalada y configurada en su dispositivo o AWS CloudShell. Para comprobar su versión actual, utilice `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente. Para actualizarla, consulte Instalación de la CLI de AWS en su directorio principal en la Guía del usuario de AWS CloudShell.
- La herramienta de línea de comandos de `kubectl` está instalada en su dispositivo o AWS CloudShell. La versión puede ser la misma o hasta una versión secundaria anterior o posterior a la versión de Kubernetes de su clúster. Por ejemplo, si la versión del clúster es `1.29`, puede usar la versión `1.28`, `1.29` o `1.30` de `kubectl` con él. Para instalar o actualizar `kubectl`, consulte the section called “Configure kubectl y eksctl”.
- Un archivo `config` de `kubectl` existente que contenga la configuración del clúster. Para crear un archivo `config` de `kubectl`, consulte the section called “Acceso al clúster con kubectl”.

1. Use el siguiente comando para crear un manifiesto de implementación que puede implementar en un pod con el que se puede confirmar la configuración. Sustituya los valores de ejemplo por sus propios valores.

```
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. Implemente el manifiesto en el clúster.

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

3. Confirme que existan las variables de entorno necesarias para su pod.

   a. Consulte los pods que se implementaron en el paso anterior.

   ```
   kubectl get pods | grep my-app
   ```

   Un ejemplo de salida sería el siguiente.

   ```
   my-app-6f4dfff6cb-76cv9   1/1     Running   0          3m28s
   ```

   b. Consulte el ARN del rol de IAM que esté utilizando el pod.

   ```
   kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep AWS_ROLE_ARN:
   ```

Un ejemplo de salida sería el siguiente.

```
AWS_ROLE_ARN:                 arn:aws:iam::111122223333:role/my-role
```

El ARN del rol debe coincidir con el ARN del rol con el que anotó la cuenta de servicio existente. Para obtener más información sobre cómo anotar la cuenta de servicio, consulte the section called “Asignación del rol de IAM”.

c. Confirme que el pod tenga un montaje de archivo de token de identidad web.

```
kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep AWS_WEB_IDENTITY_TOKEN_FILE:
```

Un ejemplo de salida sería el siguiente.

```
AWS_WEB_IDENTITY_TOKEN_FILE:  /var/run/secrets/eks.amazonaws.com/serviceaccount/
token
```

El `kubelet` solicita y almacena el token en nombre del pod. De forma predeterminada, el `kubelet` actualiza el token si tiene más del 80 % de su tiempo de vida total o más de 24 horas. Puede modificar la duración del vencimiento de cualquier cuenta, excepto la cuenta de servicio predeterminada, con la configuración en la especificación del pod. Para obtener más información, consulte Proyección del volumen del token de la cuenta de servicio en la documentación de Kubernetes.

El webhook de Pod Identity de Amazon EKS del clúster observa los pods que usan una cuenta de servicio con la siguiente anotación:

```
eks.amazonaws.com/role-arn: arn:aws:iam::111122223333:role/my-role
```

El webhook aplica las variables de entorno anteriores a esos pods. El clúster no necesita utilizar el webhook para configurar las variables de entorno y los montajes de archivos de token. Puede configurar manualmente los pods para que tengan estas variables de entorno. Las versiones compatibles del SDK de AWS buscan primero estas variables de entorno en el proveedor de la cadena de credenciales. Las credenciales del rol se utilizan para los pods que cumplen estos criterios.

4. Confirme que sus pods puedan interactuar con los servicios de AWS mediante los permisos que asignó en la política de IAM adjunta a su rol.

> **Note**
>
> Cuando un pod utiliza credenciales de AWS de un rol de IAM asociado a una cuenta de servicio, la AWS CLI u otros SDK de los contenedores de ese pod utilizan las credenciales proporcionadas por dicho rol. Si no restringe el acceso a las credenciales que se proporcionan al rol de IAM del nodo de Amazon EKS, el pod sigue teniendo acceso a esas credenciales. Para obtener más información, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.

Si sus pods no pueden interactuar con los servicios como esperaba, siga estos pasos para confirmar que todo esté configurado correctamente.

a. Confirme que los pods usen una versión del AWS SDK que admita asumir un rol de IAM a través de un archivo de token de identidad web de OpenID Connect. Para obtener más información, consulte the section called “ SDK admitidos”.

b. Confirme que la implementación use la cuenta de servicio.

```
kubectl describe deployment my-app | grep "Service Account"
```

Un ejemplo de salida sería el siguiente.

```
Service Account:  my-service-account
```

c. Si sus pods siguen sin poder acceder a los servicios, revise los pasos que se describen en Asignación de roles de IAM a cuentas de servicio de Kubernetes para confirmar que el rol y la cuenta de servicio se hayan configurado correctamente.

Configure el punto de conexión AWS Security Token Service de una cuenta de servicio

Si utiliza una cuenta de servicio de Kubernetes con roles de IAM para cuentas de servicio, puede configurar el tipo de punto de conexión de AWS Security Token Service utilizado por la cuenta de servicio.

AWS recomienda utilizar los puntos de conexión de AWS STS regionales en lugar del punto de conexión global. Esto reduce la latencia, proporciona redundancia integrada y aumenta la validez de

los tokens de sesión. AWS Security Token Service debe estar activo en la región de AWS donde el pod se ejecuta. Además, su aplicación debe tener incorporada una redundancia para una región de AWS diferente en caso de que ocurra un error en el servicio en la región de AWS. Para obtener más información, consulte Administración de AWS STS en una región de AWS en la Guía del usuario de IAM.

- Un clúster existente. Si no tiene uno, puede crearlo mediante una de las guías en Introducción.
- Un proveedor de OIDC de IAM existente para el clúster. Para obtener más información, consulte the section called “Proveedor de OIDC de IAM”.
- Una cuenta de servicio de Kubernetes existente configurada para su uso con la característica IAM de Amazon EKS para cuentas de servicio.

Todos los siguientes ejemplos utilizan la cuenta de servicio de Kubernetes aws-node utilizada por el complemento CNI de Amazon VPC. Puede reemplazar los *valores de ejemplo* por sus propias cuentas de servicio, pods, espacios de nombres y otros recursos.

1. Seleccione un pod que utilice una cuenta de servicio para la que desee cambiar el punto de conexión. Determine en qué región de AWS se ejecuta el pod. Reemplace *aws-node-6mfgv* por el nombre de su pod y *kube-system* con el espacio de nombres de su pod.

   ```
   kubectl describe pod aws-node-6mfgv -n kube-system |grep Node:
   ```

   Un ejemplo de salida sería el siguiente.

   ```
   ip-192-168-79-166.us-west-2/192.168.79.166
   ```

   En el resultado anterior, el pod se ejecuta en un nodo de la región de AWS us-west-2.
2. Determine el tipo de punto de conexión que utiliza la cuenta de servicio del pod.

   ```
   kubectl describe pod aws-node-6mfgv -n kube-system |grep AWS_STS_REGIONAL_ENDPOINTS
   ```

   Un ejemplo de salida sería el siguiente.

   ```
   AWS_STS_REGIONAL_ENDPOINTS: regional
   ```

Si el punto de conexión actual es global, `global` se devuelven en la salida. Si no se devuelve ningún resultado, el tipo de punto de conexión predeterminado está en uso y no se ha anulado.

3. Si la versión de clúster o plataforma es la misma o posterior a las enumeradas en la tabla, puede cambiar el tipo de punto de conexión utilizado por la cuenta de servicio del tipo predeterminado a otro con uno de los siguientes comandos. Reemplace *aws-node* con el nombre de su cuenta de servicio y *kube-system* con el espacio de nombres de su cuenta de servicio.

   - Si el tipo de punto de conexión predeterminado o actual es global y desea cambiarlo a regional:

     ```
     kubectl annotate serviceaccount -n kube-system aws-node eks.amazonaws.com/sts-regional-endpoints=true
     ```

     Si utiliza roles de IAM para cuentas de servicio con el fin de generar URL de S3 previamente firmadas en la aplicación que se ejecuta en los contenedores de los pods, el formato de la URL de los puntos de conexión regionales es similar al siguiente ejemplo:

     ```
     https://bucket.s3.us-west-2.amazonaws.com/path?...&X-Amz-Credential=your-access-key-id/date/us-west-2/s3/aws4_request&...
     ```

   - Si el tipo de punto de conexión predeterminado o actual es regional y desea cambiarlo a global:

     ```
     kubectl annotate serviceaccount -n kube-system aws-node eks.amazonaws.com/sts-regional-endpoints=false
     ```

     Si su aplicación realiza solicitudes explícitamente a puntos de conexión globales de AWS STS y no anula el comportamiento predeterminado de utilizar puntos de conexión regionales en clústeres de Amazon EKS, las solicitudes fallarán y mostrarán un error. Para obtener más información, consulte the section called “Los contenedores de pods muestran el siguiente error: `An error occurred (SignatureDoesNotMatch) when calling the GetCallerIdentity operation: Credential should be scoped to a valid region`”.

     Si utiliza roles de IAM para cuentas de servicio con el fin de generar URL de S3 previamente firmadas en la aplicación que se ejecuta en los contenedores de los pods, el formato de la URL de los puntos de conexión globales es similar al siguiente ejemplo:

     ```
     https://bucket.s3.amazonaws.com/path?...&X-Amz-Credential=your-access-key-id/date/us-west-2/s3/aws4_request&...
     ```

Si tiene una automatización que espera la URL prefirmada en un formato específico, o si su aplicación o sus dependencias descendentes que utilizan URL prefirmadas requieren una región de AWS de destino específica, realice los cambios necesarios para utilizar el punto de conexión de AWS STS.

4. Elimine y vuelva a crear todos los pods existentes asociados a la cuenta de servicio para aplicar las variables de entorno de credenciales. El enlace web mutante no se aplica a los pods que ya están en ejecución. Puede reemplazar *pods*, *kube-system* y *-l k8s-app=aws-node* por la información de los pods para los que configuró la anotación.

```
kubectl delete Pods -n kube-system -l k8s-app=aws-node
```

5. Confirme que todos los pods se reiniciaron.

```
kubectl get Pods -n kube-system -l k8s-app=aws-node
```

6. Consulte las variables de entorno de uno de los pods. Compruebe que el valor `AWS_STS_REGIONAL_ENDPOINTS` sea el que estableció en un paso anterior.

```
kubectl describe pod aws-node-kzbtr -n kube-system |grep AWS_STS_REGIONAL_ENDPOINTS
```

Un ejemplo de salida sería el siguiente.

```
AWS_STS_REGIONAL_ENDPOINTS=regional
```

Autenticación en otra cuenta mediante IRSA

Puede configurar permisos de IAM entre cuentas mediante la creación de un proveedor de identidades a partir del clúster de otra cuenta o mediante operaciones `AssumeRole` encadenadas. En los siguientes ejemplos, la cuenta A posee un clúster de Amazon EKS que admite roles de IAM para las cuentas de servicio. Los pods que se ejecutan en ese clúster deben asumir permisos de IAM de la cuenta B.

Example Creación de un proveedor de identidades a partir del clúster de otra cuenta

Example

En este ejemplo, la Cuenta A proporciona a la Cuenta B la URL del emisor de OpenID Connect (OIDC) desde su clúster. La cuenta B sigue las instrucciones que se encuentran en Creación de

un proveedor de OIDC de IAM para su clúster y the section called “Asignación del rol de IAM” mediante la URL del emisor de OIDC del clúster de la cuenta A. A continuación, un administrador del clúster anota la cuenta de servicio en el clúster de la cuenta A para utilizar el rol de la cuenta B (*444455556666*).

```
apiVersion: v1
kind: ServiceAccount
metadata:
  annotations:
    eks.amazonaws.com/role-arn: arn:aws:iam::444455556666:role/account-b-role
```

Example Uso de operaciones **`AssumeRole`** encadenadas

Example

En este ejemplo, la cuenta B crea una política de IAM con los permisos que debe otorgar a los pods del clúster de la cuenta A. La cuenta B (*444455556666*) adjunta dicha política a un rol de IAM con una relación de confianza que concede permisos de `AssumeRole` a la cuenta A (*111122223333*).

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "sts:AssumeRole",
      "Condition": {}
    }
  ]
}
```

La cuenta A crea un rol con una política de confianza que obtiene las credenciales del proveedor de identidades creado con la dirección del emisor OIDC del clúster.

```
{
  "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"
        }
      }
    }
  ]
}
```

La cuenta A asocia una política a ese rol con los siguientes permisos para asumir el rol que ha creado la cuenta B.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::444455556666:role/account-b-role"
        }
    ]
}
```

El código de aplicación para que los pods asuman el rol de la cuenta B utiliza dos perfiles: `account_b_role` y `account_a_role`. El perfil `account_b_role` utiliza el perfil `account_a_role` como origen. Para la CLI de AWS, el archivo `~/.aws/config` es similar al siguiente.

```
[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
```

Para especificar perfiles encadenados para otros SDK de AWS, consulte la documentación del SDK que utiliza. Para obtener más información, consulte herramientas para crear en AWS.

Uso de IRSA con el SDK de AWS

Uso de las credenciales

Para usar las credenciales de los roles de IAM para cuentas de servicio (IRSA), el código puede usar cualquier AWS SDK para crear un cliente para un servicio de AWS con un SDK y, de forma predeterminada, el SDK busca en una cadena de ubicaciones las credenciales de AWS Identity and Access Management para usar. Las credenciales de los roles de IAM para cuentas de servicio se utilizarán si no especifica un proveedor de credenciales al crear el cliente o al iniciar el SDK de otro modo.

Esto funciona porque los roles de IAM para cuentas de servicio se han agregado como un paso en la cadena de credenciales predeterminada. Si sus cargas de trabajo utilizan actualmente credenciales que se encuentran en una fase anterior de la cadena de credenciales, esas credenciales seguirán utilizándose aunque configure un rol de IAM para cuentas de servicio de la misma carga de trabajo.

El SDK intercambia automáticamente el token de OIDC de la cuenta de servicio por credenciales temporales de AWS Security Token Service mediante la acción `AssumeRoleWithWebIdentity`. Amazon EKS y esta acción de SDK siguen rotando las credenciales temporales y las renuevan antes de que caduquen.

Al usar roles de IAM para cuentas de servicio, los contenedores de los pods deben usar una versión del AWS SDK que admita asumir un rol de IAM a través de un archivo de token de identidad web de OpenID Connect. Asegúrese de usar las siguientes versiones, o posteriores, para el SDK de AWS:

- Java (Versión 2): 2.10.11
- Java: 1.12.782
- AWS SDK para Go v1: 1.23.13
- AWS SDK para Go v2: compatibilidad con todas las versiones
- Python (Boto3): 1.9.220
- Python (botocore): 1.12.200
- AWS CLI: 1.16.232
- Nodo: 2.525.0 y 3.27.0

- Ruby: 3.58.0
- C++: 1.7.174
- .NET: 3.3.659.1: también debe incluir `AWSSDK.SecurityToken`.
- PHP: 3.110.7

Muchos complementos populares de Kubernetes, como el Escalador automático de clústeres, el Enrutamiento de tráfico de Internet con el controlador del equilibrador de carga de AWS y el complemento CNI de Amazon VPC para Kubernetes permiten utilizar roles de IAM para cuentas de servicio.

Para asegurarse de que esté utilizando un SDK compatible, siga las instrucciones de instalación para su SDK preferido en Herramientas para crear en AWS al crear los contenedores.

Consideraciones

Java

Cuando se utiliza Java, se debe incluir el módulo `sts` en el classpath. Para obtener más información, consulte WebIdentityTokenFileCredentialsProvider en la documentación del SDK de Java.

Obtención de las claves de firma para validar los tokens de OIDC

Kubernetes emite un `ProjectedServiceAccountToken` para cada cuenta de servicio de Kubernetes. Este token es un token de OIDC, que, además, es un tipo de token web JSON (JWT). Amazon EKS aloja un punto de conexión de OIDC público por cada clúster que contiene las claves de firma para el token de modo que los sistemas externos pueden validarlo.

Para validar un `ProjectedServiceAccountToken`, necesita buscar las claves de firma pública de OIDC, también conocidas como JSON Web Key Set (JWKS). Utilice estas claves en su aplicación para validar el token. Por ejemplo, puede usar la biblioteca Python PyJWT para validar los tokens con estas claves. Para más información sobre `ProjectedServiceAccountToken`, consulte the section called “Información general de IAM, Kubernetes y OpenID Connect (OIDC)”.

Requisitos previos

- Un proveedor existente de OpenID Connect (OIDC) de AWS Identity and Access Management (IAM) para su clúster. Para determinar si ya tiene un proveedor o para crear uno, consulte the section called “Proveedor de OIDC de IAM”.

- AWS CLI: una herramienta de línea de comandos para trabajar con servicios de AWS, incluido Amazon EKS. Para obtener más información, consulte Instalación en la Guía del usuario de AWS Command Line Interface. Después de instalar la AWS CLI, recomendamos que también la configure. Para obtener más información, consulte Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS.

Procedimiento

1. Recupere la URL de OIDC del clúster de Amazon EKS mediante la AWS CLI.

```
$ aws eks describe-cluster --name my-cluster --query 'cluster.identity.oidc.issuer'
"https://oidc.eks.us-west-2.amazonaws.com/id/8EBDXXXX00BAE"
```

2. Recupere la clave de firma pública con curl o una herramienta similar. El resultado es un Conjunto de claves web JSON (JWKS).

> **Important**
>
> Amazon EKS limita las llamadas al punto de conexión de OIDC. Debe almacenar en caché la clave de firma pública. Respete el encabezado `cache-control` incluido en la respuesta.

> **Important**
>
> Amazon EKS rota la clave de firma de OIDC cada siete días.

```
$ 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"}]
```

Más información sobre cómo Pod Identity de EKS concede a los pods acceso a los servicios de AWS

Las aplicaciones de los contenedores de un pod pueden usar un SDK de AWS o AWS CLI para llevar a cabo solicitudes de API a servicios de AWS mediante permisos de AWS Identity and Access

Management (IAM). Las aplicaciones deben firmar sus solicitudes de API AWS con credenciales de AWS.

Pod Identities de EKS ofrecen la posibilidad de administrar las credenciales para las aplicaciones, de un modo similar a cómo los perfiles de instancia de Amazon EC2 proporcionan credenciales a instancias de Amazon EC2. En lugar de crear y distribuir las credenciales de AWS a los contenedores o de utilizar el rol de la instancia de Amazon EC2, puede asociar el rol de IAM con una cuenta de servicio de Kubernetes y configurar los pods para usar la cuenta de servicio.

Cada asociación de Pod Identity de EKS asigna un rol a una cuenta de servicio en un espacio de nombres del clúster especificado. Si tiene la misma aplicación en varios clústeres, puede crear asociaciones idénticas en cada clúster sin modificar la política de confianza del rol.

Si un pod usa una cuenta de servicio que tiene una asociación, Amazon EKS establece las variables de entorno en los contenedores del pod. Las variables de entorno configuran los SDK de AWS, incluida la AWS de CLI, para usar las credenciales de la Pod Identity de EKS.

Ventajas de las Pod Identities de EKS

Las Pod Identities de EKS proporcionan los siguientes beneficios:

- Privilegio mínimo: puede limitar los permisos de IAM a una cuenta de servicio y solo los pods que utilizan esa cuenta de servicio tienen acceso a esos permisos. Esta característica también elimina la necesidad de soluciones de terceros como `kiam` o `kube2iam`.
- Aislamiento de credenciales: cuando se restringe el acceso al Servicio de metadatos de instancias (IMDS) de Amazon EC2, los contenedores de un pod solo pueden recuperar las credenciales para el rol de IAM asociado a la cuenta de servicio que usa el contenedor. Un contenedor nunca tiene acceso a credenciales que utilizan otros contenedores de otros pods. Si no se restringe el IMDS, los contenedores del pod también tienen acceso al rol de IAM del nodo de Amazon EKS y es posible que los contenedores puedan acceder a las credenciales de los roles de IAM de otros pods del mismo nodo. Para obtener más información, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.

> **Note**
>
> Los pods configurados con `hostNetwork: true` siempre tendrán acceso al IMDS, pero los SDK y la CLI de AWS utilizarán las credenciales de Pod Identity cuando estén habilitados.

- Auditabilidad: El acceso y el registro de eventos está disponible a través de AWS CloudTrail para facilitar una auditoría retrospectiva.

> **Important**
>
> Los contenedores no funcionan como un límite de seguridad, y el uso de Pod Identity no cambia esto. Los pods asignados al mismo nodo compartirán un kernel y posiblemente otros recursos, según la configuración del pod. Aunque los pods que se ejecutan en nodos separados estarán aislados en la capa de procesamiento, hay aplicaciones de nodos que tienen permisos adicionales en la API de Kubernetes más allá del ámbito de una instancia individual. Algunos ejemplos son `kubelet`, `kube-proxy`, controladores de almacenamiento CSI o sus propias aplicaciones de Kubernetes.

Pod Identity de EKS es un método más sencillo que the section called “Credenciales con IRSA”, ya que no utiliza proveedores de identidad de OIDC. Pod Identity de EKS incluye las siguientes mejoras:

- Operaciones independientes: en muchas organizaciones, la creación de proveedores de identidad de OIDC es una responsabilidad de equipos diferentes a la de administrar los clústeres de Kubernetes. Pod Identity de EKS tiene una clara separación de funciones, en donde toda la configuración de las asociaciones de Pod Identity de EKS se realiza en Amazon EKS y toda la configuración de los permisos de IAM se realiza en IAM.
- Reutilización: La Pod Identity de EKS utiliza una única entidad principal de IAM en lugar de los principios independientes para cada clúster que utilizan los roles de IAM para cuentas de servicio. El administrador de IAM añade la siguiente entidad principal a la política de confianza de cualquier función para que Pod Identities de EKS puedan utilizarla.

```
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            }
```

- Escalabilidad: cada conjunto de credenciales temporales lo asume el servicio de autenticación de EKS en Pod Identity de EKS, en lugar de cada AWS SDK que se ejecuta en cada pod. A continuación, el agente de Pod Identity de Amazon EKS que se ejecuta en cada nodo emite las credenciales de los SDK. De este modo, la carga se reduce a una vez para cada nodo y no se duplica en cada pod. Para obtener más información del proceso, consulte the section called “Funcionamiento”.

Para obtener más información sobre cómo comparar las dos alternativas, consulte the section called “Acceso de la carga de trabajo a AWS ”.

Información general de configuración de las Pod Identities de EKS

Siga estos procedimientos para activar Pod Identities de EKS:

1. the section called “Configuración del agente”: solo complete este procedimiento una vez para cada clúster. No necesita completar este paso si el modo automático de EKS está habilitado en el clúster.
2. the section called “Asignación del rol de IAM”: complete este procedimiento para cada conjunto único de permisos que desee que tenga una aplicación.
3. the section called “Cuenta de servicio del pod”: complete este procedimiento para cada pod que necesite acceso a servicios de AWS.
4. the section called “ SDK admitidos”: confirme que la carga de trabajo utilice un SDK de AWS de una versión compatible y que utilice la cadena de credenciales predeterminada.

Límites

- Se admiten hasta 5000 asociaciones de EKS Pod Identity por clúster para asignar roles de IAM a cuentas de servicio de Kubernetes.

Consideraciones

- Asociación de roles de IAM: cada cuenta de servicio de Kubernetes en un clúster se puede asociar a un rol de IAM de la misma cuenta de AWS que el clúster. Para cambiar el rol, edite la asociación de EKS Pod Identity. Para el acceso entre cuentas, delegue el acceso al rol mediante los roles de IAM. Para obtener más información, consulte Delegación del acceso entre cuentas de AWS mediante roles de IAM en la Guía del usuario de IAM.
- Agente de EKS Pod Identity: se requiere el agente de Pod Identity para utilizar EKS Pod Identity. El agente se ejecuta como un `DaemonSet` de Kubernetes en los nodos del clúster y proporciona credenciales solo a los pods del mismo nodo. Utiliza la `hostNetwork` del nodo y ocupa los puertos `80` y `2703` en la dirección local de enlace (`169.254.170.23` para IPv4, `[fd00:ec2::23]` para IPv6). Si IPv6 está deshabilitado en su clúster, deshabilite IPv6 para el agente de Pod Identity. Para obtener más información, consulte Deshabilitar IPv6 en el agente de EKS Pod Identity.

- Consistencia eventual: las asociaciones de EKS Pod Identity tienen consistencia final, con posibles demoras de varios segundos después de las llamadas a la API. Evite crear o actualizar asociaciones en rutas de código críticas y de alta disponibilidad. En cambio, realice estas acciones en rutinas de inicialización o configuración separadas y menos frecuentes. Para obtener más información, consulte Grupos de seguridad por pod en la Guía de prácticas recomendadas de EKS.
- Consideraciones sobre el proxy y el grupo de seguridad: para los pods que utilizan un proxy, añada `169.254.170.23` (IPv4) y `[fd00:ec2::23]` (IPv6) a las variables de entorno de `no_proxy/NO_PROXY` para evitar que se produzcan errores en las solicitudes al agente de EKS Pod Identity. Si utiliza grupos de seguridad para pods con la CNI de AWS VPC, establezca el indicador `ENABLE_POD_ENI` en “true” y el indicador `POD_SECURITY_GROUP_ENFORCING_MODE` en “standard”. Para obtener más información, consulte Asignación de los grupos de seguridad a pods individuales.

Versiones del clúster de Pod Identity de EKS

Para usar EKS Pod Identity, el clúster debe tener una versión de la plataforma igual o posterior a la que se indica en la siguiente tabla, o una versión de Kubernetes posterior a las versiones que se muestran en la tabla. Para buscar la versión sugerida del Agente de Amazon EKS Pod Identity para una versión de Kubernetes, consulte the section called “Comprobación de la compatibilidad”.

Versión de Kubernetes	Versión de la plataforma
Versiones de Kubernetes no enumeradas	Todas las versiones de la plataforma compatibles
`1.28`	`eks.4`

Restricciones de Pod Identity de EKS

Pod Identities de EKS están disponibles en las siguientes ubicaciones:

- Versiones del clúster de Amazon EKS enumeradas en el tema anterior the section called “Versiones del clúster de Pod Identity de EKS”.
- Nodos de trabajo del clúster que son instancias Linux Amazon EC2.

Pod Identities de EKS no están disponibles en las siguientes ubicaciones:

- AWS Outposts.
- Amazon EKS Anywhere.
- Los clústeres de Kubernetes que se crean y ejecutan en Amazon EC2. Los componentes de Pod Identity de EKS solo están disponibles en Amazon EKS.

No puede usar Pod Identities de EKS con:

- Pods que se ejecutan en cualquier lugar, excepto instancias Linux Amazon EC2. No se admiten los pods de Linux y Windows que se ejecutan en AWS Fargate (Fargate). No se admiten pods que se ejecutan en instancias Windows de Amazon EC2.

Descripción del funcionamiento de Pod Identity de EKS

Las asociaciones de Pod Identity de Amazon EKS ofrecen la posibilidad de administrar las credenciales para las aplicaciones, de un modo similar a cómo los perfiles de instancia de Amazon EC2 proporcionan credenciales a instancias de Amazon EC2.

Pod Identity de Amazon EKS proporciona credenciales a sus cargas de trabajo con una API de autenticación de EKS adicional y un pod de agente que se ejecuta en cada nodo.

En sus complementos, como los complementos de Amazon EKS y el controlador autoadministrado, los operadores y otros complementos, el autor debe actualizar el software para utilizar los SDK de AWS más recientes. Para ver la lista de compatibilidad entre Pod Identity de EKS y los complementos fabricados por Amazon EKS, consulte la sección anterior the section called “Restricciones de Pod Identity de EKS”.

Uso de Identidades de pod de EKS en el código

En su código, puede usar los SDK de AWS para acceder a los servicios de AWS. El código se escribe para crear un cliente para un servicio de AWS con un SDK y, de forma predeterminada, el SDK busca en una cadena de ubicaciones las credenciales de AWS Identity and Access Management que se van a utilizar. Una vez que se ha comprobado que las credenciales son válidas, se detiene la búsqueda. Para obtener más información sobre las ubicaciones predeterminadas utilizadas, consulte la cadena de proveedores de credenciales en la Guía de referencia de herramientas y SDK de AWS.

Se han agregado las Pod Identities de EKS al proveedor de credenciales del contenedor, que se busca en un paso de la cadena de credenciales predeterminada. Si sus cargas de trabajo utilizan actualmente credenciales que se encuentran en una fase anterior de la cadena de credenciales, esas credenciales seguirán utilizándose aunque configure una asociación de Pod Identity de EKS para la misma carga de trabajo. De esta forma, puede migrar de forma segura desde otros tipos de credenciales creando primero la asociación antes de eliminar las credenciales antiguas.

El proveedor de credenciales del contenedor proporciona credenciales temporales de un agente que se ejecuta en cada nodo. En Amazon EKS, el agente de Pod Identity de Amazon EKS y en Servicio de contenedor elástico de Amazon, el agente es el `amazon-ecs-agent`. Los SDK utilizan variables de entorno para localizar el agente al que conectarse.

Por el contrario, los roles de IAM para las cuentas de servicio proporcionan un token de identidad web que el SDK de AWS debe intercambiar con AWS Security Token Service usando `AssumeRoleWithWebIdentity`.

Cómo funciona el agente de Pod Identity de EKS con un pod

1. Cuando Amazon EKS inicia un nuevo pod que utiliza una cuenta de servicio con una asociación de Pod Identity de EKS, el clúster agrega el siguiente contenido al manifiesto de pod:

```
    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 selecciona en qué nodo se va a ejecutar el pod. A continuación, el agente de Pod Identity de Amazon EKS del nodo utiliza la acción AssumeRoleForPodIdentity para recuperar las credenciales temporales de la API de autenticación de EKS.
3. El agente de Pod Identity de EKS pone estas credenciales a disposición de los SDK de AWS que ejecuta en sus contenedores.
4. Utilice el SDK en su aplicación sin especificar un proveedor de credenciales para utilizar la cadena de credenciales predeterminada. O bien, puede especificar el proveedor de credenciales del contenedor. Para obtener más información sobre las ubicaciones predeterminadas utilizadas, consulte la cadena de proveedores de credenciales en la Guía de referencia de herramientas y SDK de AWS.
5. El SDK utiliza las variables de entorno para conectarse al agente de Pod Identity de EKS y recuperar las credenciales.

> **Note**
>
> Si sus cargas de trabajo utilizan actualmente credenciales que se encuentran en una fase anterior de la cadena de credenciales, esas credenciales seguirán utilizándose aunque configure una asociación de Pod Identity de EKS para la misma carga de trabajo.

Configuración del agente de Pod Identity de Amazon EKS

Las asociaciones de Pod Identity de Amazon EKS ofrecen la posibilidad de administrar las credenciales para las aplicaciones, de un modo similar a cómo los perfiles de instancia de Amazon EC2 proporcionan credenciales a instancias de Amazon EC2.

Pod Identity de Amazon EKS proporciona credenciales a sus cargas de trabajo con una API de autenticación de EKS adicional y un pod de agente que se ejecuta en cada nodo.

> **Tip**
>
> No es necesario instalar el agente de EKS Pod Identity en los clústeres del modo automático de EKS. Esta capacidad está integrada en el modo automático de EKS.

Consideraciones

- De forma predeterminada, el agente de Pod Identity de EKS está preinstalado en los clústeres del Modo automático de EKS. Para obtener más información, consulte Modo automático de EKS.
- De forma predeterminada, el agente de Pod Identity de EKS escucha en una dirección `IPv4` e `IPv6` para que los pods soliciten credenciales. El agente usa la dirección IP de bucle invertido (localhost) `169.254.170.23` para `IPv4` y la dirección IP de localhost `[fd00:ec2::23]` para `IPv6`.
- Si deshabilita las direcciones `IPv6` o evita las direcciones IP `IPv6` de localhost, el agente no podrá iniciarse. Para iniciar el agente en los nodos que no pueden usar `IPv6`, siga los pasos que se indican en the section called “Desactivación de IPv6” a fin de deshabilitar la configuración `IPv6`.

Creación del agente de Pod Identity de Amazon EKS

Requisitos previos de agente

- Un clúster existente de Amazon EKS. Para implementar uno, consulte Introducción. La versión del clúster y la versión de la plataforma deben ser iguales o posteriores a las versiones indicadas en Versiones del clúster de Pod Identity de EKS.
- El rol de nodo tiene permisos para que el agente realice la acción `AssumeRoleForPodIdentity` en la API de autenticación de EKS. Puede usar la política administrada de AWS: AmazonEKSWorkerNodePolicy o agregar una política personalizada similar a la siguiente:

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks-auth:AssumeRoleForPodIdentity"
            ],
            "Resource": "*"
        }
    ]
}
```

  Esta acción se puede limitar mediante etiquetas para restringir los roles que pueden asumir los pods que utilizan el agente.

- Los nodos pueden acceder y descargar imágenes de Amazon ECR. La imagen del contenedor del complemento se encuentra en los registros que figuran en Visualización de los registros de imágenes de contenedores de Amazon para los complementos de Amazon EKS.

  Tenga en cuenta que puede cambiar la ubicación de la imagen y proporcionar `imagePullSecrets` para los complementos de EKS en Valores de configuración opcionales en la Consola de administración de AWS, y en `--configuration-values` en la AWS CLI.
- Los nodos pueden acceder a la API de autenticación de Amazon EKS. En el caso de los clústeres privados, se requiere el punto de conexión `eks-auth` en AWS PrivateLink.

Configuración del agente con la consola de AWS

1. Abra la consola de Amazon EKS.
2. En el panel de navegación izquierdo, seleccione Clústeres y, a continuación, seleccione el nombre del clúster para el que desea configurar el complemento de agente de Pod Identity de EKS.
3. Elija la pestaña Complementos.
4. Escoja Obtener más complementos.
5. Seleccione la casilla situada en la parte superior derecha del cuadro de complementos para el agente de Pod Identity de EKS y, a continuación, elija Siguiente.
6. En la página Configurar las opciones de complementos seleccionados, selecciona cualquier versión de la lista desplegable Versión.
7. (Opcional) Expanda Valores de configuración opcionales para introducir una configuración adicional. Por ejemplo, puede proporcionar una ubicación de imagen de contenedor alternativa e `ImagePullSecrets`. El esquema JSON con las claves aceptadas se muestra en Esquema de configuración de complementos.

   Introduzca las claves y los valores de configuración en Valores de configuración.
8. Elija Siguiente.
9. Confirme que los pods de agente de Pod Identity de EKS se estén ejecutando en su clúster.

```
kubectl get pods -n kube-system | grep 'eks-pod-identity-agent'
```

Un ejemplo de salida sería el siguiente.

```
eks-pod-identity-agent-gmqp7                                          1/1     Running
   1 (24h ago)   24h
```

```
eks-pod-identity-agent-prnsh                                    1/1     Running
   1 (24h ago)   24h
```

Ahora puede usar las asociaciones de Pod Identity de EKS en su clúster. Para obtener más información, consulte the section called “Asignación del rol de IAM”.

Configuración del agente con AWS CLI

1. Ejecute el siguiente comando de AWS CLI. Reemplace `my-cluster` por el nombre del clúster.

```
aws eks create-addon --cluster-name my-cluster --addon-name eks-pod-identity-agent --
addon-version v1.0.0-eksbuild.1
```

> **Note**
>
> El agente de Pod Identity de EKS no utiliza el `service-account-role-arn` en Roles de IAM para cuentas de servicio. Debe proporcionar al agente de Pod Identity de EKS los permisos en el rol de nodo.

2. Confirme que los pods de agente de Pod Identity de EKS se estén ejecutando en su clúster.

```
kubectl get pods -n kube-system | grep 'eks-pod-identity-agent'
```

Un ejemplo de salida sería el siguiente.

```
eks-pod-identity-agent-gmqp7                                    1/1     Running
   1 (24h ago)   24h
eks-pod-identity-agent-prnsh                                    1/1     Running
   1 (24h ago)   24h
```

Ahora puede usar las asociaciones de Pod Identity de EKS en su clúster. Para obtener más información, consulte the section called “Asignación del rol de IAM”.

Asignación de un rol de IAM a una cuenta de servicio de Kubernetes

En este tema se explica cómo configurar una cuenta de servicio de Kubernetes para asumir un rol de AWS Identity and Access Management (IAM) con Pod Identity de EKS. Los pods que estén

configurados para usar la cuenta de servicio pueden acceder a cualquier servicio de AWS al que el rol tenga permisos para acceder.

Para crear una asociación de Pod Identity de EKS, solo hay un paso: se crea la asociación en EKS a través de la Consola de administración de AWS, AWS CLI, los SDK de AWS, AWS CloudFormation y otras herramientas. No existen datos ni metadatos sobre las asociaciones dentro del clúster en ningún objeto de Kubernetes y no se agrega ninguna anotación a las cuentas de servicio.

Requisitos previos

- Un clúster existente. Si no tiene uno, puede crearlo mediante una de las siguientes guías de Introducción.
- La entidad principal de IAM que va a crear la asociación debe tener `iam:PassRole`.
- La última versión de AWS CLI instalada y configurada en su dispositivo o AWS CloudShell. Puede comprobar su versión actual con `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente. Para actualizarla, consulte Instalación de AWS CLI en su directorio de usuarios principal en la Guía del usuario de AWS CloudShell.
- La herramienta de línea de comandos de `kubectl` está instalada en su dispositivo o AWS CloudShell. La versión puede ser la misma o hasta una versión secundaria anterior o posterior a la versión de Kubernetes de su clúster. Por ejemplo, si la versión del clúster es `1.29`, puede usar la versión `1.28`, `1.29` o `1.30` de `kubectl` con él. Para instalar o actualizar `kubectl`, consulte the section called “Configure kubectl y eksctl”.
- Un archivo `config` de `kubectl` existente que contenga la configuración del clúster. Para crear un archivo `config` de `kubectl`, consulte the section called “Acceso al clúster con kubectl”.

Creación de una asociación de Pod Identity (consola de AWS)

1. Abra la consola de Amazon EKS.
2. En el panel de navegación izquierdo, seleccione Clústeres y, a continuación, seleccione el nombre del clúster para el que desea configurar el complemento de agente de Pod Identity de EKS.
3. Elija la pestaña Acceder.

4. En las Asociaciones de Pod Identity, elija Crear.
5. Para el Rol de IAM, seleccione el rol de IAM con los permisos que quiere que tenga la carga de trabajo.

> **Note**
>
> La lista solo contiene roles que tienen la siguiente política de confianza, que permite a Pod Identity de EKS utilizarlas.

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

`sts:AssumeRole`: Pod Identity de EKS usa `AssumeRole` para asumir el rol de IAM antes de pasar las credenciales temporales a sus pods.

`sts:TagSession`: Pod Identity de EKS usa `TagSession` para incluir etiquetas de sesión en las solicitudes para AWS.

Puede utilizar estas etiquetas en las claves de condición de la política de confianza para restringir qué cuentas de servicio, espacios de nombres y clústeres pueden utilizar este rol.

Para obtener una lista de las claves de condición de Amazon EKS, consulte Condiciones de Amazon Elastic Kubernetes Service en la Referencia de autorizaciones de servicio. Para obtener más información sobre las acciones y los recursos con los que puede utilizar una clave de condición, consulte Acciones definidas por Amazon Elastic Kubernetes Service.

6. Para el espacio de nombres de Kubernetes, seleccione el espacio de nombres de Kubernetes que contiene la cuenta de servicio y la carga de trabajo. Si lo desea, puede especificar un espacio de nombres por nombre que no existe en el clúster.
7. Para la cuenta de servicio de Kubernetes, seleccione la cuenta de servicio de Kubernetes que se usará. El manifiesto de su carga de trabajo de Kubernetes debe especificar esta cuenta de servicio. Si lo desea, puede especificar una cuenta de servicio por nombre que no exista en el clúster.
8. (Opcional) Para las Etiquetas, elija Agregar etiqueta para agregar metadatos en un par clave-valor. Estas etiquetas se aplican a la asociación y se pueden utilizar en las políticas de IAM.

   Puede repetir este paso para agregar varias etiquetas.
9. Seleccione Crear.

Cree una asociación de Pod Identity (AWS CLI)

1. Si desea asociar una política de IAM existente a su rol de IAM, vaya al siguiente paso.

   Cree una política de IAM. Puede crear su propia política o copiar una política administrada de AWS que ya conceda algunos de los permisos que necesita y personalizarla según sus requisitos específicos. Para obtener más información, consulte Creación de políticas de IAM en la Guía del usuario de IAM.

   a. Cree un archivo que incluya los permisos para los servicios de AWS a los que quiere que accedan sus pods. Para obtener una lista de todas las acciones para todos los servicios de AWS, consulte la Referencia de autorizaciones de servicio.

      Puede ejecutar el siguiente comando para crear un archivo de política de ejemplo que permita el acceso de solo lectura a un bucket de Amazon S3. Opcionalmente, puede almacenar información de configuración o un script de arranque en este bucket, y los contenedores de su pod pueden leer el archivo desde el bucket y cargarlo en su aplicación. Si desea crear esta política de ejemplo, copie el siguiente contenido en su dispositivo. Sustituya *my-pod-secrets-bucket* por el nombre de su bucket y ejecute el comando.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
```

```
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
```

b. Creación de la política de IAM.

```
aws iam create-policy --policy-name my-policy --policy-document file://my-
policy.json
```

2. Cree un rol de IAM y asócielo a una cuenta de servicio de Kubernetes.

a. Si tiene una cuenta de servicio de Kubernetes existente que desea que asuma un rol de IAM, puede omitir este paso.

Cree una cuenta de servicio de Kubernetes. Copie los siguientes contenidos en su dispositivo. Reemplace *my-service-account* por el nombre que desee y *default* por un espacio de nombres diferente, si es necesario. Si cambia *default*, el espacio de nombres debe existir previamente.

```
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
```

Ejecute el siguiente comando.

```
kubectl apply -f my-service-account.yaml
```

b. Ejecute el siguiente comando para crear un archivo de política de confianza para el rol de IAM.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEksAuthToAssumeRoleForPodIdentity",
            "Effect": "Allow",
            "Principal": {
```

```
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
```

c. Creación del rol. Reemplace *my-role* con un nombre para el rol de IAM, y *my-role-description* con una descripción de su rol.

```
aws iam create-role --role-name my-role --assume-role-policy-document file://
trust-relationship.json --description "my-role-description"
```

d. Adjunte la política de IAM al rol. Reemplace *my-role* por el nombre de su rol de IAM y *my-policy* por el nombre de una política existente que haya creado.

```
aws iam attach-role-policy --role-name my-role --policy-
arn=arn:aws:iam::111122223333:policy/my-policy
```

> **Note**
>
> A diferencia de los roles de IAM para cuentas de servicio, Pod Identity de EKS no utiliza ninguna anotación en la cuenta de servicio.

e. Ejecute el siguiente comando para crear la asociación. Reemplace `my-cluster` por el nombre del clúster, reemplace *my-service-account* por el nombre que desee y *default* por un espacio de nombres diferente, si es necesario.

```
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
```

Un ejemplo de salida sería el siguiente.

```
{
    "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**
>
> Puede especificar un espacio de nombres y una cuenta de servicio por nombre que no existe en el clúster. Debe crear el espacio de nombres, la cuenta de servicio y la carga de trabajo que utiliza la cuenta de servicio para que funcione la asociación de Pod Identity de EKS.

Confirmación de configuración

1. Confirme que la política de confianza del rol de IAM se haya configurado correctamente.

```
aws iam get-role --role-name my-role --query Role.AssumeRolePolicyDocument
```

Un ejemplo de salida sería el siguiente.

```
{
    "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. Confirme que la política que adjuntó a su rol en un paso anterior se encuentre adjunta al rol.

```
aws iam list-attached-role-policies --role-name my-role --query
 'AttachedPolicies[].PolicyArn' --output text
```

Un ejemplo de salida sería el siguiente.

```
            arn:aws:iam::111122223333:policy/my-policy
```

3. Establezca una variable para almacenar el nombre de recurso de Amazon (ARN) de la política que quiera utilizar. Reemplace *my-policy* por el nombre de la política para la que desea confirmar los permisos.

```
export policy_arn=arn:aws:iam::111122223333:policy/my-policy
```

4. Vea la versión predeterminada de la política.

```
aws iam get-policy --policy-arn $policy_arn
```

Un ejemplo de salida sería el siguiente.

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "EXAMPLEBIOWGLDEXAMPLE",
        "Arn": "arn:aws:iam::111122223333:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        [...]
    }
}
```

5. Vea el contenido de la política para asegurarse de que incluye todos los permisos que su pod necesita. Si es necesario, reemplace *1* en el siguiente comando por la versión devuelta en la salida anterior.

```
aws iam get-policy-version --policy-arn $policy_arn --version-id v1
```

Un ejemplo de salida sería el siguiente.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
```

Si creó la política de ejemplo en un paso anterior, el resultado es el mismo. Si creó una política diferente, el contenido de *example* es diferente.

Siguientes pasos

the section called “Cuenta de servicio del pod”

Acceso a los recursos de AWS mediante los roles de IAM de destino de EKS Pod Identity

Al ejecutar aplicaciones en Amazon Elastic Kubernetes Service (Amazon EKS), es posible que necesite acceder a los recursos de AWS que existen en cuentas de AWS diferentes. Esta guía le muestra cómo configurar el acceso entre estas cuentas mediante EKS Pod Identity, que permite a sus pods de Kubernetes acceder a otros recursos de AWS con roles de destino.

Requisitos previos

Antes de comenzar, asegúrese de haber seguido estos pasos:

- Configuración del agente de Amazon EKS Pod Identity
- Creación de un rol de EKS Pod Identity

Cómo funciona

Pod Identity permite a las aplicaciones de su clúster de EKS acceder a los recursos de AWS entre cuentas mediante un proceso denominado encadenamiento de roles.

Al crear una asociación de Pod Identity, puede proporcionar dos roles de IAM: un rol de EKS Pod Identity en la misma cuenta que su clúster de EKS y un rol de IAM de destino de la cuenta que contiene los recursos de AWS a los que desea acceder (como buckets de S3 o bases de datos de RDS). El rol de EKS Pod Identity debe estar en la cuenta de su clúster de EKS debido a los requisitos de PassRole de IAM, mientras que el rol de IAM de destino puede estar en cualquier cuenta de AWS. PassRole permite a una entidad de AWS delegar la asunción de roles a otro servicio. EKS Pod Identity usa PassRole para conectar un rol a una cuenta de servicio de Kubernetes, lo que requiere que tanto el rol como la identidad que lo pasa estén en la misma cuenta de AWS que el clúster de EKS. Cuando el pod de la aplicación necesita acceder a los recursos de AWS, solicita las credenciales a Pod Identity. A continuación, Pod Identity lleva a cabo automáticamente dos asunciones de rol en secuencia: primero asume el rol de EKS Pod Identity y, a continuación, utiliza esas credenciales para asumir el rol de IAM de destino. Este proceso proporciona al pod credenciales temporales con los permisos definidos en el rol de destino, lo que permite un acceso seguro a los recursos de otras cuentas de AWS.

Consideraciones sobre el almacenamiento en caché

Debido a los mecanismos de almacenamiento en caché, es posible que las actualizaciones de un rol de IAM en una asociación de Pod Identity existente no se apliquen inmediatamente en los pods que se ejecutan en el clúster de EKS. El agente de Pod Identity almacena en caché las credenciales de IAM en función de la configuración de la asociación en el momento en que se obtienen las credenciales. Si la asociación incluye solo un rol de EKS Pod Identity y ningún rol de IAM de destino, las credenciales almacenadas en caché duran 6 horas. Si la asociación incluye tanto el ARN del rol de EKS Pod Identity como un rol de IAM de destino, las credenciales almacenadas en caché duran 59 minutos. La modificación de una asociación existente, como actualizar el ARN de rol de EKS Pod Identity o agregar un rol de IAM de destino, no restablece la caché existente. Como resultado, el agente no reconocerá las actualizaciones hasta que se actualicen las credenciales almacenadas en caché. Para aplicar los cambios antes, puede volver a crear los pods existentes; de lo contrario, tendrá que esperar a que caduque la caché.

Paso 1: creación y asociación de un rol de IAM de destino

En este paso, establecerá una cadena de confianza segura mediante la creación y configuración de un rol de IAM de destino. A modo de demostración, crearemos un nuevo rol de IAM de destino

para establecer una cadena de confianza entre dos cuentas de AWS: el rol de EKS Pod Identity (por ejemplo, `eks-pod-identity-primary-role`) de la cuenta de AWS del clúster de EKS obtiene permiso para asumir el rol de IAM de destino (por ejemplo, `eks-pod-identity-aws-resources`) en su cuenta de destino, lo que permite el acceso a recursos de AWS como los buckets de Amazon S3.

Creación del rol de IAM de destino

1. Abra la consola de Amazon IAM.
2. En la barra de navegación superior, compruebe que haya iniciado sesión en la cuenta que contiene los recursos de AWS (como buckets de S3 o tablas de DynamoDB) para el rol de IAM de destino.
3. En el panel de navegación izquierdo, elija Roles.
4. Elija el botón Crear rol y, a continuación, seleccione Cuenta de AWS en “Tipo de entidad de confianza”.
5. Elija Otra cuenta de AWS, ingrese su número de cuenta de AWS (la cuenta en la que existe su rol de EKS Pod Identity) y, a continuación, elija Siguiente.
6. Agregue las políticas de permisos que desee asociar al rol (por ejemplo, AmazonS3FullAccess) y, a continuación, elija Siguiente.
7. Ingrese un nombre de rol (por ejemplo, `MyCustomIAMTargetRole`) y, a continuación, elija Crear rol.

Actualización de la política de confianza del rol de IAM de destino

1. Después de crear el rol, volverá a la lista de Roles. Busque y seleccione el nuevo rol que ha creado en el procedimiento anterior (por ejemplo, `MyCustomIAMTargetRole`).
2. Seleccione la pestaña Relaciones de confianza.
3. Haga clic en Editar política de confianza en la parte derecha.
4. En el editor de políticas, reemplace el JSON predeterminado por la política de confianza. Reemplace los valores de marcadores de posición de nombre del rol y `111122223333` en el ARN del rol de IAM por el ID de cuenta de AWS que aloja el clúster de EKS. Si lo desea, también puede usar PrincipalTags en la política de confianza de roles para autorizar solo cuentas de servicio específicas de un clúster y espacio de nombres determinados para que asuman su rol de destino. Por ejemplo:

```
{
  "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"
        }
      }
    }
  ]
}
```

La política anterior permite que el rol `eks-pod-identeity-primary-role` de la cuenta 111122223333 de AWS con las etiquetas de sesión de Pod Identity de EKS correspondientes asuma este rol.

Si ha desactivado las etiquetas de sesión en su Pod Identity de EKS, este también establece el `sts:ExternalId` con información sobre el clúster, el espacio de nombres y la cuenta de servicio de un pod al asumir un rol de destino.

```
{
  "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"
    }
  ]
}
```

La política anterior ayuda a garantizar que solo el clúster, el espacio de nombres y la cuenta de servicio esperados puedan asumir el rol de destino.

Actualización de la política de permisos para el rol de EKS Pod Identity

En este paso, actualizará la política de permisos del rol de EKS Pod Identity asociado a su clúster de Amazon EKS. Para ello, agregará el ARN del rol de IAM de destino como recurso.

1. Abra la consola de Amazon EKS.
2. En el panel de navegación izquierdo, seleccione Clústeres y, a continuación, seleccione el nombre del clúster de EKS.
3. Elija la pestaña Acceso.
4. En Asociaciones de Pod Identity, seleccione su rol de EKS Pod Identity.
5. Elija Permisos, Agregar permisos y, a continuación, Crear política insertada.
6. Elija JSON en el lado derecho.

7. En el editor de políticas, reemplace el JSON predeterminado por la política de permisos. Reemplace el valor de marcadores de posición de nombre del rol y `222233334444` en el ARN del rol de IAM por el rol de IAM de destino. Por ejemplo:

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ],
            "Resource": "arn:aws:iam::222233334444:role/eks-pod-identity-aws-resources"
        }
    ]
}
```

Paso 2: asociación del rol de IAM de destino a una cuenta de servicio de Kubernetes

En este paso, creará una asociación entre el rol de IAM de destino y la cuenta de servicio de Kubernetes en su clúster de EKS.

1. Abra la consola de Amazon EKS.
2. En el panel de navegación izquierdo, seleccione Clústeres y, a continuación, seleccione el nombre del clúster al que desea agregar la asociación.
3. Elija la pestaña Acceder.
4. En las Asociaciones de Pod Identity, elija Crear.
5. Elija el rol de EKS Pod Identity en Rol de IAM para que lo asuman sus cargas de trabajo.
6. Elija el rol de IAM de destino en Rol de IAM de destino que asumirá el rol de EKS Pod Identity.
7. En el campo Espacio de nombres de Kubernetes, ingrese el nombre del espacio de nombres donde desea crear la asociación (por ejemplo, `my-app-namespace`). Esto define dónde reside la cuenta de servicio.
8. En el campo Cuenta de servicio de Kubernetes, ingrese el nombre de la cuenta de servicio (por ejemplo, `my-service-account`) que utilizará las credenciales de IAM. Esto vincula el rol de IAM a la cuenta de servicio.
9. Elija Crear para crear la asociación.

Configuración de pods para acceder a los servicios de AWS con cuentas de servicio

Si un pod necesita acceder a los servicios de AWS, debe configurarlo para que use una cuenta de servicio de Kubernetes. La cuenta de servicio debe estar asociada a un rol de AWS Identity and Access Management (IAM) que tenga permisos para acceder a los servicios de AWS.

- Un clúster existente. Si no tiene uno, puede crearlo mediante una de las guías de Introducción.
- Una cuenta de servicio de Kubernetes existente y una asociación de Pod Identity de EKS que asocia la cuenta de servicio a un rol de IAM. El rol debe tener una política de IAM asociada que contenga los permisos que desee que tengan sus pods para usar servicios de AWS. Para obtener más información acerca de cómo crear y configurar la cuenta de servicio y el rol, consulte the section called “Asignación del rol de IAM”.
- La última versión de AWS CLI instalada y configurada en su dispositivo o AWS CloudShell. Puede comprobar su versión actual con `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente. Para actualizarla, consulte Instalación de AWS CLI en su directorio de usuarios principal en la Guía del usuario de AWS CloudShell.
- La herramienta de línea de comandos de `kubectl` está instalada en su dispositivo o AWS CloudShell. La versión puede ser la misma o hasta una versión secundaria anterior o posterior a la versión de Kubernetes de su clúster. Por ejemplo, si la versión del clúster es `1.29`, puede usar la versión `1.28`, `1.29` o `1.30` de `kubectl` con él. Para instalar o actualizar `kubectl`, consulte the section called “Configure kubectl y eksctl”.
- Un archivo `config` de `kubectl` existente que contenga la configuración del clúster. Para crear un archivo `config` de `kubectl`, consulte the section called “Acceso al clúster con kubectl”.

1. Use el siguiente comando para crear un manifiesto de implementación que puede implementar en un pod con el que se puede confirmar la configuración. Sustituya los valores de ejemplo por sus propios valores.

```
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. Implemente el manifiesto en el clúster.

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

3. Confirme que existan las variables de entorno necesarias para su pod.
   a. Consulte los pods que se implementaron en el paso anterior.

```
kubectl get pods | grep my-app
```

   Un ejemplo de salida sería el siguiente.

```
my-app-6f4dfff6cb-76cv9   1/1     Running   0          3m28s
```

   b. Confirme que el pod tenga un archivo de token de cuenta de servicio montado.

```
kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep
 AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE:
```

   Un ejemplo de salida sería el siguiente.

```
AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE:  /var/run/secrets/
pods.eks.amazonaws.com/serviceaccount/eks-pod-identity-token
```

4. Confirme que sus pods puedan interactuar con los servicios de AWS mediante los permisos que asignó en la política de IAM adjunta a su rol.

> **Note**
>
> Cuando un pod utiliza credenciales de AWS de un rol de IAM asociado a una cuenta de servicio, la AWS CLI u otros SDK de los contenedores de ese pod utilizan las credenciales proporcionadas por dicho rol. Si no restringe el acceso a las credenciales que se proporcionan al rol de IAM del nodo de Amazon EKS, el pod sigue teniendo acceso a esas credenciales. Para obtener más información, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.

Si sus pods no pueden interactuar con los servicios como esperaba, siga estos pasos para confirmar que todo esté configurado correctamente.

a. Confirme que los pods usen una versión del AWS SDK que admita asumir un rol de IAM a través de una asociación de Pod Identity de EKS. Para obtener más información, consulte the section called “ SDK admitidos”.

b. Confirme que la implementación use la cuenta de servicio.

```
kubectl describe deployment my-app | grep "Service Account"
```

Un ejemplo de salida sería el siguiente.

```
Service Account:  my-service-account
```

Concesión de acceso a los pods a los recursos de AWS en función de las etiquetas

El control de acceso basado en atributos (ABAC) concede derechos a los usuarios mediante políticas que combinan atributos. Pod Identity de EKS adjunta etiquetas a las credenciales temporales de cada pod con atributos como el nombre del clúster, el espacio de nombres y el nombre de la cuenta de servicio. Estas etiquetas de sesión de roles permiten a los administradores crear un único rol que funcione en todas las cuentas de servicio al permitir el acceso a los recursos de AWS en función de las etiquetas coincidentes. Al agregar la compatibilidad con las etiquetas de sesión de roles, puede reforzar los límites de seguridad entre los clústeres y las cargas de trabajo dentro de los clústeres, mientras reutilizan los mismos roles y políticas de IAM.

Ejemplo de política con etiquetas

A continuación, se muestra un ejemplo de política de IAM que concede permisos `s3:GetObject` cuando el objeto correspondiente está etiquetado con el nombre del clúster de EKS.

```
{
    "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}"
                }
            }
        }
    ]
}
```

Activación o desactivación de etiquetas de sesión

EKS Pod Identity agrega un conjunto predefinido de etiquetas de sesión cuando asume el rol. Estas etiquetas de sesión permiten a los administradores crear un único rol que funcione en todos los recursos al permitir el acceso a los recursos de AWS en función de las etiquetas coincidentes.

Activación de las etiquetas de sesión

Las etiquetas de sesión se activan automáticamente con EKS Pod Identity, por lo que no es necesario que haga nada. De forma predeterminada, EKS Pod Identity adjunta un conjunto

de etiquetas predefinidas a la sesión. Para hacer referencia a estas etiquetas en las políticas, utilice la sintaxis `${aws:PrincipalTag/` seguida de la clave de etiqueta. Por ejemplo, `${aws:PrincipalTag/kubernetes-namespace}`.

- `eks-cluster-arn`
- `eks-cluster-name`
- `kubernetes-namespace`
- `kubernetes-service-account`
- `kubernetes-pod-name`
- `kubernetes-pod-uid`

Desactivación de las etiquetas de sesión

AWS comprime las políticas de sesión insertadas, los ARN de políticas administradas y las etiquetas de sesión en un formato binario empaquetado que tiene un límite separado. Si recibe un error `PackedPolicyTooLarge` que indica que el formato binario empaquetado ha superado el límite de tamaño, puede desactivar las etiquetas de sesión agregadas por EKS Pod Identity para intentar reducir el tamaño. Para desactivar estas etiquetas de sesión, siga estos pasos:

1. Abra la consola de Amazon EKS.
2. En el panel de navegación izquierdo, seleccione Clústeres y, luego, seleccione el nombre del clúster que desea modificar.
3. Elija la pestaña Acceso.
4. En Asociaciones de Pod Identity, elija el ID de asociación que desea modificar en ID de asociación y, a continuación, Editar.
5. En Etiquetas de sesión, elija Desactivar etiquetas de sesión.
6. Seleccione Save changes (Guardar cambios).

Etiquetas entre cuentas

Todas las etiquetas de sesión que agrega Pod Identity de EKS son transitivas; las claves y los valores de las etiquetas se transfieren a cualquier acción `AssumeRole` que sus cargas de trabajo utilicen para cambiar los roles a otra cuenta. Puede utilizar estas etiquetas en las políticas de otras cuentas para limitar el acceso en situaciones entre cuentas. Para obtener más información, consulte Encadenar roles con etiquetas de sesión en la Guía del usuario de IAM.

Etiquetas personalizadas

Pod Identity de Amazon EKS no puede agregar etiquetas personalizadas adicionales a la acción `AssumeRole` que realiza. Sin embargo, las etiquetas que se aplican al rol de IAM siempre están disponibles en el mismo formato: `${aws:PrincipalTag/` seguido de la clave, por ejemplo, `${aws:PrincipalTag/MyCustomTag}`.

> **Note**
>
> Las etiquetas agregadas a la sesión mediante la solicitud de `sts:AssumeRole` tienen prioridad en caso de conflicto. Por ejemplo, supongamos que:
>
> - Amazon EKS agrega una clave `eks-cluster-name` y un valor `my-cluster` a la sesión cuando EKS asume el rol de cliente y
> - Agrega una etiqueta `eks-cluster-name` al rol de IAM con el valor `my-own-cluster`.
>
> En este caso, prevalecerá la primera y el valor de la etiqueta `eks-cluster-name` será `my-cluster`.

Uso de Pod Identity con el SDK de AWS

Uso de las credenciales de Pod Identity de EKS

Para usar las credenciales de una asociación de Pod Identity de EKS, el código puede usar cualquier SDK de AWS para crear un cliente para un servicio de AWS con un SDK y, de forma predeterminada, el SDK busca en una cadena de ubicaciones las credenciales de AWS Identity and Access Management que desee utilizar. Las credenciales de Pod Identity de EKS se utilizarán si no especifica un proveedor de credenciales al crear el cliente o al iniciar el SDK de otro modo.

Esto funciona porque las Pod Identities de EKS se han agregado al proveedor de credenciales del contenedor, que se busca en un paso de la cadena de credenciales predeterminada. Si sus cargas de trabajo utilizan actualmente credenciales que se encuentran en una fase anterior de la cadena de credenciales, esas credenciales seguirán utilizándose aunque configure una asociación de Pod Identity de EKS para la misma carga de trabajo.

Para obtener más información sobre cómo funcionan las Pod Identities de EKS, consulte the section called “Funcionamiento”.

Para usar Más información sobre cómo EKS Pod Identity concede a los pods acceso a los servicios de AWS, los contenedores de los pods deben utilizar una versión del AWS SDK que admita asumir un rol de IAM del agente de Pod Identity de EKS. Asegúrese de usar las siguientes versiones, o posteriores, para el SDK de AWS:

- Java (Versión 2): 2.21.30
- Java: 1.12.746
- Go v1: v1.47.11
- Go v2: lanzamiento-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: lanzamiento-2024-03-13
- C++: 1.11.263
- .NET: 3.7.734.0
- PowerShell: 4.1.502
- PHP: 3.289.0

Para asegurarse de que esté utilizando un SDK compatible, siga las instrucciones de instalación para su SDK preferido en Herramientas para crear en AWS al crear los contenedores.

Para obtener una lista de los complementos compatibles con Pod Identity de EKS, consulte the section called “Referencia de compatibilidad de Pod Identity”.

Desactivar **IPv6** en el agente de Pod Identity de EKS

Consola de administración de AWS

1. Para deshabilitar IPv6 en el agente de Pod Identity de EKS, añada la siguiente configuración a los ajustes de configuración opcionales del complemento de EKS.
   a. Abra la consola de Amazon EKS.
   b. En el panel de navegación izquierdo, seleccione Clusters (Clústeres) y, a continuación, seleccione el nombre del clúster para el que desea configurar el complemento.
   c. Elija la pestaña Complementos.
   d. Seleccione la casilla situada en la parte superior derecha del cuadro de complementos para el agente de Pod Identity de EKS y, a continuación, elija Editar.
   e. En la página de configuración del agente de Pod Identity de EKS, haga lo siguiente:
      i. Seleccione la Versión que desea utilizar. Le recomendamos que mantenga la misma versión que en el paso anterior y que actualice la versión y la configuración en acciones separadas.
      ii. Seleccione Ajustes de configuración opcionales.
      iii. Introduzca la clave JSON `"agent":` y el valor de un objeto JSON anidado con una clave `"additionalArgs":` en Valores de configuración. El texto resultante debe ser un objeto JSON válido. Si esta clave y este valor son los únicos datos del cuadro de texto, rodee la clave y el valor entre corchetes `{ }`. En el siguiente ejemplo se muestra que la política de red está habilitada:

      ```
      {
          "agent": {
              "additionalArgs": {
                  "-b": "169.254.170.23"
              }
          }
      }
      ```

      En esta configuración, se establece que la dirección IPv4 sea la única dirección que utilice el agente.
   f. Para aplicar la nueva configuración mediante la sustitución de los pods del agente de Pod Identity de EKS, seleccione Guardar cambios.

      Amazon EKS aplica los cambios a los complementos de EKS mediante el despliegue del `DaemonSet` de Kubernetes para el agente de Pod Identity de EKS. Puede hacer un

seguimiento del estado del lanzamiento en el historial de actualizaciones en la Consola de administración de AWS y con `kubectl rollout status daemonset/eks-pod-identity-agent --namespace kube-system`.

`kubectl rollout` tiene los siguientes comandos:

```
$ 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
```

Si la implementación lleva demasiado tiempo, Amazon EKS la anulará y se agregará al historial de actualizaciones del complemento un mensaje con el tipo de actualización del complemento y el estado Fallido. Para investigar cualquier problema, comience por el historial de la implementación y ejecute `kubectl logs` en un pod del agente de Pod Identity de EKS para ver los registros del agente de Pod Identity de EKS.

2. Si la nueva entrada en el historial de actualizaciones tiene el estado Correcto, esto significa que la implementación se ha completado y que el complemento está utilizando la nueva configuración en todos los pods del agente de Pod Identity de EKS.

AWS CLI

1. Para deshabilitar `IPv6` en el agente de Pod Identity de EKS, agregue la siguiente configuración a los valores de configuración del complemento de EKS.

   Ejecute el siguiente comando de AWS CLI. Reemplace `my-cluster` por el nombre del clúster y el ARN del rol de IAM por el rol que va a usar.

   ```
   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"}}}'
   ```

   En esta configuración, se establece que la dirección `IPv4` sea la única dirección que utilice el agente.

Amazon EKS aplica los cambios a los complementos de EKS mediante el despliegue del DaemonSet de Kubernetes para el agente de Pod Identity de EKS. Puede hacer un seguimiento del estado del lanzamiento en el historial de actualizaciones en la Consola de administración de AWS y con `kubectl rollout status daemonset/eks-pod-identity-agent --namespace kube-system`.

`kubectl rollout` tiene los siguientes comandos:

```
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
```

Si la implementación lleva demasiado tiempo, Amazon EKS la anulará y se agregará al historial de actualizaciones del complemento un mensaje con el tipo de actualización del complemento y el estado Fallido. Para investigar cualquier problema, comience por el historial de la implementación y ejecute `kubectl logs` en un pod del agente de Pod Identity de EKS para ver los registros del agente de Pod Identity de EKS.

Creación de un rol de IAM con la política de confianza requerida por Pod Identity de EKS

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

```
        }
    ]
}
```

`sts:AssumeRole`

Pod Identity de EKS usa `AssumeRole` para asumir el rol de IAM antes de pasar las credenciales temporales a sus pods.

`sts:TagSession`

Pod Identity de EKS usa `TagSession` para incluir etiquetas de sesión en las solicitudes para AWS STS.

Establecimiento de condiciones

Puede utilizar estas etiquetas en las claves de condición de la política de confianza para restringir qué cuentas de servicio, espacios de nombres y clústeres pueden utilizar este rol. Para ver la lista de etiquetas de solicitud que agrega Pod Identity, consulte the section called “Activación o desactivación de etiquetas de sesión”.

Por ejemplo, puede restringir los pods que pueden asumir el rol de IAM de Pod Identity a la `ServiceAccount` y el `Namespace` específicos con la siguiente política de confianza con la `Condition` agregada:

```
{
    "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"
                    ]
                }
            }
        }
    ]
}
```

Para obtener una lista de las claves de condición de Amazon EKS, consulte Condiciones de Amazon Elastic Kubernetes Service en la Referencia de autorizaciones de servicio. Para obtener más información sobre las acciones y los recursos con los que puede utilizar una clave de condición, consulte Acciones definidas por Amazon Elastic Kubernetes Service.

Administración de los recursos de computación mediante nodos

Un nodo de Kubernetes es una máquina que ejecuta aplicaciones en contenedores. Cada nodo tiene los siguientes componentes:

- Tiempo de ejecución del contenedor: es el software responsable de ejecutar los contenedores.
- kubelet: se asegura de que los contenedores estén en buen estado y funcionando dentro de su pod asociado.
- kube-proxy: mantiene las reglas de red que permiten la comunicación con sus pods.

Para obtener más información, consulte Nodos en la documentación de Kubernetes.

El clúster de Amazon EKS puede programar pods en cualquier combinación de nodos administrados del modo automático de EKS, nodos autoadministrados, grupos de nodos administrados de Amazon EKS, AWS Fargate y Nodos híbridos de Amazon EKS. Para obtener más información sobre los nodos implementados en el clúster, consulte the section called “Acceso a recursos del clúster”.

> **Note**
>
> A excepción de los nodos híbridos, los nodos deben estar en la misma VPC que las subredes que seleccionó al crear el clúster. Sin embargo, los nodos no tienen que estar en las mismas subredes.

Comparación de opciones de computación

La siguiente tabla proporciona varios criterios para evaluar a la hora de decidir qué opciones se ajustan mejor a sus requisitos. Los nodos autoadministrados son otra opción que cumple todos los criterios enumerados, pero requieren mucho más mantenimiento manual. Para obtener más información, consulte the section called “Nodos autoadministrados”.

> **Note**
>
> Bottlerocket presenta algunas diferencias específicas con respecto a la información general de esta tabla. Para obtener más información, consulte la documentación de Bottlerocket en GitHub.

Criterios	Grupos de nodos administrados por EKS	Modo automático de EKS	Nodos híbridos de Amazon EKS
Se puede implementar en AWS Outposts	No	No	No
Se puede implementar en zonas locales de AWS.	Sí	No	No
Puede ejecutar contenedores que requieren Windows	Sí	No	No
Puede ejecutar contenedores que requieran Linux.	Sí	Sí	Sí
Puede ejecutar cargas de trabajo que requieren el chip de inferencias.	Sí. Solo nodos de Amazon Linux.	Sí	No
Puede ejecutar cargas de trabajo que requieren una GPU.	Sí. Solo nodos de Amazon Linux.	Sí	Sí
Puede ejecutar cargas de trabajo que	Sí	Sí	Sí

Criterios	Grupos de nodos administrados por EKS	Modo automático de EKS	Nodos híbridos de Amazon EKS
requieren procesadores Arm.			
Puede ejecutar AWS Bottlerocket.	Sí	Sí	No
Los pods comparten CPU, memoria, almacenamiento y recursos de red con otros pods.	Sí	Sí	Sí
Debe implementar y administrar instancias de Amazon EC2.	Sí	No, obtenga más información sobre las instancias administradas por EC2	Sí, usted administra las máquinas virtuales o físicas en las instalaciones con las herramientas que elija.
Debe proteger, mantener y aplicar parches al sistema operativo de las instancias de Amazon EC2.	Sí	No	Sí, usted administra el sistema operativo que se ejecuta en las máquinas virtuales o físicas con las herramientas que elija.
Puede proporcionar argumentos de arranque en la implementación de un nodo, como argumentos kubelet adicionales.	Sí, mediante `eksctl` o una plantilla de lanzamiento con una AMI personalizada.	No, utilice `NodeClass` para configurar nodos	Sí, puede personalizar los argumentos de arranque con nodeadm. Consulte the section called “Nodeadm de nodos híbridos”.

Criterios	Grupos de nodos administrados por EKS	Modo automático de EKS	Nodos híbridos de Amazon EKS
Puede asignar direcciones IP a pods desde un bloque de CIDR diferente de la dirección IP asignada al nodo.	Sí. Mediante una plantilla de lanzamiento con una AMI personalizada. Para obtener más información, consulte the section called “Plantillas de inicialización”.	No	Sí, consulte the section called “Cómo configurar una CNI”.
Se puede utilizar SSH en el nodo.	Sí	No, aprenda a solucionar problemas con los nodos	Sí
Puede implementar su propia AMI personalizada en los nodos.	Sí. Mediante una plantilla de lanzamiento.	No	Sí
Puede implementar su propia CNI personalizada en los nodos.	Sí. Mediante una plantilla de lanzamiento con una AMI personalizada.	No	Sí

Criterios	Grupos de nodos administrados por EKS	Modo automático de EKS	Nodos híbridos de Amazon EKS
Debe actualizar la AMI de nodos por su cuenta	Sí: si ha implement ado una AMI optimizada de Amazon EKS, recibirá una notificación en la consola de Amazon EKS cuando las actualizaciones estén disponibles. Puede realizar la actualiza ción con un solo clic de en la consola. Si ha implementado una AMI personalizada, no recibirá una notificac ión en la consola de Amazon EKS cuando las actualiza ciones estén disponibl es. Debe realizar la actualización por su cuenta.	No	Sí, usted administra el sistema operativo que se ejecuta en las máquinas virtuales o físicas con las herramientas que elija. Consulte the section called “Cómo preparar el sistema operativo”.

Criterios	Grupos de nodos administrados por EKS	Modo automático de EKS	Nodos híbridos de Amazon EKS
Debe actualizar la versión de Kubernetes de los nodos por su cuenta.	Sí: si ha implementado una AMI optimizada de Amazon EKS, recibirá una notificación en la consola de Amazon EKS cuando las actualizaciones estén disponibles. Puede realizar la actualización con un solo clic de en la consola. Si ha implementado una AMI personalizada, no recibirá una notificación en la consola de Amazon EKS cuando las actualizaciones estén disponibles. Debe realizar la actualización por su cuenta.	No	Sí, usted administra las actualizaciones de los nodos híbridos con las herramientas que elija o con `nodeadm`. Consulte the section called “Actualización de nodos híbridos”.
Puede utilizar el almacenamiento de Amazon EBS con pods.	Sí	Sí, como una capacidad integrada. Aprenda a crear una clase de almacenamiento.	No
Puede utilizar el almacenamiento de Amazon EFS con pods.	Sí	Sí	No

Criterios	Grupos de nodos administrados por EKS	Modo automático de EKS	Nodos híbridos de Amazon EKS
Puede utilizar el almacenamiento de Amazon FSx para Lustre con pods.	Sí	Sí	No
Puede utilizar el Network Load Balancer para servicios.	Sí	Sí	Sí, debe usar el tipo de destino `ip`.
Los pods pueden ejecutarse en una subred pública.	Sí	Sí	No, los pods se ejecutan en un entorno en las instalaciones.
Puede asignar diferentes grupos de seguridad de VPC a pods individuales.	Sí. Solo nodos de Linux.	No	No
Puede ejecutar DaemonSets de Kubernetes.	Sí	Sí	Sí
Soporte de `HostPort` y `HostNetwork` en el manifiesto del pod.	Sí	Sí	Sí

Criterios	Grupos de nodos administrados por EKS	Modo automático de EKS	Nodos híbridos de Amazon EKS
Disponibilidad regional de AWS	Todas las regiones admitidas por Amazon EKS	Todas las regiones admitidas por Amazon EKS	Todas las regiones compatibles con Amazon EKS excepto las regiones de AWS GovCloud (EE. UU.) y China.
Puede ejecutar contenedores en hosts dedicados de Amazon EC2	Sí	No	No

Criterios	Grupos de nodos administrados por EKS	Modo automático de EKS	Nodos híbridos de Amazon EKS
Precios	Costo de la instancia de Amazon EC2 que ejecuta varios pods. Para obtener más información, consulte Precios de Amazon EC2.	Cuando el modo automático de EKS está habilitado en el clúster, se paga una tarifa aparte, además de las tarifas estándar de las instancias de EC2, correspon diente a las instancia s lanzadas mediante la capacidad de computación del modo automático. El importe varía en función del tipo de instancia lanzada y de la región de AWS en la que se encuentre el clúster. Para obtener más información, consulte los precios de Amazon EKS.	Costo por hora de vCPU de nodos híbridos. Para obtener más información, consulte los precios de Amazon EKS.

Simplificación del ciclo de vida de los nodos con grupos de nodos administrados

Los grupos de nodos administrados por Amazon EKS automatizan el aprovisionamiento y la gestión del ciclo de vida de nodos (instancias de Amazon EC2) para clústeres de Kubernetes de Amazon EKS.

Con los grupos de nodos administrados por Amazon EKS, no es necesario aprovisionar o registrar por separado las instancias de Amazon EC2 que proporcionan capacidad informática para ejecutar

sus aplicaciones de Kubernetes. Puede crear, actualizar o terminar nodos para el clúster con una sola operación y de manera automática. Las actualizaciones y terminaciones de nodos drenan de forma automática los nodos para garantizar que sus aplicaciones permanezcan disponibles.

Todos los nodos administrados se aprovisionan como parte de un grupo de Amazon EC2 Auto Scaling administrado por usted a través de Amazon EKS. Todos los recursos, incluidas las instancias y los grupos de Auto Scaling, se ejecutan dentro de su cuenta de AWS. Cada grupo de nodos puede ejecutarse en varias zonas de disponibilidad que defina.

Los grupos de nodos administrados también pueden aprovechar opcionalmente la reparación automática de nodos, que supervisa continuamente el estado de los nodos. Reacciona automáticamente ante los problemas detectados y reemplaza los nodos cuando es posible. Esto contribuye a mantener la disponibilidad general del clúster con una intervención manual mínima. Para obtener más información, consulte the section called “Estado de los nodos”.

Puede agregar un grupo de nodos administrado a clústeres nuevos o existentes mediante la consola de Amazon EKS, `eksctl`, AWS CLI, la API de AWS o herramientas de infraestructura como código que incluyen AWS CloudFormation. Los nodos lanzados como parte de un grupo de nodos administrado se etiquetan automáticamente para la detección automática por el Escalador automático de clústeres de Kubernetes. Puede utilizar el grupo de nodos para aplicar etiquetas de Kubernetes a los nodos y actualizarlos en cualquier momento.

No incurre en costos adicionales por usar grupos de nodos administrados por Amazon EKS, solo paga por los recursos de AWS que aprovisione. Estos incluyen instancias de Amazon EC2, volúmenes de Amazon EBS, horas de clúster de Amazon EKS y cualquier otra infraestructura de AWS. No se requieren pagos mínimos ni compromisos iniciales.

Para comenzar a utilizar un grupo de nodos administrado y un clúster de Amazon EKS nuevos, consulte the section called “Creación de un clúster (consola y CLI)”.

Para agregar un grupo de nodos administrados a un clúster existente, consulte the section called “Creación”.

Conceptos de grupos de nodos administrados

- Los grupos de nodos administrados por Amazon EKS crean y administran instancias de Amazon EC2 para usted.
- Todos los nodos administrados se aprovisionan como parte de un grupo de Amazon EC2 Auto Scaling administrado por usted a través de Amazon EKS. Además, todos los recursos, incluidas

las instancias de Amazon EC2 y los grupos de Auto Scaling, se ejecutan dentro de su cuenta de AWS.

- El grupo de Auto Scaling de un grupo de nodos administrados abarca todas las subredes que especifique al crear el grupo.
- Las etiquetas de Amazon EKS administran recursos de grupo de nodos para que estén configurados para usar el Escalador automático de clústeres de Kubernetes.

> **Important**
>
> Si ejecuta una aplicación con estado en varias zonas de disponibilidad basadas en volúmenes de Amazon EBS y utiliza el Escalador automático de clústeres de Kubernetes, debe configurar varios grupos de nodos, cada uno enfocado a una sola zona de disponibilidad. Además, debe habilitar la característica `--balance-similar-node-groups`.

- Puede utilizar una plantilla de lanzamiento personalizada para obtener un mayor nivel de flexibilidad y personalización al implementar nodos administrados. Por ejemplo, puede especificar argumentos `kubelet` adicionales y utilizar una AMI personalizada. Para obtener más información, consulte the section called “Plantillas de inicialización”. Si no usa una plantilla de lanzamiento personalizada al crear un grupo de nodos administrados, hay una plantilla de lanzamiento generada automáticamente. No modifique manualmente esta plantilla generada automáticamente o se producirán errores.
- Amazon EKS sigue el modelo de responsabilidad compartida para CVE y los parches de seguridad en grupos de nodos administrados. Cuando los nodos administrados ejecutan una AMI optimizada para Amazon EKS, Amazon EKS es responsable de crear versiones con parches de la AMI cuando se informa de errores o problemas. Podemos publicar una solución. Sin embargo, es responsable de implementar estas versiones de AMI con parches en los grupos de nodos administrados. Cuando los nodos administrados ejecutan una AMI personalizada, es responsable de crear versiones con parches de la AMI cuando se informa de errores o problemas y, a continuación, implementar la AMI. Para obtener más información, consulte the section called “Actualización”.
- Los grupos de nodos administrados por Amazon EKS se pueden lanzar tanto en subredes públicas como privadas. Si lanza un grupo de nodos administrado en una subred pública el 22 de abril de 2020 o después, la subred debe tener la opción `MapPublicIpOnLaunch` establecida en verdadero para que las instancias puedan unirse a un clúster correctamente. Si la subred pública se creó con `eksctl` o las plantillas de AWS CloudFormation ofrecidas por Amazon EKS el 26 de

marzo de 2020 o después, esta configuración ya está establecida en verdadero. Si las subredes públicas se crearon antes del 26 de marzo de 2020, debe cambiar el valor de forma manual. Para obtener más información, consulte Modificación del atributo de direcciones IPv4 públicas de su subred.

- Al implementar un grupo de nodos administrado en subredes privadas, debe asegurarse de que pueda acceder a Amazon ECR para extraer imágenes de contenedores. Para ello, conecte una puerta de enlace de NAT a la tabla de enrutamiento de la subred o agregue los siguientes puntos de conexión de VPC de AWS PrivateLink:
  - Interfaz de punto de conexión de la API de Amazon ECR: `com.amazonaws.region-code.ecr.api`
  - Interfaz de punto de conexión de la API de registro de Docker de Amazon ECR: `com.amazonaws.region-code.ecr.dkr`
  - Punto de conexión de puerta de enlace de Amazon S3: – `com.amazonaws.region-code.s3`

  Para ver otros servicios y puntos de conexión de uso común, consulte the section called “Clústeres privados”.
- Los grupos de nodos administrados no se pueden implementar en AWS Outposts ni en AWS Wavelength. Los grupos de nodos administrados se pueden crear en zonas locales de AWS. Para obtener más información, consulte the section called “ Zonas locales de AWS”.
- Puede crear varios grupos de nodos administrados dentro de un único clúster. Por ejemplo, puede crear un grupo de nodos con la AMI optimizada de Amazon Linux para Amazon EKS estándar para algunas cargas de trabajo y otro con la variante de GPU para las cargas de trabajo que requieren compatibilidad con GPU.
- Si el grupo de nodos administrado encuentra un error de comprobación de estado de instancia de Amazon EC2, Amazon EKS devuelve un código de error para ayudarlo a diagnosticar el problema. Para obtener más información, consulte the section called “Códigos de error del grupo de nodos administrado”.
- Amazon EKS agrega etiquetas de Kubernetes a instancias de grupos de nodos administrados. Estas etiquetas proporcionadas por Amazon EKS tienen el prefijo `eks.amazonaws.com`.
- Amazon EKS drena nodos automáticamente a través de la API de Kubernetes durante las terminaciones o actualizaciones.
- Los presupuestos de interrupción del pod no se respetan al terminar un nodo con `AZRebalance` o al reducir el número de nodos deseado. Estas acciones intentan expulsar los pods en el nodo. Sin embargo, si estas acciones tardan más de 15 minutos, el nodo se termina independientemente de si todos los pods del nodo están terminados. Para ampliar el período hasta que finalice el

nodo, agregue un enlace de ciclo de vida al grupo de escalado automático. Para obtener más información, consulte Agregar enlaces de enlace de ciclo de vida en la guía de hombre del usuario de Amazon EC2 Auto Scaling.

- Para ejecutar el proceso de drenaje correctamente después de recibir una notificación de interrupción puntual o una notificación de reequilibrio de capacidad, `CapacityRebalance` debe configurarse en `true`.
- La actualización de los grupos de nodos administrados respeta los presupuestos de interrupción de pods que ha establecido para sus pods. Para obtener más información, consulte the section called “Actualizar los detalles del comportamiento”.
- No incurre en costos adicionales por usar grupos de nodos administrados de Amazon EKS. Solo pagará por los recursos de AWS que aprovisione.
- Si desea cifrar volúmenes de Amazon EBS para sus nodos, puede implementarlos mediante una plantilla de lanzamiento. Para implementar nodos administrados con volúmenes cifrados de Amazon EBS sin utilizar una plantilla de lanzamiento, cifre todos los nuevos volúmenes de Amazon EBS creados en su cuenta. Para obtener más información, consulte Cifrado de forma predeterminada en la Guía del usuario de Amazon EC2.

Tipos de capacidad de grupo de nodos administrado

Al crear un grupo de nodos administrado, puede elegir el tipo de capacidad bajo demanda o Spot. Amazon EKS implementa un grupo de nodos administrado con un grupo de Amazon EC2 Auto Scaling que solo contiene instancias de spot bajo demanda o solo instancias de spot de Amazon EC2. Puede programar pods para aplicaciones con tolerancia a errores en grupos de nodos administrados de spot y aplicaciones intolerantes a errores a grupos de nodos bajo demanda dentro de un único clúster de Kubernetes. De forma predeterminada, un grupo de nodos administrado implementa instancias de Amazon EC2 bajo demanda.

Bajo demanda

Con instancias bajo demanda, paga la capacidad informática por segundo, sin compromisos a largo plazo.

De forma predeterminada, si no especifica un Tipo de capacidad, el grupo de nodos administrado se aprovisionará con instancias bajo demanda. En su nombre, un grupo de nodos administrado configura un grupo de Amazon EC2 Auto Scaling con la siguiente configuración aplicada:

- La estrategia de asignación para aprovisionar capacidad bajo demanda se establece en `prioritized`. El grupo de nodos administrado utiliza el orden de los tipos de instancia de la API para determinar qué tipo de instancia se utilizará en primer lugar al cumplir con la capacidad bajo demanda. Por ejemplo, puede especificar tres tipos de instancias en el siguiente orden: `c5.large`, `c4.large` y `c3.large`. Cuando se lanzan las instancias bajo demanda, el grupo de nodos administrado satisface la capacidad bajo demanda, primero en `c5.large`, luego en `c4.large` y luego en `c3.large`. Para obtener más información, consulte Grupo de Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.
- Amazon EKS agrega la siguiente etiqueta de Kubernetes a todos los nodos del grupo de nodos administrado que especifica el tipo de capacidad: `eks.amazonaws.com/capacityType: ON_DEMAND`. Puede utilizar esta etiqueta para programar aplicaciones con estado o intolerantes a errores en nodos bajo demanda.

Spot

Las instancias de spot de Amazon EC2 son una capacidad de Amazon EC2 de reemplazo que ofrece grandes descuentos con respecto a los precios bajo demanda. Las instancias de spot de Amazon EC2 se pueden interrumpir con una notificación previa de dos minutos cuando EC2 necesita recuperar la capacidad. Para obtener más información, consulte Instancias de spot en Guía del usuario de Amazon EC2. Puede configurar un grupo de nodos administrado con instancias de spot de Amazon EC2 para optimizar los costos de los nodos informáticos que se ejecutan en su clúster de Amazon EKS.

Para utilizar instancias de spot dentro de un grupo de nodos administrados, cree uno y establezca el tipo de capacidad como `spot`. Un grupo de nodos administrado configura un grupo de Amazon EC2 Auto Scaling en su nombre con las siguientes prácticas recomendadas de spot aplicadas:

- Para garantizar que los nodos de spot se aprovisionen en los grupos de capacidad de spot óptima, la estrategia de asignación se establece en una de las siguientes:
  - `price-capacity-optimized` (PCO): Al crear nuevos grupos de nodos en un clúster con la versión de Kubernetes `1.28` o una versión posterior, la estrategia de asignación se establece en `price-capacity-optimized`. Sin embargo, la estrategia de asignación no cambiará para los grupos de nodos ya creados con `capacity-optimized` antes de que los grupos de nodos administrados por Amazon EKS comenzaran a admitir PCO.
  - `capacity-optimized` (CO): Al crear nuevos grupos de nodos en un clúster con la versión de Kubernetes `1.27` o una versión anterior, la estrategia de asignación se establece en `capacity-optimized`.

Para aumentar el número de grupos de capacidad de spot disponibles a fin de asignar su capacidad, configure un grupo de nodos administrados para utilizar varios tipos de instancias.

- El reequilibrio de capacidad de spot de Amazon EC2 está habilitado para que Amazon EKS pueda drenar y reequilibrar los nodos de spot de forma sencilla para minimizar la interrupción de la aplicación cuando un nodo de spot corre un riesgo elevado de interrupción. Para obtener más información, consulte Reequilibrio de capacidad de Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.
  - Cuando un nodo de Spot recibe una recomendación de reequilibrio, Amazon EKS automáticamente intenta lanzar un nuevo nodo de Spot de reemplazo.
  - Si llega un aviso de interrupción de spot de dos minutos antes de que el nodo de spot de reemplazo se encuentre en estado `Ready`, Amazon EKS comienza a vaciar el nodo de spot que recibió la recomendación de reequilibrio. Amazon EKS drena el nodo haciendo todo lo posible. En consecuencia, no hay garantía de que Amazon EKS espere a que el nodo de reemplazo se una al clúster antes de agotar el nodo existente.
  - Cuando se inicia un nodo spot de reemplazo con el estado `Ready` en Kubernetes, Amazon EKS acordona y drena el nodo de spot que recibió la recomendación de reequilibrio. El acordonamiento del nodo de spot garantiza que el controlador de servicio no envíe nuevas solicitudes a este nodo de spot. También lo elimina de su lista de nodos de spot activos y en buen estado. Drenar el nodo de spot garantiza que los pods en funcionamiento se expulsen de manera sencilla.
- Amazon EKS agrega la siguiente etiqueta de Kubernetes a todos los nodos del grupo de nodos administrado que especifica el tipo de capacidad: `eks.amazonaws.com/capacityType: SPOT`. Puede utilizar esta etiqueta para programar aplicaciones con tolerancia a errores en nodos de spot.

> **Important**
>
> EC2 emite un aviso de interrupción de spot dos minutos antes de terminar la instancia de spot. Sin embargo, es posible que los pods de nodos de spot no reciban el plazo completo de dos minutos para que se terminen sin problemas. Cuando EC2 emite el aviso, se produce un retraso antes de que Amazon EKS comience a expulsar los pods. Las expulsiones se producen de forma secuencial para proteger el servidor de la API de Kubernetes, por lo que, durante varias recuperaciones simultáneas de spot, algunos pods pueden recibir avisos de expulsión con demora. Los pods se pueden terminar por la fuerza sin recibir señales de terminación, especialmente en los nodos con una alta densidad de pods, durante las recuperaciones simultáneas o cuando se utilizan periodos

de gracia de terminación prolongados. Para las cargas de trabajo de spot, recomendamos diseñar las aplicaciones para que sean tolerantes a las interrupciones, utilizar periodos de gracia de terminación de 30 segundos o menos, evitar los enlaces preStop de larga duración y supervisar las métricas de expulsión de pods para comprender los periodos de gracia reales de los clústeres. En el caso de las cargas de trabajo que requieren una terminación correcta y garantizada, recomendamos utilizar la capacidad bajo demanda como alternativa.

Al decidir si desea implementar un grupo de nodos con capacidad bajo demanda o de spot, debe tener en cuenta las siguientes condiciones:

- Las instancias de spot son una buena opción para aplicaciones sin estado, tolerantes a errores y flexibles. Estos incluyen cargas de trabajo de formación de machine learning y por lotes, ETL de big data como Apache Spark, aplicaciones de procesamiento de colas y puntos de conexión de API sin estado. Dado que spot es capacidad de Amazon EC2 de reemplazo y puede cambiar con el tiempo, le recomendamos que utilice la capacidad de spot para cargas de trabajo tolerantes a interrupciones. Más concretamente, la capacidad de spot es adecuada para cargas de trabajo que pueden tolerar periodos en los que la capacidad requerida no está disponible.
- Recomendamos que utilice la opción bajo demanda para aplicaciones intolerantes a errores. Esto incluye herramientas de administración de clústeres como herramientas operativas y de supervisión, implementaciones que requieren `StatefulSets` y aplicaciones con estado, como bases de datos.
- Para maximizar la disponibilidad de las aplicaciones mientras se utilizan instancias de spot, se recomienda configurar un grupo de nodos administrado de spot para que utilice varios tipos de instancias. Se recomienda aplicar las siguientes reglas al utilizar varios tipos de instancias:
  - Dentro de un grupo de nodos administrado, si está utilizando el escalador automático de clústeres, se recomienda utilizar un conjunto flexible de tipos de instancias con la misma cantidad de vCPU y recursos de memoria. Esto es para garantizar que los nodos del clúster se escalen según lo esperado. Por ejemplo, si necesita cuatro vCPU y 8 GiB de memoria, utilice `c3.xlarge`, `c4.xlarge`, `c5.xlarge`, `c5d.xlarge`, `c5a.xlarge`, `c5n.xlarge` u otros tipos de instancias similares.
  - Para mejorar la disponibilidad de las aplicaciones, recomendamos implementar varios grupos de nodos administrados por spot. Para ello, cada grupo debe utilizar un conjunto flexible de tipos de instancias que tengan los mismos recursos de memoria y vCPU. Por ejemplo, si necesita 4 vCPU y 8 GiB de memoria, le recomendamos que cree un grupo de nodos administrado con

`c3.xlarge`, `c4.xlarge`, `c5.xlarge`, `c5d.xlarge`, `c5a.xlarge`, `c5n.xlarge` u otros tipos de instancias similares, y un segundo grupo de nodos administrado con `m3.xlarge`, `m4.xlarge`, `m5.xlarge`, `m5d.xlarge`, `m5a.xlarge`, `m5n.xlarge` u otros tipos de instancias similares.

- Al implementar el grupo de nodos con el tipo de capacidad spot que utiliza una plantilla de lanzamiento personalizada, utilice la API para pasar varios tipos de instancia. No pase ni un solo tipo de instancia a través de la plantilla de lanzamiento. Para obtener más información sobre cómo implementar un grupo de nodos con una plantilla de lanzamiento, consulte the section called “Plantillas de inicialización”.

Creación de un grupo de nodos administrados para un clúster

En este tema, se describe cómo puede lanzar grupos de nodos administrados de Amazon EKS que se registra en el clúster de Amazon EKS. Una vez que los nodos se hayan unido al clúster, puede implementar aplicaciones de Kubernetes en ellos.

Si es la primera vez que lanza un grupo de nodos administrado de Amazon EKS, le recomendamos que siga una de nuestras guías en Introducción en su lugar. Estas guías proporcionan explicaciones para crear un clúster de Amazon EKS con nodos.

Important

- Los nodos de Amazon EKS son instancias estándar de Amazon EC2. Se le facturará en función de los precios normales de Amazon EC2. Para obtener más información, consulte Precios de Amazon EC2.
- No puede crear nodos administrados en una región de AWS en la que tenga habilitado AWS Outposts o AWS Wavelength. Puede crear nodos autoadministrados en su lugar. Para obtener más información, consulte the section called “Amazon Linux”, the section called “Windows” y the section called “Bottlerocket”. También puede crear un grupo de nodos autoadministrados de Amazon Linux en un Outpost. Para obtener más información, consulte the section called “Nodos”.
- Si no especifica un ID de AMI para el archivo `bootstrap.sh` incluido en Amazon EKS optimizado para Linux o Bottlerocket, los grupos de nodos administrados imponen un número máximo al valor de `maxPods`. Para las instancias con menos de 30 vCPU, el número máximo es `110`. Para las instancias con más de 30 vCPU, el número máximo aumenta a `250`. Estas cifras se basan en los Umbrales de escalabilidad de Kubernetes

y en las configuraciones recomendadas mediante las pruebas internas del equipo de escalabilidad de Amazon EKS. Para obtener más información, consulte la entrada del blog Complemento CNI de Amazon VPC que aumenta los límites de pods por nodo.

- Un clúster existente de Amazon EKS. Para implementar uno, consulte the section called “Creación de un clúster ”.
- Un rol de IAM existente para que lo utilicen los nodos. Para crear uno, consulte the section called “Rol de IAM de nodo”. Si este rol no tiene ninguna de las políticas de la CNI de la VPC, es necesario el rol independiente que se indica a continuación para los pods de la CNI de la VPC.
- (Opcional, pero recomendado) El complemento CNI de Amazon VPC para Kubernetes configurado con su propio rol de IAM que tenga adjunta la política de IAM necesaria. Para obtener más información, consulte the section called “Configuración para IRSA”.
- Familiaridad con las consideraciones que se enumeran en Elección de un tipo de instancia de nodo de Amazon EC2 óptimo. Según el tipo de instancia que elija, es posible que haya requisitos previos adicionales para su clúster y VPC.
- Para agregar un grupo de nodos administrado de Windows, primero debe habilitar la compatibilidad con Windows de su clúster. Para obtener más información, consulte the section called “Habilitación de la compatibilidad con Windows”.

Puede crear un grupo de nodos administrados con cualquiera de las siguientes opciones:

- the section called “`eksctl`”
- the section called “Consola de administración de AWS”

`eksctl`

Crear un grupo de nodos administrados con eksctl

En este procedimiento, se requiere la versión `0.215.0` o posterior de `eksctl`. Puede verificar la versión con el siguiente comando:

```
eksctl version
```

Para obtener instrucciones sobre cómo instalar o actualizar `eksctl`, consulte Instalación en la documentación de `eksctl`.

1. (Opcional) Si la política de IAM administrada AmazonEKS_CNI_Policy se adjunta a su rol de IAM de nodos de Amazon EKS, recomendamos asignarla a un rol de IAM asociado a la cuenta de servicio de `aws-node` de Kubernetes en su lugar. Para obtener más información, consulte the section called “Configuración para IRSA”.
2. Cree un grupo de nodos administrados con una plantilla de lanzamiento personalizada o sin ella. La especificación manual de una plantilla de lanzamiento permite una mayor personalización de un grupo de nodos. Por ejemplo, puede permitir implementar una AMI personalizada o proporcionar argumentos al script `boostrap.sh` en una AMI optimizada para Amazon EKS. Para obtener una lista completa de todas las opciones y valores predeterminados disponibles, ingrese el siguiente comando.

   ```
   eksctl create nodegroup --help
   ```

   En el siguiente comando, reemplace *`my-cluster`* por el nombre del clúster y *`my-mng`* por el nombre del grupo de nodos. El nombre del grupo de nodos no puede tener más de 63 caracteres. Debe empezar por una letra o un dígito, pero también puede incluir guiones y guiones bajos como caracteres no iniciales.

   > **Important**
   >
   > Si no utiliza una plantilla de lanzamiento personalizada cuando crea un grupo de nodos administrados, evite utilizarla más adelante para ese mismo grupo. Si no especificó una plantilla de lanzamiento personalizada, el sistema genera automáticamente una plantilla de lanzamiento que no recomendamos que modifique manualmente. Si se modifica manualmente esta plantilla de lanzamiento generada automáticamente, se podrían producir errores.

Sin una plantilla de lanzamiento

`eksctl` crea una plantilla de lanzamiento predeterminada de Amazon EC2 en su cuenta e implementa el grupo de nodos mediante una plantilla de lanzamiento que crea en función de las opciones que especifique. Antes de especificar un valor para `--node-type`, consulte the section called “Tipos de instancias de Amazon EC2”.

Reemplace *`ami-family`* por una palabra clave permitida. Para obtener más información, consulte Configuración de la familia AMI del nodo en la documentación de `eksctl`. Reemplace *`my-key`* con

el nombre de su par de claves de Amazon EC2 o la clave pública. Esta clave se utiliza para SSH en sus nodos después de que se lancen.

> **Note**
>
> Para Windows, este comando no habilita SSH. En su lugar, asocia el par de claves de Amazon EC2 con la instancia y le permite RDP en la instancia.

Si aún no tiene un par de claves de Amazon EC2, puede crear uno en la Consola de administración de AWS. Para obtener información sobre Linux, consulte Pares de claves e instancias de Amazon EC2 para Linux en la Guía del usuario de Amazon EC2. Para obtener información sobre Windows, consulte Pares de claves e instancias de Amazon EC2 para Windows en la Guía del usuario de Amazon EC2.

Se recomienda bloquear el acceso al pod a IMDS si se cumplen las siguientes condiciones:

- Tiene previsto asignar roles de IAM a todas sus cuentas de servicio de Kubernetes para que los pods solo tengan los permisos mínimos que necesitan.
- Ninguno de los pods del clúster requiere acceso al servicio de metadatos de instancias (IMDS) de Amazon EC2 por otros motivos, como la recuperación de la región de AWS actual.

Para obtener más información, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.

Para bloquear el acceso del pod a IMDS, agregue la opción `--disable-pod-imds` al siguiente comando.

```
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
```

Las instancias pueden asignar de manera opcional un número significativamente mayor de direcciones IP a los pods, asignar direcciones IP a los pods de un bloque de CIDR diferente al de la instancia e implementar en un clúster sin acceso a Internet. Para obtener más información, consulte the section called “Aumento de las direcciones IP”, the section called “Redes personalizadas” y the section called “Clústeres privados” para obtener opciones adicionales que se pueden agregar al comando anterior.

Los grupos de nodos administrados calculan y aplican un único valor para el número máximo de pods que se pueden ejecutar en cada nodo del grupo de nodos, según el tipo de instancia. Si crea un grupo de nodos con distintos tipos de instancias, el valor más pequeño calculado en todos los tipos de instancias se aplica como el número máximo de pods que se pueden ejecutar en cada tipo de instancia del grupo de nodos. Los grupos de nodos administrados calculan el valor mediante el script al que se hace referencia en el Número máximo de pods recomendado por Amazon EKS para cada tipo de instancia de Amazon EC2.

Con una plantilla de lanzamiento

La plantilla de lanzamiento ya debe existir y cumplir con los requisitos especificados en Conceptos básicos de configuración de plantillas de lanzamiento. Se recomienda bloquear el acceso al pod a IMDS si se cumplen las siguientes condiciones:

- Tiene previsto asignar roles de IAM a todas sus cuentas de servicio de Kubernetes para que los pods solo tengan los permisos mínimos que necesitan.
- Ninguno de los pods del clúster requiere acceso al servicio de metadatos de instancias (IMDS) de Amazon EC2 por otros motivos, como la recuperación de la región de AWS actual.

Para obtener más información, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.

Si desea bloquear el acceso de pod a IMDS, especifique la configuración necesaria en la plantilla de lanzamiento.

a. Copie los siguientes contenidos en su dispositivo. Sustituya los valores de ejemplo y, a continuación, ejecute el comando modificado para crear el archivo `eks-nodegroup.yaml`. Varias configuraciones que especifique al implementar sin una plantilla de lanzamiento se mueven a la plantilla de lanzamiento. Si no especifica una `version`, se utiliza la versión de plantilla predeterminada.

```
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
```

Para obtener una lista completa de configuraciones de archivo de config de `eksctl`, consulte Esquema de archivo de Config en la documentación de `eksctl`. Las instancias pueden asignar de manera opcional un número significativamente mayor de direcciones IP a los pods, asignar direcciones IP a los pods de un bloque de CIDR diferente al de la instancia e implementar en un clúster sin acceso de salida a Internet. Para obtener más información, consulte the section called "Aumento de las direcciones IP", the section called "Redes personalizadas" y the section called "Clústeres privados" para obtener opciones adicionales que se pueden agregar al archivo de configuración.

Si no especificó ningún ID de AMI en la plantilla de lanzamiento, los grupos de nodos administrados calculan y aplican un único valor para el número máximo de pods que se pueden ejecutar en cada nodo del grupo de nodos, según el tipo de instancia. Si crea un grupo de nodos con distintos tipos de instancias, el valor más pequeño calculado en todos los tipos de instancias se aplica como el número máximo de pods que se pueden ejecutar en cada tipo de instancia del grupo de nodos. Los grupos de nodos administrados calculan el valor mediante el script al que se hace referencia en el número máximo de pods recomendado por Amazon EKS para cada tipo de instancia de Amazon EC2.

Si especificó un ID de AMI en la plantilla de lanzamiento, debe especificar el número máximo de pods que se pueden ejecutar en cada nodo del grupo de nodos si usa redes personalizadas o quiere aumentar el número de direcciones IP asignadas a la instancia. Para obtener más información, consulte the section called "Número máximo de pods recomendado por Amazon EKS para cada tipo de instancia de Amazon EC2".

b. Implemente el grupo de nodos con el siguiente comando.

```
eksctl create nodegroup --config-file eks-nodegroup.yaml
```

Consola de administración de AWS

Crear un grupo de nodos administrados con la Consola de administración de AWS

1. Espere a que el estado del clúster sea `ACTIVE`. No se puede crear un grupo de nodos administrados para un clúster que aún no está `ACTIVE`.
2. Abra la consola de Amazon EKS.
3. Elija el nombre del clúster en el que desea crear un grupo de nodos administrados.
4. En la pestaña Informática.
5. Elija Agregar grupo de nodos.
6. En la página Configurar grupo de nodos rellene los parámetros en consecuencia y, a continuación, elija Siguiente.
   - Nombre: Ingrese un nombre único para el grupo de nodos administrado. El nombre del grupo de nodos no puede tener más de 63 caracteres. Debe empezar por una letra o un dígito, pero también puede incluir guiones y guiones bajos como caracteres no iniciales.
   - Rol de IAM de nodos: Elija el rol de instancia de nodo que se va a utilizar con su grupo de nodos. Para obtener más información, consulte the section called “Rol de IAM de nodo”.

   > **Important**
   > - No se puede utilizar el mismo rol que se utiliza para crear clústeres.
   > - Es recomendable utilizar un rol que no se esté utilizando actualmente en ningún grupo de nodos autoadministrados. De lo contrario, se utiliza en un nuevo grupo de nodos autoadministrados. Para obtener más información, consulte the section called “Eliminar”.

   - Utilizar la plantilla de lanzamiento: (opcional) seleccione esta opción si desea utilizar una plantilla de lanzamiento existente. Seleccione un Nombre de plantilla de lanzamiento. A continuación, seleccione Versión de plantilla de lanzamiento. Si no selecciona una versión, Amazon EKS utiliza la versión predeterminada de la plantilla. Las plantillas de lanzamiento permiten una mayor personalización del grupo de nodos, como permitir implementar una AMI personalizada, asignar un número significativamente mayor de direcciones IP a los pods, asignar direcciones IP a los pods de un bloque de CIDR diferente al de la instancia e implementar nodos en un clúster sin acceso a Internet saliente. Para obtener más información, consulte the section called “Aumento de las direcciones IP”, the section called “Redes personalizadas” y the section called “Clústeres privados”.

La plantilla de lanzamiento debe cumplir los requisitos de Personalización de nodos administrados con plantillas de lanzamiento. Si no utiliza su propia plantilla de lanzamiento, la API de Amazon EKS crea una plantilla de lanzamiento predeterminada de Amazon EC2 en su cuenta e implementa el grupo de nodos utilizando la plantilla de lanzamiento predeterminada.

Si implementa roles de IAM para cuentas de servicio, asigne los permisos necesarios directamente a todos los pods que requieren acceso a servicios de AWS y ningún pod del clúster requiere acceso a IMDS por otros motivos (como recuperar la región de AWS actual). También puede desactivar el acceso a IMDS para los pods que no utilizan redes de host en una plantilla de lanzamiento. Para obtener más información, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.

- Etiquetas de Kubernetes: (Opcional) puede optar por aplicar etiquetas de Kubernetes a los nodos del grupo de nodos administrado.
- Taints de Kubernetes: (opcional) puede optar por aplicar taints de Kubernetes a los nodos de su grupo de nodos administrados. Las opciones disponibles en el menú Efecto son **`NoSchedule`**, **`NoExecute`** y **`PreferNoSchedule`**. Para obtener más información, consulte the section called “Limitación para que los pods no se programen en nodos específicos”.
- Etiquetas: (Opcional) puede elegir etiquetar su grupo de nodos administrado de Amazon EKS. Estas etiquetas no se propagan a otros recursos del grupo de nodos, como instancias o grupos de escalado automático. Para obtener más información, consulte the section called “Etiquetado de recursos”.

7. En la página Establecer configuración de informática y escalado, rellene los parámetros según corresponda y, a continuación, elija Siguiente.
   - Tipo de AMI: seleccione un tipo de AMI. Si implementa instancias Arm, asegúrese de revisar las consideraciones que se indican en AMI de Arm de Amazon Linux optimizadas para Amazon EKS antes de la implementación.

     Si especificó una plantilla de lanzamiento en la página anterior y especificó una AMI en la plantilla de lanzamiento, no podrá seleccionar un valor. Se muestra el valor de la plantilla. La AMI especificada en la plantilla debe cumplir los requisitos que están en Especificación de una AMI.
   - Tipo de capacidad: Seleccione un tipo de capacidad. Para obtener más información sobre cómo elegir un tipo de capacidad, consulte the section called “Tipos de capacidad de grupo de nodos administrado”. No se pueden mezclar diferentes tipos de capacidad dentro del mismo grupo de nodos. Si desea utilizar ambos tipos de capacidad, cree grupos de nodos independientes,

cada uno con su propia capacidad y tipos de instancias. Consulte Reserva de GPU para grupos de nodos administrados para obtener información sobre el aprovisionamiento y el escalado de nodos de trabajo acelerados por GPU.

- Tipos de instancia: Se especifican uno o más tipos de instancia de forma predeterminada. Para eliminar un tipo de instancia predeterminado, seleccione la casilla X en la parte derecha del tipo de instancia. Elija los tipos de instancia que se van a utilizar en el grupo de nodos administrado. Para obtener más información, consulte the section called “Tipos de instancias de Amazon EC2”.

  La consola muestra un conjunto de tipos de instancia de uso frecuente. Si necesita crear un grupo de nodos administrados con un tipo de instancia que no figura en la lista, utilice `eksctl`, la CLI de AWS, AWS CloudFormation o un SDK para crear el grupo de nodos. Si especificó una plantilla de lanzamiento en la página anterior, no podrá seleccionar un valor porque el tipo de instancia debe especificarse en la plantilla de lanzamiento. Se muestra el valor de la plantilla de lanzamiento. Si seleccionó Spot como Capacity type (Tipo de capacidad), recomendamos especificar varios tipos de instancia para mejorar la disponibilidad.
- Tamaño del disco: ingrese el tamaño del disco (en GiB) que se va a utilizar para el volumen raíz de su nodo.

  Si especificó una plantilla de lanzamiento en la página anterior, no podrá seleccionar un valor porque debe especificarse en la plantilla de lanzamiento.
- Tamaño deseado: Especifique el número actual de nodos que debe mantener el grupo de nodos administrado durante el lanzamiento.

  > **Note**
  >
  > Amazon EKS no escala automáticamente el grupo de nodos de entrada o salida. Sin embargo, puede configurar el Escalador automático de clústeres de Kubernetes para que lo haga. Para obtener más información, consulte Escalador automático de clústeres en AWS.

- Tamaño mínimo: Especifica la cantidad mínima de nodos a los que puede escalar el grupo de nodos administrado.
- Tamaño máximo: Especifica el número máximo de nodos a los que puede escalar el grupo de nodos administrado.
- Configuración de la actualización del grupo de nodos: (Opcional) puede seleccionar el número o el porcentaje de nodos que se actualizarán en paralelo. Estos nodos no estarán disponibles

durante la actualización. En Máximo no disponible, seleccione una de las siguientes opciones y especifique un valor:

- Número: Seleccione y especifique el número de nodos del grupo de nodos que se pueden actualizar en paralelo.
- Porcentaje: Seleccione y especifique el porcentaje de nodos del grupo de nodos que se pueden actualizar en paralelo. Esto es útil si tiene un gran número de nodos en su grupo de nodos.

- Configuración de reparación automática de nodos: (opcional) si activa la casilla Habilitar la reparación automática de nodos, Amazon EKS reemplazará automáticamente los nodos cuando se produzcan problemas detectados. Para obtener más información, consulte the section called "Estado de los nodos".

8. En la página Especificar redes, rellene los parámetros como corresponda y, a continuación, elija Siguiente.
   - Subredes: Elija las subredes en las que lanzar los nodos administrados.

> **Important**
>
> Si ejecuta una aplicación con estado en varias zonas de disponibilidad basadas en volúmenes de Amazon EBS y utiliza el Escalador automático de clústeres de Kubernetes, debe configurar varios grupos de nodos, cada uno enfocado a una sola zona de disponibilidad. Además, debe habilitar la característica `--balance-similar-node-groups`.

> **Important**
>
> - Si elige una subred pública y el clúster solo tiene habilitado el punto de conexión del servidor de API público, la subred debe tener `MapPublicIPOnLaunch` establecido en `true` para que las instancias se unan correctamente a un clúster. Si la subred se creó con `eksctl` o las plantillas de AWS CloudFormation ofrecidas por Amazon EKS el 26 de marzo de 2020 o después, este valor ya está establecido en `true`. Si las subredes se crearon con `eksctl` o las plantillas de AWS CloudFormation antes del 26 de marzo de 2020, debe cambiar el valor de forma manual. Para obtener más información, consulte Modificación del atributo de direcciones IPv4 públicas de su subred.

- Si utiliza una plantilla de lanzamiento y especifica varias interfaces de red, Amazon EC2 no asignará automáticamente una dirección `IPv4` pública, incluso si `MapPublicIpOnLaunch` se establece en `true`. Para que los nodos se unan al clúster en este escenario, debe habilitar el punto de conexión del servidor de API privado del clúster o lanzar nodos en una subred privada con acceso a Internet saliente proporcionado a través de un método alternativo, como una puerta de enlace NAT. A fin de obtener más información, consulte Direcciones IP de instancias de Amazon EC2 en la Guía del usuario de Amazon EC2.

- Configure el acceso SSH a los nodos (opcional). El acceso SSH le permite conectarse a sus instancias y recopilar información de diagnóstico si hay algún problema. Le recomendamos que habilite el acceso remoto cuando cree un grupo de nodos. No puede habilitar el acceso remoto después de crear el grupo de nodos.

  Si eligió utilizar una plantilla de lanzamiento, esta opción no se muestra. Para habilitar el acceso remoto a los nodos, especifique un par de claves en la plantilla de lanzamiento y asegúrese de que el puerto adecuado esté abierto para los nodos de los grupos de seguridad que especifique en la plantilla de lanzamiento. Para obtener más información, consulte the section called “Uso de grupos de seguridad personalizados”.

  > **Note**
  >
  > Para Windows, este comando no habilita SSH. En su lugar, asocia el par de claves de Amazon EC2 con la instancia y le permite RDP en la instancia.

- En Par de claves de SSH seleccione una clave SSH de Amazon EC2 para utilizar. Para obtener información sobre Linux, consulte Pares de claves e instancias de Amazon EC2 para Linux en la Guía del usuario de Amazon EC2. Para obtener información sobre Windows, consulte Pares de claves e instancias de Amazon EC2 para Windows en la Guía del usuario de Amazon EC2. Si eligió utilizar una plantilla de lanzamiento, no puede seleccionar una. Cuando se proporciona una clave SSH de Amazon EC2 para grupos de nodos que utilizan AMI de Bottlerocket, el contenedor administrativo también se habilita. Para obtener más información, consulte Contenedor de administración en GitHub.
- En Allow SSH remote access from (Permitir acceso remoto de SSH desde), si desea limitar el acceso a instancias específicas, seleccione los grupos de seguridad asociados a dichas instancias. Si no se seleccionan grupos de seguridad específicos, se permite el acceso SSH desde cualquier lugar de Internet (`0.0.0.0/0`).

9. En la página Revisar y crear, revise la configuración del grupo de nodos administrados y elija Crear.

   Si los nodos no se unen al clúster, consulte the section called “Los nodos no pueden unirse al clúster” en el capítulo de solución de problemas.

10. Observe el estado de los nodos y espere a que aparezca el estado `Ready`.

```
kubectl get nodes --watch
```

11. (Solo para nodos de GPU) Si ha elegido un tipo de instancia de GPU y una AMI acelerada optimizada para Amazon EKS, deberá aplicar el complemento de dispositivo NVIDIA para Kubernetes como un DaemonSet en el clúster. Reemplace *vX.X.X* con la versión NVIDIA/k8s-device-plugin deseada antes de ejecutar el siguiente comando.

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/
deployments/static/nvidia-device-plugin.yml
```

Instalación de complementos de Kubernetes

Ahora que tiene un clúster de Amazon EKS en funcionamiento con nodos, está listo para comenzar a instalar los complementos de Kubernetes e implementar aplicaciones en su clúster. Los siguientes temas de documentación lo ayudarán a ampliar la funcionalidad de su clúster.

- La entidad principal de IAM que creó el clúster es la única entidad principal que puede hacer llamadas al servidor de API de Kubernetes con `kubectl` o la Consola de administración de AWS. Si desea que otras entidades principales de IAM tengan acceso al clúster, debe agregarlas. Para obtener más información, consulte the section called “Acceso a la API de Kubernetes” y the section called “Permisos necesarios”.
- Se recomienda bloquear el acceso al pod a IMDS si se cumplen las siguientes condiciones:
  - Tiene previsto asignar roles de IAM a todas sus cuentas de servicio de Kubernetes para que los pods solo tengan los permisos mínimos que necesitan.
  - Ninguno de los pods del clúster requiere acceso al servicio de metadatos de instancias (IMDS) de Amazon EC2 por otros motivos, como la recuperación de la región de AWS actual.

  Para obtener más información, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.

- Configure el Escalador automático de clústeres de Kubernetes para ajustar automáticamente el número de nodos en sus grupos de nodos.
- Implemente una aplicación de muestra en su clúster.
- Organice y supervise los recursos del clúster con herramientas importantes para administrar el clúster.

Actualización de un grupo de nodos administrados para un clúster

Cuando inicia una actualización de grupo de nodos administrados, Amazon EKS actualiza los nodos de forma automática al completar los pasos que se indican en Descripción de cada fase de las actualizaciones de los nodos. Si utiliza una AMI optimizada para Amazon EKS, Amazon EKS aplica automáticamente los últimos parches de seguridad y actualizaciones del sistema operativo a los nodos como parte de la versión más reciente de la AMI.

Existen varios escenarios en los que resulta útil actualizar la versión o configuración del grupo de nodos administrado de Amazon EKS:

- Ha actualizado la versión de Kubernetes para su clúster de Amazon EKS y desea actualizar los nodos para que utilicen la misma versión de Kubernetes.
- Hay disponible una nueva versión de la AMI para el grupo de nodos administrados. Para obtener más información acerca de las versiones de AMI, consulte estas secciones:
  - the section called “Obtención de información de la versión”
  - the section called “Bottlerocket”
  - the section called “Obtención de información de la versión”
- Desea ajustar el número mínimo, máximo o deseado de las instancias del grupo de nodos administrados.
- Desea agregar o quitar etiquetas Kubernetes de las instancias del grupo de nodos administrados.
- Desea agregar o quitar etiquetas de AWS del grupo de nodos administrados.
- Debe implementar una nueva versión de una plantilla de lanzamiento con cambios de configuración, como una AMI personalizada actualizada.
- Ha implementado la versión `1.9.0` o posterior del complemento CNI de Amazon VPC, ha habilitado el complemento para la delegación de prefijos y desea nuevas instancias de AWS Nitro System en un grupo de nodos para admitir un número significativamente mayor de pods. Para obtener más información, consulte the section called “Aumento de las direcciones IP”.

- Ha habilitado la delegación de prefijos IP para los nodos de Windows y quiere que las nuevas instancias de AWS Nitro System en un grupo de nodos admitan un número significativamente mayor de pods. Para obtener más información, consulte the section called “Aumento de las direcciones IP”.

Si hay una versión de lanzamiento de AMI más reciente para la versión de Kubernetes del grupo de nodos administrado, puede actualizar la versión de su grupo de nodos para utilizar esa nueva versión de la AMI. De manera similar, si su clúster está ejecutando una versión de Kubernetes más reciente que su grupo de nodos, puede actualizar el grupo de nodos para que utilice la última versión de la AMI que coincida con la versión de Kubernetes del clúster.

Cuando se termina un nodo de un grupo de nodos administrados debido a una operación de escalado o actualización, los pods de ese nodo se drenan primero. Para obtener más información, consulte the section called “Actualizar los detalles del comportamiento”.

Actualizar una versión de grupo de nodos

Puede actualizar una versión del grupo de nodos con una de las siguientes opciones:

- the section called “`eksctl`”
- the section called “Consola de administración de AWS”

La versión a la que se actualiza no puede ser superior a la versión del plano de control.

`eksctl`

Actualice un grupo de nodos administrados mediante **`eksctl`**

Actualice un grupo de nodos administrado a la última versión de AMI de la misma versión de Kubernetes implementada actualmente en los nodos de trabajo con el comando siguiente. Reemplace todos los *`valores de ejemplo`* por sus propios valores.

```
eksctl upgrade nodegroup \
  --name=node-group-name \
  --cluster=my-cluster \
  --region=region-code
```

> **Note**
>
> Si va a actualizar un grupo de nodos que se implementa con una plantilla de lanzamiento a una nueva versión de plantilla de lanzamiento, agregue `--launch-template-version` *`version-number`* al comando anterior. La plantilla de lanzamiento debe cumplir los requisitos descritos en Personalización de nodos administrados con plantillas de lanzamiento. Si la plantilla de lanzamiento incluye una AMI personalizada, la AMI debe cumplir los requisitos en Especificación de una AMI. Cuando actualiza el grupo de nodos a una versión más reciente de la plantilla de lanzamiento, todos los nodos se reciclan para que coincidan con la nueva configuración de la versión de la plantilla de lanzamiento especificada.

No puede actualizar directamente un grupo de nodos que se implementa sin una plantilla de lanzamiento a una nueva versión de la plantilla de lanzamiento. En su lugar, debe implementar un nuevo grupo de nodos mediante la plantilla de lanzamiento para actualizar el grupo de nodos a una nueva versión de la plantilla de lanzamiento.

Puede actualizar un grupo de nodos a la misma versión que la versión de Kubernetes del plano de control. Por ejemplo, si tiene un clúster que ejecuta Kubernetes `1.33`, puede actualizar los nodos que ejecutan Kubernetes `1.32` actualmente a la versión `1.33` con el comando siguiente.

```
eksctl upgrade nodegroup \
  --name=node-group-name \
  --cluster=my-cluster \
  --region=region-code \
  --kubernetes-version=1.33
```

Consola de administración de AWS

Actualice un grupo de nodos administrado mediante la Consola de administración de AWS

1. Abra la consola de Amazon EKS.
2. Elija el clúster que contiene el grupo de nodos que desea actualizar.
3. Si al menos un grupo de nodos tiene una actualización disponible, aparece un cuadro en la parte superior de la página con una notificación sobre la actualización disponible. Si selecciona la pestaña Computación, verá Actualizar ahora en la columna Versión de lanzamiento de la AMI de

la tabla Grupos de nodos para el grupo de nodos que tenga una actualización disponible. Para actualizar el grupo de nodos, elija Update now (Actualizar ahora).

No verá una notificación para los grupos de nodos que se implementaron con una AMI personalizada. Si los nodos se implementan con una AMI personalizada, complete los siguientes pasos para implementar una nueva AMI personalizada actualizada.

a. Cree una nueva versión de su AMI.

b. Cree una nueva versión de la plantilla de lanzamiento con el nuevo ID de AMI.

c. Actualice los nodos a la nueva versión de la plantilla de lanzamiento.

4. En el cuadro de diálogo Update node group version (Actualizar la versión del grupo de nodos), active o desactive las siguientes opciones:

   - Update node group version (Actualizar la versión del grupo de nodos): esta opción no está disponible si ha implementado una AMI personalizada o su AMI optimizada para Amazon EKS está actualmente en la versión más reciente del clúster.
   - Change launch template version (Cambiar la versión de la plantilla de lanzamiento): esta opción no está disponible si el grupo de nodos se implementa sin una plantilla de lanzamiento personalizada. Solo puede actualizar la versión de la plantilla de lanzamiento para un grupo de nodos que se haya implementado con una plantilla de lanzamiento personalizada. Seleccione la versión de la plantilla de lanzamiento a la que desea actualizar el grupo de nodos. Si el grupo de nodos está configurado con una AMI personalizada, la versión que seleccione también debe especificar una AMI. Al actualizar a una versión más reciente de la plantilla de lanzamiento, todos los nodos se reciclan para que coincidan con la nueva configuración de la versión de la plantilla de lanzamiento especificada.

5. En Actualizar estrategia, seleccione una de las siguientes opciones:

   - Actualización continua: esta opción respeta los presupuestos de interrupción del pod para el clúster. Se produce un error en las actualizaciones si hay un problema de presupuesto de interrupción de pod que hace que Amazon EKS no pueda vaciar correctamente los pods que se están ejecutando en este grupo de nodos.
   - Actualización forzada: esta opción no respeta los presupuestos de interrupción del pod. Las actualizaciones se producen independientemente de los problemas presupuestarios de la interrupción del pod al forzar el reinicio de los nodos.

6. Elija Actualizar.

Editar una configuración de grupo de nodos

Puede modificar algunas de las opciones de configuración de un grupo de nodos administrado.

1. Abra la consola de Amazon EKS.
2. Elija el clúster que contiene el grupo de nodos que desea editar.
3. Seleccione la pestaña Compute (Informática).
4. Seleccione el grupo de nodos que desea editar y elija Edit (Editar).
5. (Opcional) En la página Editar grupo de nodos, haga lo siguiente:
   a. Edite la configuración de escalado del grupo de nodos.
      - Tamaño deseado: especifica el número actual de nodos que debe mantener el grupo de nodos administrado.
      - Tamaño mínimo: Especifica la cantidad mínima de nodos a los que puede escalar el grupo de nodos administrado.
      - Tamaño máximo: especifica el número máximo de nodos a los que puede escalar el grupo de nodos administrado. Para obtener el número máximo de nodos admitidos en un grupo de nodos, consulte the section called “Service Quotas”.
   b. (Opcional) Agregue o elimine etiquetas de Kubernetes para los nodos de su grupo de nodos. Las etiquetas que se muestran aquí son solo las que se han aplicado con Amazon EKS. Pueden existir otras etiquetas en los nodos que no se muestran aquí.
   c. (Opcional) Agregue o elimine taints de Kubernetes para los nodos de su grupo de nodos. Las taints agregadas pueden tener el efecto de **`NoSchedule`**, **`NoExecute`** o **`PreferNoSchedule`**. Para obtener más información, consulte the section called “Limitación para que los pods no se programen en nodos específicos”.
   d. (Opcional) Agregue o elimine etiquetas del recurso de su grupo de nodos. Estas etiquetas solo se aplican al grupo de nodos de Amazon EKS. No se propagan a ningún otro recurso, como las subredes o instancias de Amazon EC2 en el grupo de nodos.
   e. (Opcional) Edite la Configuración de la actualización del grupo de nodos. Seleccione el Number (Número) o el Percentage (Porcentaje).
      - Número: seleccione y especifique el número de nodos del grupo de nodos que se pueden actualizar en paralelo. Estos nodos no estarán disponibles durante la actualización.
      - Porcentaje: seleccione y especifique el porcentaje de nodos del grupo de nodos que se pueden actualizar en paralelo. Estos nodos no estarán disponibles durante la actualización. Esto es útil si tiene varios nodos en su grupo de nodos.

f. Cuando haya terminado de editar, elija Guardar cambios.

> **Important**
>
> Al actualizar la configuración del grupo de nodos, cuando se modifica el `NodegroupScalingConfig`, no se respetan los presupuestos de interrupción de pods (PDB). A diferencia del proceso de actualización de grupos de nodos (que vacía los nodos y respeta los PDB durante la fase de actualización), al actualizar la configuración de escalado, los nodos se terminan inmediatamente a través de una llamada de reducción vertical del grupo de escalado automático (ASG). Esto ocurre sin tener en cuenta los PDB, independientemente del tamaño objetivo al que se reduce verticalmente. Esto significa que cuando se reduce el `desiredSize` de un grupo de nodos administrados por Amazon EKS, los pods se expulsan en cuanto se terminan los nodos, sin respetar ningún PDB.

Descripción de cada fase de las actualizaciones de los nodos

La estrategia de actualización del nodo de trabajo administrado de Amazon EKS tiene cuatro fases diferentes descritas en las siguientes secciones.

Fase de configuración

La fase de configuración incluye los siguientes pasos:

1. Crea una nueva versión de plantilla de lanzamiento de Amazon EC2 para el grupo de escalado automático asociado al grupo de nodos. La nueva versión de la plantilla de lanzamiento utiliza la AMI de destino o la versión de plantilla de lanzamiento personalizada para la actualización.
2. El grupo de escalado automático se actualiza para utilizar la versión más reciente de la plantilla de lanzamiento.
3. Determina la cantidad máxima de nodos que se van a actualizar en paralelo mediante la propiedad de `updateConfig` para el grupo de nodos. El máximo no disponible tiene una cuota de 100 nodos. El valor predeterminado es un nodo. Para obtener más información, consulte la propiedad updateConfig en la Referencia de la API de Amazon EKS.

Fase de escalado

Al actualizar los nodos de un grupo de nodos administrados, los nodos actualizados se lanzan en la misma zona de disponibilidad que los que se están actualizando. Para garantizar esta ubicación, utilizamos el reequilibrio de la zona de disponibilidad de Amazon EC2. Para obtener más información, consulte Reequilibrio de la zona de disponibilidad en la Guía del usuario de Amazon EC2 Auto Scaling. Para cumplir este requisito, es posible que lancemos hasta dos instancias por zona de disponibilidad en su grupo de nodos administrado.

La fase de escalado incluye los siguientes pasos:

1. Aumenta el tamaño máximo del grupo de escalado automático y el tamaño deseado en el mayor:
   - Hasta el doble del número de zonas de disponibilidad en la que se implementa el grupo de escalado automático.
   - El máximo no disponible de actualización.

     Por ejemplo, si el grupo de nodos tiene cinco zonas de disponibilidad y `maxUnavailable` como uno solo, el proceso de actualización puede lanzar un máximo de 10 nodos. Sin embargo, cuando `maxUnavailable` es 20 (o cualquier número superior a 10), el proceso puede lanzar 20 nuevos nodos.
2. Después de escalar el grupo de Auto Scaling, comprueba si los nodos que utilizan la configuración más reciente están presentes en el grupo de nodos. Este paso solo se efectúa correctamente cuando cumple estos criterios:
   - Se lanza al menos un nuevo nodo en cada zona de disponibilidad en la que existe el nodo.
   - Todos los nuevos nodos deberían estar en estado `Ready`.
   - Los nuevos nodos deben tener etiquetas aplicadas de Amazon EKS.

     Estas son las etiquetas aplicadas de Amazon EKS en los nodos de trabajo de un grupo de nodos normal:

     - `eks.amazonaws.com/nodegroup-image=$amiName`
     - `eks.amazonaws.com/nodegroup=$nodeGroupName`

     Estas son las etiquetas aplicadas de Amazon EKS en los nodos de trabajo en una plantilla de lanzamiento personalizado o grupo de nodos de AMI:

     - `eks.amazonaws.com/nodegroup-image=$amiName`
     - `eks.amazonaws.com/nodegroup=$nodeGroupName`

- `eks.amazonaws.com/sourceLaunchTemplateId=$launchTemplateId`
- `eks.amazonaws.com/sourceLaunchTemplateVersion=$launchTemplateVersion`

3. Marca los nodos como no programables para evitar programar nuevos pods. También etiqueta los nodos con el `node.kubernetes.io/exclude-from-external-load-balancers=true` para eliminar los nodos antiguos de los equilibradores de carga antes de terminarlos.

Las siguientes son las razones conocidas que llevan a un error de `NodeCreationFailure` en esta fase:

Capacidad insuficiente en la zona de disponibilidad

Existe la posibilidad de que la zona de disponibilidad no tenga la capacidad de los tipos de instancias solicitados. Se recomienda configurar varios tipos de instancias al crear un grupo de nodos administrados.

Límites de instancia de EC2 en su cuenta

Es posible que tenga que aumentar el número de instancias de Amazon EC2 que su cuenta puede ejecutar simultáneamente mediante Service Quotas. Para obtener más información, consulte EC2 Service Quotas en la Guía del usuario de Amazon Elastic Compute Cloud para instancias de Linux.

Datos de usuario personalizados

Los datos de usuario personalizados a veces pueden interrumpir el proceso de arranque. Este escenario puede llevar a que la `kubelet` no se inicie en el nodo o que los nodos no reciban etiquetas de Amazon EKS esperadas en ellos. Para obtener más información, consulte the section called “Especificación de una AMI”.

Cualquier cambio que haga que un nodo no esté en buen estado o no esté preparado

La presión del disco del nodo, la presión de la memoria y condiciones similares pueden provocar que un nodo no funcione en estado `Ready`.

La mayoría de los nodos se arrancan en 15 minutos

Si algún nodo tarda más de 15 minutos en arrancar y unirse al clúster, se agotará el tiempo de espera de la actualización. Este es el tiempo de ejecución total para el arranque de un nodo nuevo, medido desde el momento en que se necesita un nodo nuevo hasta el momento en que se une al clúster. Al actualizar un grupo de nodos administrado, el contador de tiempo se inicia en cuanto aumenta el tamaño del grupo de escalado automático.

Fase de actualización

La fase de actualización se comporta de dos maneras diferentes, en función de la estrategia de actualización. Hay dos estrategias de actualización: predeterminada y mínima.

Recomendamos la estrategia predeterminada en la mayoría de los casos. Crea nuevos nodos antes de terminar los antiguos, de modo que la capacidad disponible se mantiene durante la fase de actualización. La estrategia mínima resulta útil en situaciones en las que los recursos o los costos se ven limitados, por ejemplo, con aceleradores de hardware como las GPU. Consiste en terminar los nodos anteriores antes de crear los nuevos, de modo que la capacidad total nunca aumente más allá de la cantidad configurada.

La estrategia de actualización predeterminada consta de los siguientes pasos:

1. Aumenta la cantidad de nodos (cantidad deseada) en el grupo de escalado automático, lo que provoca que el grupo de nodos cree nodos adicionales.
2. Selecciona aleatoriamente un nodo que necesita una actualización hasta el máximo no disponible configurado para el grupo de nodos.
3. Drena los pods del nodo. Si los pods no salen del nodo en 15 minutos y no hay un indicador de fuerza, la fase de actualización falla con un error `PodEvictionFailure`. Para este escenario, puede aplicar el indicador de fuerza con la solicitud `update-nodegroup-version` para eliminar los pods.
4. Acordona el nodo después de expulsar todos los pods y espera 60 segundos. Esto se hace para que el controlador del servicio no envíe ninguna solicitud nueva a este nodo y elimine este nodo de su lista de nodos activos.
5. Envía una solicitud de terminación al grupo de escalado automático para el nodo acordonado.
6. Repite los pasos anteriores de actualización hasta que no haya nodos en el grupo de nodos que se implementen con la versión anterior de la plantilla de lanzamiento.

La estrategia de actualización mínima consta de los siguientes pasos:

1. Marca todos los nodos del grupo como no programables al inicio, para que el controlador de servicio no envíe nuevas solicitudes a estos nodos.
2. Selecciona aleatoriamente un nodo que necesita una actualización hasta el máximo no disponible configurado para el grupo de nodos.
3. Drena los pods de los nodos seleccionados. Si los pods no salen del nodo en 15 minutos y no hay un indicador de fuerza, la fase de actualización falla con un error `PodEvictionFailure`.

Para este escenario, puede aplicar el indicador de fuerza con la solicitud `update-nodegroup-version` para eliminar los pods.

4. Después de que se haya desalojado cada pod y transcurran 60 segundos, se envía una solicitud de terminación al grupo de escalado automático para los nodos seleccionados. El grupo de escalado automático crea nuevos nodos (en la misma cantidad que los nodos seleccionados) para reemplazar la capacidad faltante.
5. Repite los pasos anteriores de actualización hasta que no haya nodos en el grupo de nodos que se implementen con la versión anterior de la plantilla de lanzamiento.

Errores **`PodEvictionFailure`** durante la fase de actualización

Las siguientes son las razones conocidas que llevan a un error de `PodEvictionFailure` en esta fase:

PDB agresivo

El PDB agresivo se define en el pod o hay varios PDB que apuntan al mismo pod.

Implementación que tolera todas las taints

Una vez expulsado cada pod, se espera que el nodo esté vacío porque el nodo se marcó como taint en los pasos anteriores. Sin embargo, si la implementación tolera todas las taints, es más probable que el nodo no esté vacío, lo que provoca un error en la expulsión del pod.

Fase de reducción vertical

La fase de reducción vertical disminuye el tamaño máximo del grupo de Auto Scaling y el tamaño deseado en uno para volver a los valores antes de que se inicie la actualización.

Si el flujo de trabajo de actualización determina que el escalador automático del clúster realiza el escalado vertical del grupo de nodos durante la fase de reducción vertical del flujo de trabajo, se cierra inmediatamente sin que el grupo de nodos vuelva a su tamaño original.

Personalización de nodos administrados con plantillas de lanzamiento

Para obtener el nivel más alto de personalización, puede implementar nodos administrados con su propia plantilla de lanzamiento según los pasos de esta página. El uso de una plantilla de lanzamiento permite funciones como proporcionar argumentos de arranque durante la

implementación de un nodo (por ejemplo, argumentos de kubelet adicionales), asignar direcciones IP a los pods desde un bloque de CIDR diferente al de la dirección IP asignada al nodo, implementar una AMI o una CNI propias personalizadas en los nodos.

Si proporciona su propia plantilla de lanzamiento al crear por primera vez un grupo de nodos administrado, también tendrá mayor flexibilidad más adelante. Siempre que implemente un grupo de nodos administrado con su propia plantilla de lanzamiento, puede actualizarlo de forma iterativa con una versión diferente de la misma plantilla de lanzamiento. Cuando actualiza el grupo de nodos a una versión diferente de la plantilla de lanzamiento, todos los nodos del grupo se reciclan para que coincidan con la nueva configuración de la versión de la plantilla de lanzamiento especificada.

Los grupos de nodos administrados se implementan siempre con una plantilla de lanzamiento para utilizar con el grupo de Amazon EC2 Auto Scaling. Cuando no proporciona una plantilla de lanzamiento, la API de Amazon EKS crea una en su cuenta de forma automática con los valores predeterminados. Sin embargo, no le recomendamos que modifique las plantillas de lanzamiento generadas automáticamente. Además, los grupos de nodos existentes que no utilizan una plantilla de lanzamiento personalizada no se pueden actualizar directamente. En su lugar, debe crear un nuevo grupo de nodos con una plantilla de lanzamiento personalizada para hacerlo.

Conceptos básicos de configuración de plantillas de lanzamiento

Puede crear una plantilla de lanzamiento de Amazon EC2 Auto Scaling con la Consola de administración de AWS, la AWS CLI o un SDK de AWS. Para obtener más información, consulte Creación de una plantilla de lanzamiento para un grupo de Auto Scaling en la guía del usuario de Amazon EC2 Auto Scaling. Algunas de las opciones de configuración de una plantilla de lanzamiento son similares a las que se utilizan para la configuración de nodos administrados. Al implementar o actualizar un grupo de nodos con una plantilla de lanzamiento, se deben especificar algunas opciones en la configuración del grupo de nodos o en la plantilla de lanzamiento. No especifique un ajuste en ambos lugares. Si existe una configuración donde no debería, las operaciones como la creación o actualización de un grupo de nodos fallarán.

En la siguiente tabla, se enumeran los ajustes prohibidos en una plantilla de lanzamiento. También se enumeran ajustes similares, si hay alguno disponible, que se requieren en la configuración del grupo de nodos administrados. La configuración de la lista es la configuración que aparece en la consola. Pueden tener nombres similares, pero diferentes en la AWS CLI y el SDK.

Plantilla de lanzamiento: opciones prohibidas	Configuración del grupo de nodos de Amazon EKS
Subred en Interfaces de red (Agregar interfaz de red)	Subredes en Configuración de red del grupo de nodos en la página Especificar red
Perfil de instancia de IAM en Detalles avanzados	Rol de IAM del nodo en Configuración del grupo de nodos en la página Configurar grupo de nodos.
Comportamiento de apagado y Detener: comportamiento de hibernación en Detalles avanzados. Mantenga la opción predeterm inada No incluir en la configuración de la plantilla de lanzamiento en la plantilla de lanzamiento para ambas configuraciones.	Sin equivalente. Amazon EKS debe controlar el ciclo de vida de la instancia, no el grupo de escalado automático.

En la siguiente tabla, se enumeran los ajustes prohibidos de una configuración de grupo de nodos administrados. También se enumeran configuraciones similares, si hay alguna disponible, que son necesarias en una plantilla de lanzamiento. La configuración de la lista es la configuración que aparece en la consola. Es posible que tengan nombres similares en la AWS CLI y el SDK.

Configuración del grupo de nodos de Amazon EKS: opciones prohibidas	Plantilla de inicialización
(Solo si especificó una AMI personalizada en una plantilla de lanzamiento) Tipo de AMI en Configuración de computación del grupo de nodos en la página Establecer la configuración informática y de escalado: la consola muestra Se especifica en la plantilla de lanzamiento y el ID de la AMI que se especificó. Si no se especificó nada en Imágenes de aplicaciones y sistema operativo (Imagen de máquina de Amazon) en la plantilla de	Imágenes de aplicaciones y sistema operativo (Imagen de máquina de Amazon) en Contenido de la plantilla de lanzamiento: debe especificar un ID si tiene alguno de los siguientes requisito s: • Uso de una AMI personalizada. Si especific a una AMI que no cumple los requisitos enumerados en Especificación de una AMI, se producirá un error en la implementación del grupo de nodos.

Configuración del grupo de nodos de Amazon EKS: opciones prohibidas	Plantilla de inicialización
lanzamiento, puede seleccionar una AMI en la configuración del grupo de nodos.	• Desea facilitar datos de usuario para proporcionar argumentos al archivo `bootstrap.sh` incluido con una AMI optimizada para Amazon EKS. Puede habilitar sus instancias para asignar un número significativamente mayor de direcciones IP a los pods, asignar direccion es IP a los pods de un bloque de CIDR diferente al de la instancia, habilitar el tiempo de ejecución o implementar un clúster privado sin acceso a Internet saliente. Para obtener más información, consulte los temas siguientes: • Asignación de más direcciones IP a los nodos de Amazon EKS con prefijos • Implementación de pods en subredes alternativas con redes personalizadas • Implementación de clústeres privados con acceso limitado a Internet • Especificación de una AMI
Tamaño del disco en Configuración de computación del grupo de nodos en la página Establecer la configuración de informática y escalado: la consola muestra Especificado en la plantilla de lanzamiento.	Tamaño en Almacenamiento (Volúmenes) (Agregar nuevo volumen). Debe especificarlo en la plantilla de lanzamiento.
Par de claves de SSH en Configuración del grupo de nodos en la página Especificar redes: la consola muestra la clave especificada en la plantilla de lanzamiento o muestra No especific ado en la plantilla de lanzamiento.	Nombre del par de claves en Par de claves (inicio de sesión).

Configuración del grupo de nodos de Amazon EKS: opciones prohibidas	Plantilla de inicialización
No se pueden especificar grupos de seguridad fuente a los que se permita el acceso remoto cuando se utiliza una plantilla de lanzamiento.	Grupos de seguridad en Configuración de red para la instancia o Grupos de seguridad en Interfaces de red (Agregar interfaz de red), pero no ambos. Para obtener más información, consulte the section called “Uso de grupos de seguridad personalizados”.

Note

- Si implementa un grupo de nodos mediante una plantilla de lanzamiento, especifique un tipo de instancia o ninguno en Contenido de la plantilla de lanzamiento en una plantilla de lanzamiento. Si lo desea, puede especificar entre 0 y 20 tipos de instancia para Tipos de instancias en la página Establecer la configuración de informática y escalado de la consola. O bien, puede hacerlo mediante otras herramientas que utilizan la API de Amazon EKS. Si especifica un tipo de instancia en una plantilla de lanzamiento y utiliza esa plantilla de lanzamiento para implementar el grupo de nodos, no podrá especificar ningún tipo de instancia en la consola ni utilizar otras herramientas que utilicen la API de Amazon EKS. Si no especifica un tipo de instancia en una plantilla de lanzamiento, en la consola o si utiliza otras herramientas que utilizan la API de Amazon EKS, se utiliza el tipo de instancia `t3.medium`. Si el grupo de nodos utiliza el tipo de capacidad spot, se recomienda especificar varios tipos de instancias mediante la consola. Para obtener más información, consulte the section called “Tipos de capacidad de grupo de nodos administrado”.
- Si alguno de los contenedores que implementa en el grupo de nodos utiliza el servicio de metadatos de instancia versión 2, asegúrese de establecer la propiedad Límite del salto de respuesta de metadatos en 2 en la plantilla de lanzamiento. Para obtener más información, consulte Metadatos de instancia y datos de usuario en la Guía del usuario de Amazon EC2.
- Las plantillas de lanzamiento no son compatibles con la característica `InstanceRequirements`, que permite seleccionar el tipo de instancia de forma flexible.

Etiquetado de instancias de Amazon EC2

Puede utilizar el parámetro `TagSpecification` de una plantilla de lanzamiento para especificar qué etiquetas se aplicarán a las instancias de Amazon EC2 del grupo de nodos. La entidad IAM que llama a las API `CreateNodegroup` o `UpdateNodegroupVersion` debe tener permisos para `ec2:RunInstances` y `ec2:CreateTags`, y las etiquetas deben agregarse a la plantilla de lanzamiento.

Uso de grupos de seguridad personalizados

Puede utilizar una plantilla de lanzamiento para especificar grupos de seguridad de Amazon EC2 personalizados para aplicar a instancias del grupo de nodos. Esto puede estar en el parámetro de grupos de seguridad de nivel de instancia o como parte de los parámetros de configuración de la interfaz de red. Sin embargo, no se puede crear una plantilla de lanzamiento que especifique el nivel de la instancia y los grupos de seguridad de una interfaz de red. Tenga en cuenta las siguientes condiciones que se aplican al uso de grupos de seguridad personalizados con grupos de nodos administrados:

- Cuando utiliza la Consola de administración de AWS, Amazon EKS solo permite plantillas de lanzamiento con una única especificación de interfaz de red.
- De forma predeterminada, Amazon EKS aplica el grupo de seguridad de clúster a las instancias del grupo de nodos para facilitar la comunicación entre nodos y el plano de control. Si especifica grupos de seguridad personalizados en la plantilla de lanzamiento mediante cualquiera de las opciones mencionadas anteriormente, Amazon EKS no agrega el grupo de seguridad del clúster. Debe asegurarse de que las reglas entrantes y salientes de los grupos de seguridad habiliten la comunicación con el punto de conexión del clúster. Si las reglas del grupo de seguridad son incorrectas, los nodos de trabajo no pueden unirse al clúster. Para obtener más información acerca de las reglas de los grupos de seguridad, consulte the section called “Requisitos del grupo de seguridad”.
- Si necesita acceso SSH a las instancias del grupo de nodos, incluya un grupo de seguridad que permita ese acceso.

Datos de usuario de Amazon EC2

La plantilla de lanzamiento incluye una sección para datos de usuario personalizados. Puede especificar la configuración de su grupo de nodos en esta sección sin crear manualmente AMI

personalizadas individuales. Para obtener más información sobre la configuración disponible para Bottlerocket, consulte Utilización de datos de usuario en GitHub.

Puede proporcionar datos de usuario de Amazon EC2 en su plantilla de lanzamiento mediante `cloud-init` al iniciar sus instancias. Para obtener más información, consulte la documentación de cloud-init. Los datos de usuario se pueden utilizar para realizar operaciones de configuración comunes. Esto incluye las operaciones siguientes:

- Inclusión de usuarios o grupos
- Instalación de paquetes

Los datos de usuario de Amazon EC2 en las plantillas de lanzamiento que se utilizan con grupos de nodos administrados deben estar en el formato de archivo multiparte MIME para las AMI de Amazon Linux y en el formato TOML para las AMI de Bottlerocket. Esto se debe a que los datos de usuario se combinan con los datos de usuario de Amazon EKS necesarios para que los nodos se unan al clúster. No especifique ningún comando en los datos de usuario que inicie o modifique `kubelet`. Esto se realiza como parte de los datos de usuario fusionados por Amazon EKS. Ciertos parámetros de `kubelet`, como establecer etiquetas en nodos, se pueden configurar directamente a través de la API de grupos de nodos administrados.

> **Note**
>
> Para obtener más información sobre la personalización de `kubelet` avanzada, lo que incluye un inicio manual o pasar parámetros de configuración personalizados, consulte the section called “Especificación de una AMI”. Si se especifica un ID de AMI personalizado en una plantilla de lanzamiento, Amazon EKS no fusiona los datos de usuario.

Los siguientes detalles proporcionan más información sobre la sección de datos de usuario.

Datos de usuario de Amazon Linux 2

Puede combinar varios bloques de datos de usuario en un único archivo multiparte MIME. Por ejemplo, puede combinar un boothook de nube que configure el daemon de Docker con un script de shell de datos de usuario que instala un paquete personalizado. Un archivo multiparte MIME consta de los siguientes componentes:

- El tipo de contenido y declaración de límite de partes: – `Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="`

- La declaración de versión de MIME: – `MIME-Version: 1.0`
- Uno o más bloques de datos de usuario, que contienen los siguientes componentes:
  - El límite de apertura, que señala el inicio de un bloque de datos de usuario: `--==MYBOUNDARY==`
  - La declaración de tipo de contenido para el bloque: `Content-Type: text/cloud-config; charset="us-ascii"`. Para obtener más información sobre los tipos de contenido, consulte la documentación de cloud-init.
  - El contenido de los datos de usuario, por ejemplo, una lista de comandos de shell o políticas de `cloud-init`.
  - El límite de cierre, que señala el final del archivo multiparte MIME: `--==MYBOUNDARY==--`

  A continuación, se muestra un ejemplo de un archivo multiparte MIME que puede utilizar para crear el suyo propio.

```
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==--
```

Datos de usuario de Amazon Linux 2023

Amazon Linux 2023 (AL2023) introduce un nuevo proceso de inicialización de nodos `nodeadm` que utiliza un esquema de configuración YAML. Si utiliza grupos de nodos autoadministrados o una AMI con una plantilla de lanzamiento, ahora tendrá que proporcionar metadatos del clúster adicionales de forma explícita cuando cree un nuevo grupo de nodos. A continuación, se muestra un ejemplo de los parámetros mínimos necesarios, en los que ahora se necesitan `apiServerEndpoint`, `certificateAuthority` y el servicio de `cidr`:

```
---
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
```

```
    name: my-cluster
    apiServerEndpoint: https://example.com
    certificateAuthority: Y2VydGlmaWNhdGVBdXRob3JpdHk=
    cidr: 10.100.0.0/16
```

Por lo general, establecerá esta configuración en los datos de usuario, tal y como están o incrustados en un documento MIME de varias partes:

```
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--
```

En AL2, los metadatos de estos parámetros se descubrieron a partir de la llamada a la API `DescribeCluster` de Amazon EKS. Con AL2023, este comportamiento ha cambiado, ya que la llamada a la API adicional corre el riesgo de limitarse durante los escalados verticales de nodos a gran escala. Este cambio no le afecta si utiliza grupos de nodos administrados sin una plantilla de lanzamiento o si utiliza Karpenter. Para obtener más información sobre `certificateAuthority` y el servicio `cidr`, consulte `DescribeCluster` en la Referencia de la API de Amazon EKS.

Este es un ejemplo completo de datos de usuario de AL2023 que combina un script de intérprete de comandos para personalizar el nodo (por ejemplo, instalar paquetes o almacenar previamente en caché las imágenes de contenedores) con la configuración requerida de `nodeadm`. Este ejemplo muestra personalizaciones comunes, como la instalación de paquetes de sistema adicionales, el almacenamiento previo en caché de imágenes de contenedores para mejorar el tiempo de inicio de los pods, la configuración de un proxy HTTP y la definición de marcas de `kubelet` para el etiquetado de nodos

```
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--
```

Datos de usuario de Bottlerocket

Bottlerocket estructura los datos de usuario en formato TOML. Puede facilitar datos de usuario para fusionarlos con los datos de usuario proporcionados por Amazon EKS. Por ejemplo, puede especificar un configuración adicional de `kubelet`.

```
[settings.kubernetes.system-reserved]
cpu = "10m"
memory = "100Mi"
ephemeral-storage= "1Gi"
```

Para obtener más información acerca de la configuración admitida, consulte la documentación de Bottlerocket. Puede configurar etiquetas de nodos y taints en los datos de usuario. Sin embargo, recomendamos que las configure en su grupo de nodos. Amazon EKS aplica estas configuraciones cuando lo hace de este modo.

Cuando se fusionan los datos de usuario, el formato no se conserva, pero el contenido sigue siendo el mismo. La configuración que proporciona en los datos de usuario anula cualquier configuración configurada por Amazon EKS. Entonces, si configura `settings.kubernetes.max-pods` o `settings.kubernetes.cluster-dns-ip`, estos valores de los datos de usuario se aplican a los nodos.

Amazon EKS no admite todos los TOML válidos. A continuación, se muestra una lista de formatos conocidos no compatibles:

- Comillas dentro de las claves citadas: `'quoted "value"' = "value"`
- Comillas escapadas en valores: `str = "I'm a string. \"You can quote me\""`
- Flotantes y enteros mixtos: `numbers = [ 0.1, 0.2, 0.5, 1, 2, 5 ]`
- Tipos mixtos en matrices: `contributors = ["foo@example.com", { name = "Baz", email = "baz@example.com" }]`
- Encabezados entre corchetes con claves citadas: `[foo."bar.baz"]`

Datos de usuario de Windows

Los datos de usuario de Windows utilizan comandos de PowerShell. Al crear un grupo de nodos administrado, los datos de usuario personalizados se combinan con los datos de usuario administrados de Amazon EKS. Sus comandos de PowerShell son lo primero, seguidos de los comandos de datos de usuario administrados, todo dentro de una etiqueta `<powershell></powershell>`.

> **Important**
>
> Al crear grupos de nodos de Windows, Amazon EKS actualiza el `aws-auth ConfigMap` para permitir que los nodos basados en Linux se unan al clúster. El servicio no configura automáticamente permisos para las AMI de Windows. Si utiliza nodos de Windows, deberá gestionar el acceso mediante la API de entrada de acceso o la actualización directa de `aws-auth ConfigMap`. Para obtener más información, consulte the section called “Habilitación de la compatibilidad con Windows”.

> **Note**
>
> Si no se especifica ningún ID de AMI en la plantilla de lanzamiento, no utilice el script Amazon EKS Bootstrap de Windows en los datos de usuario para configurar Amazon EKS.

Los datos de usuario de ejemplo son los siguientes.

```
<powershell>
Write-Host "Running custom user data script"
</powershell>
```

Especificación de una AMI

Si tiene alguno de los siguientes requisitos, especifique un ID de AMI en el campo `ImageId` de la plantilla de lanzamiento. Seleccione el requisito que tiene para obtener información adicional.

Proporcione datos de usuario a fin de pasar argumentos al archivo **`bootstrap.sh`** incluido con una AMI optimizada de Linux o Bottlerocket para Amazon EKS.

Bootstrapping es un término que se utiliza para describir la adición de comandos que se pueden ejecutar cuando se inicia una instancia. Por ejemplo, el arranque permite usar argumentos kubelet adicionales. Puede pasar los argumentos al script `bootstrap.sh` mediante `eksctl` sin especificar una configuración de lanzamiento. O puede hacerlo al especificar la información en la sección de datos de usuario de una plantilla de lanzamiento.

eksctl sin especificar una plantilla de lanzamiento

Cree un archivo llamado *my-nodegroup.yaml* con el siguiente contenido. Reemplace todos los *valores de ejemplo* por sus propios valores. Los argumentos `--apiserver-endpoint`, `--b64-cluster-ca` y `--dns-cluster-ip` son opcionales. Sin embargo, definirlos permite que el script `bootstrap.sh` evite crear una llamada `describeCluster`. Esto resulta útil en configuraciones de clústeres privados o clústeres en los que los nodos se reducen y escalan horizontalmente con frecuencia. Para obtener más información sobre el script `bootstrap.sh`, consulte el archivo bootstrap.sh en GitHub.

- El único argumento requerido es el nombre del clúster (*my-cluster*).
- Para recuperar el ID de una AMI optimizada para `ami-`*1234567890abcdef0* , consulte las siguientes secciones:
  - Recuperación de los ID de AMI de Amazon Linux recomendados
  - Recuperación de los ID de AMI de Bottlerocket recomendados
  - Recuperación de los ID de AMI de Microsoft Windows recomendados
- Para recuperar el *certificate-authority* de su clúster, ejecute el siguiente comando.

```
aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text
 --name my-cluster --region region-code
```

- Para recuperar el *api-server-endpoint* de su clúster, ejecute el siguiente comando.

```
aws eks describe-cluster --query "cluster.endpoint" --output text --name my-
cluster --region region-code
```

- El valor de `--dns-cluster-ip` es su servicio de CIDR con `.10` al final. Para recuperar el *service-cidr* de su clúster, ejecute el siguiente comando. Por ejemplo, si el valor devuelto es `ipv4 10.100.0.0/16`, su valor es *10.100.0.10*.

```
aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr"
 --output text --name my-cluster --region region-code
```

- En este ejemplo proporciona un argumento `kubelet` para establecer un valor de `max-pods` personalizado mediante el script `bootstrap.sh` incluido con la AMI optimizada para Amazon EKS. El nombre del grupo de nodos no puede tener más de 63 caracteres. Debe empezar por una letra o un dígito, pero también puede incluir guiones y guiones bajos como caracteres no iniciales. Para obtener ayuda con la selección de *my-max-pods-value*, consulte the section

called “Número máximo de pods recomendado por Amazon EKS para cada tipo de instancia de Amazon EC2”.

```
---
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
```

Para cada opción de archivo `eksctl config` disponible, consulte Config file schema (Esquema de archivo de configuración) en la documentación de `eksctl`. La utilidad `eksctl` sigue creando una plantilla de lanzamiento para usted y rellena sus datos de usuario con los datos que proporciona en el archivo `config`.

Cree un grupo de nodos con el siguiente comando.

```
eksctl create nodegroup --config-file=my-nodegroup.yaml
```

Datos del usuario en una plantilla de lanzamiento

Especifique la siguiente información en la sección de datos de usuario de la plantilla de lanzamiento. Reemplace todos los *valores de ejemplo* por sus propios valores. Los argumentos `--apiserver-endpoint`, `--b64-cluster-ca` y `--dns-cluster-ip` son

opcionales. Sin embargo, definirlos permite que el script `bootstrap.sh` evite crear una llamada `describeCluster`. Esto resulta útil en configuraciones de clústeres privados o clústeres en los que los nodos se reducen y escalan horizontalmente con frecuencia. Para obtener más información sobre el script `bootstrap.sh`, consulte el archivo bootstrap.sh en GitHub.

- El único argumento requerido es el nombre del clúster (*`my-cluster`*).
- Para recuperar el *`certificate-authority`* de su clúster, ejecute el siguiente comando.

```
aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text
 --name my-cluster --region region-code
```

- Para recuperar el *`api-server-endpoint`* de su clúster, ejecute el siguiente comando.

```
aws eks describe-cluster --query "cluster.endpoint" --output text --name my-
cluster --region region-code
```

- El valor de `--dns-cluster-ip` es su servicio de CIDR con `.10` al final. Para recuperar el *`service-cidr`* de su clúster, ejecute el siguiente comando. Por ejemplo, si el valor devuelto es `ipv4 10.100.0.0/16`, su valor es *`10.100.0.10`*.

```
aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr"
 --output text --name my-cluster --region region-code
```

- En este ejemplo proporciona un argumento `kubelet` para establecer un valor de `max-pods` personalizado mediante el script `bootstrap.sh` incluido con la AMI optimizada para Amazon EKS. Para obtener ayuda con la selección de *`my-max-pods-value`*, consulte the section called “Número máximo de pods recomendado por Amazon EKS para cada tipo de instancia de Amazon EC2”.

```
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==--
```

Proporcione datos de usuario a fin de pasar argumentos al archivo **`Start-EKSBootstrap.ps1`** incluido con una AMI de Windows optimizada para Amazon EKS.

Bootstrapping es un término que se utiliza para describir la adición de comandos que se pueden ejecutar cuando se inicia una instancia. Puede pasar los argumentos al script `Start-EKSBootstrap.ps1` mediante `eksctl` sin especificar una configuración de lanzamiento. O puede hacerlo al especificar la información en la sección de datos de usuario de una plantilla de lanzamiento.

Si desea especificar un ID de AMI de Windows personalizado, tenga en cuenta las siguientes consideraciones:

- Debe utilizar una plantilla de lanzamiento y proporcionar los comandos de arranque necesarios en la sección de datos de usuario. Para recuperar el ID de Windows deseado, puede utilizar la tabla de Creación de nodos con AMI de Windows optimizadas.
- Hay varios límites y condiciones. Por ejemplo, debe agregar `eks:kube-proxy-windows` a su mapa de configuración de IAM Authenticator de AWS. Para obtener más información, consulte the section called “Límites y condiciones al especificar un ID de AMI”.

Especifique la siguiente información en la sección de datos de usuario de la plantilla de lanzamiento. Reemplace todos los *`valores de ejemplo`* por sus propios valores. Los argumentos `-APIServerEndpoint`, `-Base64ClusterCA` y `-DNSClusterIP` son opcionales. Sin embargo, definirlos permite que el script `Start-EKSBootstrap.ps1` evite crear una llamada `describeCluster`.

- El único argumento requerido es el nombre del clúster (*`my-cluster`*).
- Para recuperar el *`certificate-authority`* de su clúster, ejecute el siguiente comando.

```
aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text --
name my-cluster --region region-code
```

- Para recuperar el *`api-server-endpoint`* de su clúster, ejecute el siguiente comando.

```
aws eks describe-cluster --query "cluster.endpoint" --output text --name my-cluster
 --region region-code
```

- El valor de `--dns-cluster-ip` es su servicio de CIDR con `.10` al final. Para recuperar el *`service-cidr`* de su clúster, ejecute el siguiente comando. Por ejemplo, si el valor devuelto es `ipv4 10.100.0.0/16`, su valor es *`10.100.0.10`*.

```
aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr" --
output text --name my-cluster --region region-code
```

- Para obtener argumentos adicionales, consulte the section called “Parámetros de configuración del script de arranque”.

> **Note**
>
> Si utiliza un CIDR de servicio personalizado, debe especificarlo con el parámetro `-ServiceCIDR`. De lo contrario, se producirá un error en la resolución de DNS del clúster para pods.

```
<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>
```

Ejecute una AMI personalizada debido a requisitos específicos de seguridad, conformidad o políticas internas

Para obtener más información, consulte Imágenes de máquina de Amazon (AMI) en la Guía del usuario de Amazon EC2. La especificación de compilación de la AMI de Amazon EKS contiene recursos y scripts de configuración para crear una AMI personalizada de Amazon EKS basada en Amazon Linux. Para obtener más información, consulte Especificación de compilación de AMI de Amazon EKS en GitHub. Para crear AMI personalizadas instaladas con otros sistemas operativos, consulte AMI personalizadas de ejemplo de Amazon EKS en GitHub.

No puede usar referencias de parámetros dinámicos para los ID de AMI en las plantillas de lanzamiento empleadas con grupos de nodos administrados.

> **Important**
>
> Al especificar una AMI, Amazon EKS no combina ningún dato de usuario. Más bien, usted es responsable de suministrar el comando `bootstrap` requerido para que los nodos se unan al clúster. Si los nodos no se unen al clúster, las acciones de Amazon EKS `CreateNodegroup` y `UpdateNodegroupVersion` también fallan.

Límites y condiciones al especificar un ID de AMI

A continuación se detallan los límites y las condiciones que implica especificar un ID de AMI con grupos de nodos administrados:

- Debe crear un nuevo grupo de nodos para cambiar entre especificar un ID de AMI en una plantilla de lanzamiento y no especificar un ID de AMI.
- No se le notifica en la consola cuando hay disponible una versión más reciente de AMI. Para actualizar el grupo de nodos a una versión de AMI más reciente, debe crear una nueva versión de la plantilla de lanzamiento con un ID de AMI actualizado. A continuación, debe actualizar el grupo de nodos con la nueva versión de plantilla de lanzamiento.
- Los siguientes campos no se pueden establecer en la API si especifica un ID de AMI:
  - `amiType`
  - `releaseVersion`
  - `version`
- Todas las `taints` configuradas en la API se aplican de manera asíncrona si se especifica un ID de AMI. Para aplicar taints antes de que un nodo se una al clúster, se deben transferir las taints a `kubelet` en sus datos de usuario mediante la marca `--register-with-taints` de la línea de comandos. Para obtener más información, consulte kubelet en la documentación de Kubernetes.
- Al especificar un ID de AMI personalizado para los grupos de nodos administrados de Windows, agregue `eks:kube-proxy-windows` al mapa de configuración del Autenticador de AWS IAM. Esto es necesario para que el DNS funcione correctamente.

  a. Abra el mapa de configuración de IAM Authenticator de AWS para editarlo.

  ```
  kubectl edit -n kube-system cm aws-auth
  ```

b. Agregue esta entrada a la lista de `groups` debajo de cada uno de los `rolearn` asociados con nodos de Windows. El mapa de configuración debe tener un aspecto similar al de aws-auth-cm-windows.yaml.

```
- eks:kube-proxy-windows
```

c. Guarde el archivo y salga del editor de texto.

- Para cualquier AMI que utilice una plantilla de lanzamiento personalizada, el valor predeterminado de `HttpPutResponseHopLimit` para los grupos de nodos administrados se establece en 2.

Eliminación de un grupo de nodos administrado de un clúster

En este tema se describe cómo puede eliminar un grupo de nodos administrado de Amazon EKS. Al eliminar un grupo de nodos administrados, Amazon EKS establece primero el tamaño mínimo, máximo y deseado del grupo de Auto Scaling en cero. Esto hace que el grupo de nodos se reduzca verticalmente.

Antes de terminar cada instancia, Amazon EKS envía una señal para vaciar ese nodo. Durante el proceso de vaciado, Kubernetes hace lo siguiente para cada pod del nodo: ejecuta cualquier enlace de ciclo de vida `preStop` configurado, envía señales `SIGTERM` a los contenedores y, a continuación, espera a que `terminationGracePeriodSeconds` se apague correctamente. Si el nodo no se ha drenado después de cinco minutos, Amazon EKS permite que el escalado automático continúe con la terminación forzada de la instancia. Una vez terminadas todas las instancias, se elimina el grupo de escalado automático.

> **Important**
>
> Si elimina un grupo de nodos administrado que utiliza un rol de IAM de un nodo que no se emplea en ningún otro grupo de nodos administrado en el clúster, el rol se quitará del `ConfigMap` de `aws-auth`. Si algún grupo de nodos autoadministrados del clúster utiliza el mismo rol de IAM del nodo, los nodos autoadministrados adoptarán el estado `NotReady`. Además, también se interrumpe la operación del clúster. Para añadir una asignación para el rol que está utilizando solo para los grupos de nodos autoadministrados, consulte the section called “Creación de entradas de acceso”, si la versión de la plataforma de su clúster es al menos la versión mínima que aparece en la sección de requisitos previos de Concesión de acceso a los usuarios de IAM a las entradas de acceso de Kubernetes con EKS. Si la versión de la plataforma es anterior a la versión mínima requerida para las entradas de

acceso, puede volver a añadir la entrada al `ConfigMap` de `aws-auth`. Para obtener más información, ingrese `eksctl create iamidentitymapping --help` en su terminal.

Se puede eliminar un grupo de nodos administrados con:

- the section called “`eksctl`”
- the section called “Consola de administración de AWS”
- the section called “AWS CLI”

`eksctl`

Eliminar un grupo de nodos administrados con **`eksctl`**

Escriba el siguiente comando. Reemplace cada `<example value>` con valores propios.

```
eksctl delete nodegroup \
  --cluster <my-cluster> \
  --name <my-mng> \
  --region <region-code>
```

Para obtener más opciones, consulte Eliminar y drenar grupos de nodos en la documentación `eksctl`.

Consola de administración de AWS

Eliminar un grupo de nodos administrados con la Consola de administración de AWS

1. Abra la consola de Amazon EKS.
2. En la página Clústeres, elija el clúster que contiene el grupo de nodos que desea eliminar.
3. En la página del clúster, seleccione la pestaña Computar.
4. En la sección de Node Groups (Grupos de nodos), elija el grupo de nodos que desea eliminar. A continuación, elija Eliminar.
5. En el cuadro de diálogo de confirmación Eliminar grupo de nodos, introduzca el nombre del grupo de nodos. A continuación, elija Eliminar.

AWS CLI

Eliminar un grupo de nodos administrados con la CLI de AWS

1. Escriba el siguiente comando. Reemplace cada `<example value>` con valores propios.

```
aws eks delete-nodegroup \
  --cluster-name <my-cluster> \
  --nodegroup-name <my-mng> \
  --region <region-code>
```

2. Si se ha establecido `cli_pager=` en la configuración de la CLI, use las teclas de flecha del teclado para desplazarse por el resultado de la respuesta. Pulse la tecla q cuando termine.

   Para obtener más opciones, consulte el comando `delete-nodegroup` en la Referencia de los comandos de la CLI de AWS.

Mantenimiento de nodos por cuenta propia con nodos autoadministrados

Un clúster contiene uno o varios nodos de Amazon EC2 en los que están programados los pods. Los nodos de Amazon EKS se ejecutan en su cuenta de AWS y se conectan con el plano de control del clúster a través del punto de conexión del servidor de la API del clúster. Se le factura por ellos en función de los precios de Amazon EC2. Para obtener más información, consulte Precios de Amazon EC2.

Un clúster puede contener varios grupos de nodos. Cada grupo de nodos contiene uno o varios nodos que se implementan en un grupo de Amazon EC2 Auto Scaling. El tipo de instancia de los nodos del grupo puede variar; por ejemplo, cuando se utiliza la selección del tipo de instancia basada en atributos con Karpenter. Todas las instancias de un grupo de nodos deben utilizar el rol de IAM del nodo de Amazon EKS.

Amazon EKS proporciona imágenes de máquina de Amazon (AMI) especializadas que se denominan AMI optimizadas para Amazon EKS. Las AMI están configuradas para funcionar con Amazon EKS. Sus componentes incluyen `containerd`, `kubelet` y el Autenticador de AWS IAM. Las AMI también contienen un script de arranque especializado que permite detectar el plano de control del clúster y conectarse automáticamente a él.

Si restringe el acceso al punto de conexión público del clúster mediante bloques de CIDR, recomendamos habilitar también el acceso al punto de conexión privado. Esto permite que los nodos se puedan comunicar con el clúster. Si el punto de conexión privado no está habilitado, los bloques de CIDR que especifique para el acceso público deben incluir los orígenes de salida de su VPC. Para obtener más información, consulte the section called “Acceso al punto de conexión del clúster”.

Para agregar nodos autoadministrados a su clúster de Amazon EKS, consulte los temas siguientes. Si lanza de forma manual nodos autoadministrados, debe agregar la siguiente etiqueta a cada nodo y asegurarse de que `<cluster-name>` coincida con el clúster. Para obtener más información, consulte Cómo agregar etiquetas a un recurso individual y eliminarlas de él. Si sigue los pasos de las siguientes guías, se agregará automáticamente la etiqueta necesaria a los nodos.

Clave	Valor
`kubernetes.io/cluster/<cluster-name>`	`owned`

> **Important**
>
> Las etiquetas del Servicio de metadatos de instancias (IMDS) de Amazon EC2 no son compatibles con los nodos de EKS. Cuando las etiquetas de metadatos de instancia están habilitadas, se impide el uso de barras diagonales (“/”) en los valores de las etiquetas. Esta limitación puede provocar errores en el lanzamiento de las instancias, especialmente cuando se utilizan herramientas de administración de nodos como Karpenter o el Escalador automático de clústeres, ya que estos servicios dependen de etiquetas que contienen barras diagonales para funcionar correctamente.

Para obtener más información sobre los nodos desde un punto de vista general de Kubernetes, consulte Nodos en la documentación de Kubernetes.

Temas

- Creación de nodos autoadministrados de Amazon Linux
- Creación de nodos de Bottlerocket autoadministrados
- Creación de nodos autoadministrados de Microsoft Windows
- Creación de nodos autoadministrados de Linux para Ubuntu

- Actualización de los nodos autoadministrados para un clúster

Creación de nodos autoadministrados de Amazon Linux

En este tema, se describe cómo puede lanzar grupos de escalado automático de nodos de Linux que se registrará con el clúster de Amazon EKS. Una vez que los nodos se hayan unido al clúster, puede implementar aplicaciones de Kubernetes en ellos. También puede lanzar nodos de Amazon Linux autoadministrados con `eksctl` o la Consola de administración de AWS. Si necesita lanzar nodos en AWS Outposts, consulte the section called “Nodos”.

- Un clúster existente de Amazon EKS. Para implementar uno, consulte the section called “Creación de un clúster ”. Si tiene subredes en la región de AWS, donde están habilitadas las subredes AWS Outposts, AWS Wavelength o AWS Local Zones, estas no se deben haber pasado al crear su clúster.
- Un rol de IAM existente para que lo utilicen los nodos. Para crear uno, consulte the section called “Rol de IAM de nodo”. Si este rol no tiene ninguna de las políticas de la CNI de la VPC, es necesario el rol independiente que se indica a continuación para los pods de la CNI de la VPC.
- (Opcional, pero recomendado) El complemento CNI de Amazon VPC para Kubernetes configurado con su propio rol de IAM que tenga adjunta la política de IAM necesaria. Para obtener más información, consulte the section called “Configuración para IRSA”.
- Familiaridad con las consideraciones que se enumeran en Elección de un tipo de instancia de nodo de Amazon EC2 óptimo. Según el tipo de instancia que elija, es posible que haya requisitos previos adicionales para su clúster y VPC.

Puede lanzar nodos de Linux autoadministrados con una de las siguientes opciones:

- the section called “`eksctl`”
- the section called “Consola de administración de AWS”

`eksctl`

Lanzamiento de nodos de Linux autoadministrados mediante **`eksctl`**

1. Instale la versión `0.215.0` o posterior de la herramienta de línea de comandos de `eksctl` instalada en su dispositivo o AWS CloudShell. Para instalar o actualizar `eksctl`, consulte la sección de Instalación en la documentación de `eksctl`.

2. (Opcional) Si la política de IAM administrada AmazonEKS_CNI_Policy se adjunta a su rol de IAM de nodos de Amazon EKS, recomendamos asignarla a un rol de IAM asociado a la cuenta de servicio de `aws-node` de Kubernetes en su lugar. Para obtener más información, consulte the section called “Configuración para IRSA”.
3. El siguiente comando crea un grupo de nodos en un clúster existente. Sustituya *`al-nodes`* por un nombre para su grupo de nodos. El nombre del grupo de nodos no puede tener más de 63 caracteres. Debe empezar por una letra o un dígito, pero también puede incluir guiones y guiones bajos como caracteres no iniciales. Reemplace *`my-cluster`* por el nombre de su clúster. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster. Sustituya los *`valores de ejemplo`* restantes por sus propios valores. Los nodos se crean de forma predeterminada con la misma versión de Kubernetes que el plano de control.

   Antes de elegir un valor de `--node-type`, consulte Elección de un tipo de instancia de nodo de Amazon EC2 óptimo.

   Reemplace *`my-key`* con el nombre de su par de claves de Amazon EC2 o la clave pública. Esta clave se utiliza para SSH en sus nodos después de que se lancen. Si aún no tiene un par de claves de Amazon EC2, puede crear uno en la Consola de administración de AWS. Para obtener más información, consulte Pares de claves de Amazon EC2 en la Guía del usuario de Amazon EC2.

   Cree el grupo de nodos con el siguiente comando.

   > **Important**
   >
   > Si desea implementar un grupo de nodos en las subredes de AWS Outposts, Wavelength o zonas locales, existen consideraciones adicionales:
   >
   > - Las subredes no deben haberse transferido al crear el clúster.
   > - Debe crear el grupo de nodos con un archivo de configuración, que especifique las subredes y `volumeType: gp2`. Para obtener más información, consulte Crear un grupo de nodos a partir de un archivo de Config y el Esquema de archivo de configuración en la documentación de `eksctl`.

```
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
```

Para implementar un grupo de nodos que:

- pueda asignar un número significativamente mayor de direcciones IP a pods que la configuración predeterminada, consulte the section called “Aumento de las direcciones IP”.
- pueda asignar direcciones `IPv4` a pods de un bloque de CIDR diferente que el de la instancia, consulte the section called “Redes personalizadas”.
- pueda asignar direcciones `IPv6` a los pods y los servicios, consulte the section called “IPv6”.
- no tenga acceso saliente a internet, consulte the section called “Clústeres privados”.

  Para obtener una lista completa de todas las opciones y valores predeterminados disponibles, ingrese el siguiente comando.

  ```
  eksctl create nodegroup --help
  ```

  Si los nodos no se unen al clúster, consulte the section called “Los nodos no pueden unirse al clúster” en el capítulo de solución de problemas.

  Un ejemplo de salida sería el siguiente. Se generan varias líneas mientras se crean los nodos. Una de las últimas líneas de salida es la siguiente línea de ejemplo.

  ```
  [#]  created 1 nodegroup(s) in cluster "my-cluster"
  ```

4. (Opcional) Implemente una aplicación de muestra para probar el clúster y los nodos de Linux.
5. Se recomienda bloquear el acceso al pod a IMDS si se cumplen las siguientes condiciones:
   - Tiene previsto asignar roles de IAM a todas sus cuentas de servicio de Kubernetes para que los pods solo tengan los permisos mínimos que necesitan.

- Ninguno de los pods del clúster requiere acceso al servicio de metadatos de instancias (IMDS) de Amazon EC2 por otros motivos, como la recuperación de la región de AWS actual.

Para obtener más información, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.

Consola de administración de AWS

Paso 1: lanzamiento de nodos de Linux autoadministrados mediante la Consola de administración de AWS

1. Descargue la versión más reciente de la plantilla de AWS CloudFormation.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2025-11-26/
amazon-eks-nodegroup.yaml
```

2. Espere a que el estado del clúster sea `ACTIVE`. Si lanza los nodos antes de que el clúster esté activo, los nodos no pueden registrarse con el clúster y tendrá que volver a lanzarlos.
3. Abra la AWS consola de CloudFormation.
4. Seleccione Create stack (Crear pila) y, a continuación, seleccione With new resources (standard) (Con nuevos recursos [estándar]).
5. Para Especificar plantilla, seleccione Cargar un archivo de plantilla y, a continuación, elija Elegir archivo.
6. Edite el archivo `amazon-eks-nodegroup.yaml` que ha descargado.
7. Seleccione Siguiente.
8. En la página Especificar detalles de la pila, ingrese los siguientes parámetros según corresponda y luego seleccione Siguiente:
   - Nombre de pila: elija un nombre para su pila de AWS CloudFormation. Por ejemplo, puede llamarla *`my-cluster-nodes`*. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster.
   - ClusterName: ingrese el nombre que usó al crear el clúster de Amazon EKS. Este nombre debe coincidir con el nombre del clúster o los nodos no se podrán unir al clúster.
   - ClusterControlPlaneSecurityGroup: elija el valor de SecurityGroups de la salida de AWS CloudFormation que generó al crear su VPC.

En los siguientes pasos, se muestra una operación para recuperar el grupo aplicable.

a. Abra la consola de Amazon EKS.

b. Elija el nombre del clúster.

c. Elija la pestaña Redes.

d. Use el valor de Grupo de seguridad adicional como referencia al realizar una selección en la lista desplegable ClusterControlPlaneSecurityGroup.

- ApiServerEndpoint: ingrese el punto de conexión del servidor de API para el clúster de EKS. Puede consultarlo en la sección Detalles de la consola de Clústeres de EKS.
- CertificateAuthorityData: ingrese los datos de la autoridad de certificación codificados en base64, que también se encuentran en la sección Detalles de la consola de Clústeres de EKS.
- ServiceCidr: ingrese el rango CIDR que se usó para asignar las direcciones IP a los servicios de Kubernetes en el clúster. Se encuentra en la pestaña Redes de la consola de Clústeres de EKS.
- AuthenticationMode: para seleccionar el modo de autenticación que se usa en el clúster de EKS, consulte la pestaña de acceso en la consola de Clústeres de EKS.
- NodeGroupName: escriba un nombre para el grupo de nodos. Este nombre se puede utilizar más adelante para identificar el grupo de nodos de escalado automático que se crea para los nodos. El nombre del grupo de nodos no puede tener más de 63 caracteres. Debe empezar por una letra o un dígito, pero también puede incluir guiones y guiones bajos como caracteres no iniciales.
- NodeAutoScalingGroupMinSize: ingrese el número mínimo de nodos al que se puede reducir horizontalmente el grupo de escalado automático de nodos.
- NodeAutoScalingGroupDesiredCapacity: escriba el número deseado de nodos que desea escalar cuando se crea la pila.
- NodeAutoScalingGroupMaxSize: ingrese el número máximo de nodos que pueda alcanzar el grupo de Auto Scaling de nodos.
- NodeInstanceType: elija un tipo de instancia para los nodos. Para obtener más información, consulte the section called “Tipos de instancias de Amazon EC2”.
- NodeImageIdSSMParam: se rellena previamente con el parámetro de Amazon EC2 Systems Manager de una AMI de Amazon Linux 2023 optimizada recientemente para Amazon EKS para una versión de Kubernetes variable. Para utilizar otra versión secundaria de Kubernetes compatible con Amazon EKS, reemplace *1.XX* por una versión admitida diferente. Recomendamos especificar la misma versión de Kubernetes que el clúster.

También puede sustituir *amazon-linux-2023* por un tipo de AMI diferente. Para obtener más información, consulte the section called “Obtención de los ID más recientes”.

> **Note**
>
> Las AMI de los nodos de Amazon EKS se basan en Amazon Linux. Puede realizar un seguimiento de los eventos de seguridad o privacidad de Amazon Linux 2023 en el Centro de seguridad de Amazon Linux o suscribirse a la fuente RSS asociada. Los eventos de seguridad y privacidad incluyen información general del problema, qué paquetes están afectados y cómo actualizar las instancias para corregir el problema.

- NodeImageId: (opcional) si utiliza una AMI personalizada propia (en lugar de una AMI optimizada para Amazon EKS), escriba un ID de AMI de nodo para la región de AWS. Si especifica un valor aquí, anula cualquier valor del campo NodeImageIdSSMParam.
- NodeVolumeSize: especifique un tamaño de volumen raíz para los nodos en GiB.
- NodeVolumeType: especifique un tipo de volumen raíz para sus nodos.
- KeyName: ingrese el nombre de un par de claves SSH de Amazon EC2 que pueda utilizar para conectar mediante SSH con los nodos después de haberlos lanzado. Si aún no tiene un par de claves de Amazon EC2, puede crear uno en la Consola de administración de AWS. Para obtener más información, consulte Pares de claves de Amazon EC2 en la Guía del usuario de Amazon EC2.
- VpcId: ingrese el ID de la VPC que ha creado.
- Subredes: elija las subredes que creó para la VPC. Si creó la VPC siguiendo los pasos que se describen en Creación de una Amazon VPC para su clúster de Amazon EKS, especifique solo las subredes privadas en la VPC en las que desea lanzar los nodos. Puede ver qué subredes son privadas abriendo cada enlace de subred desde la pestaña Redes de su clúster.

> **Important**
>
> - Si alguna de las subredes es pública, debe tener habilitada la configuración de asignación automática de direcciones IP públicas. Si la configuración no está habilitada para la subred pública, los nodos que implemente en dicha subred pública no tendrán asignada una dirección IP pública y no podrán comunicarse con el clúster u otros servicios de AWS. Si la subred se implementó antes del 26 de marzo de 2020 mediante cualquiera de las plantillas de VPC de AWS CloudFormation de Amazon EKS o mediante `eksctl`, la asignación automática de direcciones IP públicas se

deshabilitará en las subredes públicas. Para obtener información acerca de cómo habilitar la asignación de direcciones IP públicas en una subred, consulte Modificación del atributo de direcciones IPv4 públicas de su subred. Si el nodo se implementa en una subred privada, podrá comunicarse con el clúster y otros servicios de AWS a través de una puerta de enlace NAT.

- Si las subredes no tienen acceso a Internet, asegúrese de que conoce las consideraciones y los pasos adicionales en Implementación de clústeres privados con acceso limitado a Internet.
- Si selecciona las subredes de AWS Outposts, Wavelength o zonas locales, las subredes no se deben haber pasado cuando creó el clúster.

9. Seleccione las opciones que desee en la página Configurar las opciones de pila y, a continuación, elija Siguiente.
10. Seleccione la casilla de verificación a la izquierda de Reconozco que AWS podría crear recursos de IAM y, luego, seleccione Crear pila.
11. Una vez completada la creación de la pila, selecciónela en la consola y elija Salidas. Si utiliza los modos de autenticación `EKS API` o `EKS API and ConfigMap`, este es el último paso.
12. Si utiliza el modo de autenticación `ConfigMap`, registre el valor de NodeInstanceRole correspondiente al grupo de nodos que se creó.

Paso 2: habilitación para que los nodos se unan al clúster

> **Note**
>
> Los dos pasos siguientes solo son necesarios si se utiliza el modo de autenticación ConfigMap en el clúster de EKS. Además, si ha lanzado nodos dentro de una VPC privada sin acceso a Internet saliente, asegúrese de activar los nodos para que se unan al clúster desde dentro de la VPC.

1. Verifique si ya tiene el `ConfigMap` de `aws-auth`.

```
kubectl describe configmap -n kube-system aws-auth
```

2. Si se le muestra un `ConfigMap` de `aws-auth`, actualícelo según sea necesario.

   a. Abra el icono `ConfigMap` para editar.

```
kubectl edit -n kube-system configmap/aws-auth
```

b. Añada una nueva entrada de `mapRoles` según sea necesario. Establezca el valor de `rolearn` en el valor de NodeInstanceRole que registró en el procedimiento anterior.

```
[...]
data:
  mapRoles: |
    - rolearn: <ARN of instance role (not instance profile)>
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
[...]
```

c. Guarde el archivo y salga del editor de texto.

3. Si recibe un error que indica "`Error from server (NotFound): configmaps "aws-auth" not found`, aplique el `ConfigMap` bursátil.

a. Descargue el mapa de configuración.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/
aws-auth-cm.yaml
```

b. En el archivo `aws-auth-cm.yaml`, establezca el valor de `rolearn` al valor NodeInstanceRole que ha registrado en el procedimiento anterior. Puede hacerlo con un editor de texto o al reemplazar *`my-node-instance-role`* y ejecutar el siguiente comando:

```
sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-instance-
role|' aws-auth-cm.yaml
```

c. Aplique la configuración. Este comando puede tardar varios minutos en finalizar.

```
kubectl apply -f aws-auth-cm.yaml
```

4. Observe el estado de los nodos y espere a que aparezca el estado `Ready`.

```
kubectl get nodes --watch
```

Ingrese `Ctrl+C` para obtener un símbolo del intérprete de comandos.

> **Note**
>
> Si recibe cualquier error de tipo de recurso o autorización, consulte the section called "Acceso denegado o no autorizado (`kubectl`)" en el tema de solución de problemas.

Si los nodos no se unen al clúster, consulte the section called "Los nodos no pueden unirse al clúster" en el capítulo de solución de problemas.

5. (Solo para nodos de GPU) Si ha elegido un tipo de instancia de GPU y la AMI acelerada optimizada para Amazon EKS, debe aplicar el complemento de dispositivo NVIDIA para Kubernetes como un DaemonSet en el clúster. Reemplace *`vX.X.X`* con la versión NVIDIA/k8s-device-plugin deseada antes de ejecutar el siguiente comando.

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/
deployments/static/nvidia-device-plugin.yml
```

Paso 3: acciones adicionales

1. (Opcional) Implemente una aplicación de muestra para probar el clúster y los nodos de Linux.
2. (Opcional) Si la política de IAM administrada AmazonEKS_CNI_Policy (si tiene un clúster `IPv4`) o la política *`AmazonEKS_CNI_IPv6_Policy`* (que haya creado si tiene un clúster `IPv6`) están adjuntas a su rol de IAM de nodos en Amazon EKS, le recomendamos asignarlas a un rol de IAM que asocie a la cuenta de servicio de `aws-node` de Kubernetes como alternativa. Para obtener más información, consulte the section called "Configuración para IRSA".
3. Se recomienda bloquear el acceso al pod a IMDS si se cumplen las siguientes condiciones:
   - Tiene previsto asignar roles de IAM a todas sus cuentas de servicio de Kubernetes para que los pods solo tengan los permisos mínimos que necesitan.
   - Ninguno de los pods del clúster requiere acceso al servicio de metadatos de instancias (IMDS) de Amazon EC2 por otros motivos, como la recuperación de la región de AWS actual.

   Para obtener más información, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.

Creación de nodos de Bottlerocket autoadministrados

> **Note**
>
> Los grupos de nodos administrados podrían ofrecer algunas ventajas para su caso de uso. Para obtener más información, consulte the section called “Grupos de nodos administrados”.

En este tema, se describe cómo puede lanzar un grupo de escalado automático de nodos de Bottlerocket que se registrará con el clúster de Amazon EKS. Bottlerocket es un sistema operativo de código abierto basado en Linux de AWS que puede utilizar para ejecutar contenedores en máquinas virtuales o hosts bare metal. Una vez que los nodos se hayan unido al clúster, puede implementar aplicaciones de Kubernetes en ellos. Para obtener más información acerca de Bottlerocket, consulte Uso de una AMI de Bottlerocket con Amazon EKS en GitHub y Compatibilidad con AMI personalizada en la documentación de `eksctl`.

Para obtener información sobre actualizaciones en contexto, consulte Operador de actualización de Bottlerocket en GitHub.

> **Important**
>
> - Los nodos de Amazon EKS son instancias estándar de Amazon EC2 y se les facturarán conforme a los precios ordinarios de las instancias de Amazon EC2. Para obtener más información, consulte Precios de Amazon EC2.
> - Puede lanzar nodos de Bottlerocket en clústeres extendidos de Amazon EKS en AWS Outposts, pero no puede lanzarlos en clústeres locales en AWS Outposts. Para obtener más información, consulte Amazon EKS en AWS Outposts.
> - Puede implementar en instancias de Amazon EC2 con procesadores x86 o Arm. Sin embargo, no puede implementar en instancias que tienen chips Inferentia.
> - Bottlerocket es compatible con AWS CloudFormation. Sin embargo, no existe ninguna plantilla oficial de CloudFormation que pueda copiarse para implementar nodos de Bottlerocket para Amazon EKS.
> - Las imágenes de Bottlerocket no vienen con un servidor SSH ni un intérprete de comandos. Puede utilizar métodos de acceso fuera de banda para permitir que SSH habilite el contenedor de administración y superar algunos pasos de configuración de

arranque con datos de usuario. Para obtener más información, consulte estas secciones en bottlerocket README.md en GitHub:

- Exploration (Exploración
- Contenedor de administrador
- Configuración de Kubernetes

En este procedimiento, se requiere la versión `0.215.0` o posterior de `eksctl`. Puede verificar la versión con el siguiente comando:

```
eksctl version
```

Para obtener instrucciones acerca de cómo instalar o actualizar `eksctl`, consulte Instalación en la documentación de `eksctl`. NOTA: Este procedimiento solo funciona en los clústeres que se crearon con `eksctl`.

1. Copie los siguientes contenidos en su dispositivo. Reemplace *`my-cluster`* por el nombre de su clúster. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster. Reemplace *`ng-bottlerocket`* por un nombre para su grupo de nodos. El nombre del grupo de nodos no puede tener más de 63 caracteres. Debe empezar por una letra o un dígito, pero también puede incluir guiones y guiones bajos como caracteres no iniciales. Para implementar en instancias Arm, reemplace *`m5.large`* por un tipo de instancia Arm. Sustituya *`my-ec2-keypair-name`* por el nombre de un par de claves SSH de Amazon EC2 que pueda utilizar para conectar mediante SSH con los nodos después de haberlos lanzado. Si aún no tiene un par de claves de Amazon EC2, puede crear uno en la Consola de administración de AWS. Para obtener más información, consulte Pares de claves de Amazon EC2 en la Guía del usuario de Amazon EC2. Sustituya todos los valores de ejemplo restantes por sus propios valores. Una vez que haya llevado a cabo las sustituciones, ejecute el comando modificado para crear el archivo `bottlerocket.yaml`.

   Si especifica un tipo de instancia Arm de Amazon EC2, revise las consideraciones en AMI de Amazon Linux optimizada para Amazon EKS Arm antes de llevar a cabo la implementación. Para ver instrucciones sobre cómo implementar mediante una AMI personalizada, consulte Creación de Bottlerocket en GitHub y Compatibilidad con AMI personalizada en la documentación de `eksctl`. Para implementar un grupo de nodos administrados, implemente una AMI personalizada mediante

el uso de una plantilla de lanzamiento. Para obtener más información, consulte the section called "Plantillas de inicialización".

> **Important**
>
> Para implementar un grupo de nodos en las subredes de AWS Outposts, AWS Wavelength o zonas locales de AWS, no pase las subredes de AWS Outposts, AWS Wavelength o Zonas locales de AWS al crear el clúster. Debe especificar las subredes en el siguiente ejemplo. Para obtener más información, consulte Crear un grupo de nodos a partir de un archivo de Config y el Esquema de archivo de configuración en la documentación de `eksctl`. Reemplace *`region-code`* por la región de AWS en la que se encuentra el clúster.

```
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. Implemente los nodos con el siguiente comando.

```
eksctl create nodegroup --config-file=bottlerocket.yaml
```

   Un ejemplo de salida sería el siguiente.

   Se generan varias líneas mientras se crean los nodos. Una de las últimas líneas de salida es la siguiente línea de ejemplo.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Opcional) Cree un volumen persistente de Kubernetes en un nodo de Bottlerocket mediante el Complemento CSI de Amazon EBS. El controlador predeterminado de Amazon EBS se basa en herramientas del sistema de archivos que no están incluidas en Bottlerocket. Para obtener información adicional acerca de cómo crear una clase de almacenamiento mediante un controlador, consulte the section called “Amazon EBS”.

4. (Opcional) De forma predeterminada, `kube-proxy` establece el parámetro del kernel `nf_conntrack_max` en un valor predeterminado que puede diferir de lo que Bottlerocket establece inicialmente en el arranque. Para mantener la configuración predeterminada de Bottlerocket, edite la configuración de `kube-proxy` con el siguiente comando.

```
kubectl edit -n kube-system daemonset kube-proxy
```

   Agregue `--conntrack-max-per-core` y `--conntrack-min` a los argumentos `kube-proxy` que se encuentran en el siguiente ejemplo. Una configuración de `0` implica que no hay cambios.

```
      containers:
      - command:
        - kube-proxy
        - --v=2
        - --config=/var/lib/kube-proxy-config/config
        - --conntrack-max-per-core=0
        - --conntrack-min=0
```

5. (Opcional) Implemente una aplicación de muestra para probar los nodos de Bottlerocket.

6. Se recomienda bloquear el acceso al pod a IMDS si se cumplen las siguientes condiciones:

- Tiene previsto asignar roles de IAM a todas sus cuentas de servicio de Kubernetes para que los pods solo tengan los permisos mínimos que necesitan.
- Ninguno de los pods del clúster requiere acceso al servicio de metadatos de instancias (IMDS) de Amazon EC2 por otros motivos, como la recuperación de la región de AWS actual.

Para obtener más información, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.

Creación de nodos autoadministrados de Microsoft Windows

Este tema describe cómo lanzar un grupo de Auto Scaling de nodos de Windows que se registrará con el clúster de Amazon EKS. Una vez que los nodos se hayan unido al clúster, puede implementar aplicaciones de Kubernetes en ellos.

> **Important**
>
> - Los nodos de Amazon EKS son instancias estándar de Amazon EC2 y se le facturarán conforme a los precios ordinarios de las instancias de Amazon EC2. Para obtener más información, consulte Precios de Amazon EC2.
> - Puede lanzar nodos de Windows en clústeres extendidos de Amazon EKS en AWS Outposts, pero no puede lanzarlos en clústeres locales en AWS Outposts. Para obtener más información, consulte Amazon EKS en AWS Outposts.

Habilite la compatibilidad con Windows para su clúster Recomendamos que revise las consideraciones importantes antes de lanzar un grupo de nodos de Windows. Para obtener más información, consulte the section called “Habilitación de la compatibilidad con Windows”.

Puede lanzar nodos de Windows autoadministrados con una de las siguientes opciones:

- the section called “`eksctl`”
- the section called “Consola de administración de AWS”

`eksctl`

Lanzamiento de nodos de Windows autoadministrados mediante **`eksctl`**

En este procedimiento, se presupone que ha instalado `eksctl` y que su versión de `eksctl` es al menos `0.215.0`. Puede verificar la versión con el siguiente comando.

```
eksctl version
```

Para obtener instrucciones sobre cómo instalar o actualizar `eksctl`, consulte Instalación en la documentación de `eksctl`.

> **Note**
>
> Este procedimiento solo es válido para los clústeres que se crearon con `eksctl`.

1. (Opcional) Si la política de IAM administrada AmazonEKS_CNI_Policy (si tiene un clúster `IPv4`) o la política *`AmazonEKS_CNI_IPv6_Policy`* (que haya creado si tiene un clúster `IPv6`) están adjuntas a su rol de IAM de nodos en Amazon EKS, le recomendamos asignarlas, en cambio, a un rol de IAM que asocie a la cuenta de servicio de `aws-node` de Kubernetes. Para obtener más información, consulte the section called “Configuración para IRSA”.
2. En este procedimiento, se presupone que dispone de un clúster existente. Si aún no tiene un clúster de Amazon EKS ni un grupo de nodos de Amazon Linux al cual agregarle un grupo de nodos de Windows, le recomendamos que siga la the section called “Creación de un clúster (eksctl)”. En esta guía se proporciona una explicación completa para crear un clúster de Amazon EKS con nodos de Amazon Linux.

   Cree el grupo de nodos con el siguiente comando. Reemplace *`region-code`* por la región de AWS en la que se encuentra el clúster. Sustituya *`my-cluster`* por el nombre del clúster. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster. Reemplace *`ng-windows`* por un nombre para su grupo de nodos. El nombre del grupo de nodos no puede tener más de 63 caracteres. Debe empezar por una letra o un dígito, pero también puede incluir guiones y guiones bajos como caracteres no iniciales. Puede sustituir *`2019`* por `2022` para usar Windows Server 2022. Sustituya el resto de los valores de ejemplo por sus propios valores.

> **Important**
>
> Para implementar un grupo de nodos en las subredes de AWS Outposts, AWS Wavelength o zonas locales de AWS, no pase las subredes de AWS Outposts, Wavelength o zonas locales al crear el clúster. Cree el grupo de nodos con un archivo de configuración, que especifique las subredes de AWS Outposts, Wavelength o Local Zone. Para obtener más información, consulte Crear un grupo de nodos a partir de un archivo de Config y el Esquema de archivo de configuración en la documentación de `eksctl`.

```
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**
>
> - Si los nodos no se unen al clúster, consulte the section called “Los nodos no pueden unirse al clúster” en la Guía de solución de problemas.
> - Para ver las opciones disponibles para los comandos `eksctl`, ingrese el siguiente comando.
>
>   ```
>   eksctl command -help
>   ```

Un ejemplo de salida sería el siguiente. Se generan varias líneas mientras se crean los nodos. Una de las últimas líneas de salida es la siguiente línea de ejemplo.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Opcional) Implemente una aplicación de muestra para probar el clúster y los nodos de Windows.

4. Se recomienda bloquear el acceso al pod a IMDS si se cumplen las siguientes condiciones:
   - Tiene previsto asignar roles de IAM a todas sus cuentas de servicio de Kubernetes para que los pods solo tengan los permisos mínimos que necesitan.
   - Ninguno de los pods del clúster requiere acceso al servicio de metadatos de instancias (IMDS) de Amazon EC2 por otros motivos, como la recuperación de la región de AWS actual.

   Para obtener más información, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.

Consola de administración de AWS

Requisitos previos

- Un clúster de Amazon EKS existente y un grupo de nodos de Linux. Si no tiene estos recursos, recomendamos que siga una de nuestras guías de para crearlos Introducción. En las guías se describe cómo crear un clúster de Amazon EKS con nodos de Linux.
- Una VPC y un grupo de seguridad existentes que cumplen los requisitos para un clúster de Amazon EKS. Para obtener más información, consulte the section called “Requisitos de VPC y subred” y the section called “Requisitos del grupo de seguridad”. Las guías de Introducción crean una VPC que cumple los requisitos. Si lo desea, también puede seguir Creación de una Amazon VPC para su clúster de Amazon EKS para crear una manualmente.
- Un clúster de Amazon EKS existente que utiliza una VPC y un grupo de seguridad que cumplen los requisitos de un clúster de Amazon EKS. Para obtener más información, consulte the section called “Creación de un clúster ”. Si tiene subredes en la región de AWS, donde están habilitadas las subredes AWS Outposts, AWS Wavelength o zonas locales de AWS, estas no se deben haber pasado al crear el clúster.

Paso 1: lanzar nodos de Windows autoadministrados mediante la Consola de administración de AWS

1. Espere a que el estado del clúster sea `ACTIVE`. Si lanza los nodos antes de que el clúster esté activo, los nodos no pueden registrarse con el clúster y debe volver a lanzarlos.
2. Abra la consola de AWS CloudFormation
3. Elija Crear pila.
4. En Especificar plantilla, seleccione URL de Amazon S3.
5. Copie la siguiente URL y péguela en la URL de Amazon S3.

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2023-02-09/amazon-eks-windows-nodegroup.yaml
```

6. Seleccione Siguiente dos veces.
7. En la página Creación rápida de pila, ingrese los siguientes parámetros según corresponda:
   - Nombre de pila: elija un nombre para su pila de AWS CloudFormation. Por ejemplo, puede llamarla `my-cluster-nodes`.
   - ClusterName: ingrese el nombre que usó al crear el clúster de Amazon EKS.

     > **Important**
     >
     > El nombre debe coincidir exactamente con el nombre que utilizó en el Paso 1: Creación del clúster de Amazon EKS. De lo contrario, los nodos no podrán unirse al clúster.

   - ClusterControlPlaneSecurityGroup: elija el grupo de seguridad de la salida de AWS CloudFormation que generó al crear la VPC. En los pasos siguientes se muestra un método para recuperar el grupo aplicable.
     a. Abra la consola de Amazon EKS.
     b. Elija el nombre del clúster.
     c. Elija la pestaña Redes.
     d. Use el valor de Grupo de seguridad adicional como referencia al realizar una selección en la lista desplegable ClusterControlPlaneSecurityGroup.
   - NodeGroupName: escriba un nombre para el grupo de nodos. Este nombre se puede utilizar más adelante para identificar el grupo de nodos de escalado automático que se crea para los nodos. El nombre del grupo de nodos no puede tener más de 63 caracteres. Debe empezar por una letra o un dígito, pero también puede incluir guiones y guiones bajos como caracteres no iniciales.
   - NodeAutoScalingGroupMinSize: ingrese el número mínimo de nodos al que se puede reducir horizontalmente el grupo de escalado automático de nodos.
   - NodeAutoScalingGroupDesiredCapacity: escriba el número deseado de nodos que desea escalar cuando se crea la pila.
   - NodeAutoScalingGroupMaxSize: ingrese el número máximo de nodos que pueda alcanzar el grupo de Auto Scaling de nodos.

- NodeInstanceType: elija un tipo de instancia para los nodos. Para obtener más información, consulte the section called “Tipos de instancias de Amazon EC2”.

> **Note**
>
> Los tipos de instancias compatibles con la versión más reciente del complemento CNI de Amazon VPC para Kubernetes se muestran en vpc_ip_resource_limit.go en GitHub. Es posible que tenga que actualizar la versión de CNI para utilizar los tipos de instancia admitidos más recientes. Para obtener más información, consulte the section called “CNI de Amazon VPC”.

- NodeImageIdSSMParam: contiene un valor especificado previamente, que es el parámetro de Amazon EC2 Systems Manager del ID de la AMI de Windows Core optimizada para Amazon EKS más reciente recomendada. Para utilizar la versión completa de Windows, sustituya `Core` por `Full`.
- NodeImageId: (opcional) si utiliza una AMI personalizada propia (en lugar de una AMI optimizada para Amazon EKS), escriba un ID de AMI de nodo para la región de AWS. Si especifica un valor para este campo, anula cualquier valor del campo NodeImageIdSSMParam.
- NodeVolumeSize: especifique un tamaño de volumen raíz para los nodos en GiB.
- KeyName: ingrese el nombre de un par de claves SSH de Amazon EC2 que pueda utilizar para conectar mediante SSH con los nodos después de haberlos lanzado. Si aún no tiene un par de claves de Amazon EC2, puede crear uno en la Consola de administración de AWS. Para obtener más información, consulte Pares de claves de Amazon EC2 en la Guía del usuario de Amazon EC2.

> **Note**
>
> Si no proporciona un par de claves aquí, se produce un error al crear la pila de AWS CloudFormation.

- BootstrapArguments: especifique los argumentos opcionales que se van a pasar al script de arranque del nodo, como los argumentos de `kubelet` adicionales mediante `-KubeletExtraArgs`.
- DisableIMDSv1: cada nodo admite de forma predeterminada la versión 1 (IMDSv1) e IMDSv2 del servicio de metadatos de la instancia. Puede desactivar IMDSv1. Para evitar que los nodos

y pods futuros del grupo de nodos utilicen MDSv1, defina DisableIMDSv1 en verdadero. Para obtener más información, consulte Configurar el servicio de metadatos de instancia.

- VpcId: seleccione el ID de la VPC que creó.
- NodeSecurityGroups: seleccione el grupo de seguridad que se creó para su grupo de nodos de Linux cuando creó la VPC. Si sus nodos de Linux tienen más de un grupo de seguridad adjuntado a ellos, especifíquelos a todos aquí. Esto, por ejemplo, si el grupo de nodos Linux se creó con `eksctl`.
- Subredes: elija las subredes que creó. Si creó la VPC siguiendo los pasos que se describen en Creación de Amazon VPC para su clúster de Amazon EKS, especifique solo las subredes privadas en la VPC en las que desea lanzar los nodos.

> **Important**
>
> - Si alguna de las subredes es pública, debe tener habilitada la configuración de asignación automática de direcciones IP públicas. Si la configuración no está habilitada para la subred pública, los nodos que implemente en dicha subred pública no tendrán asignada una dirección IP pública y no podrán comunicarse con el clúster u otros servicios de AWS. Si la subred se implementó antes del 26 de marzo de 2020 mediante cualquiera de las plantillas de VPC de AWS CloudFormation de Amazon EKS o mediante `eksctl`, la asignación automática de direcciones IP públicas se deshabilitará en las subredes públicas. Para obtener información acerca de cómo habilitar la asignación de direcciones IP públicas en una subred, consulte Modificación del atributo de direcciones IPv4 públicas de su subred. Si el nodo se implementa en una subred privada, podrá comunicarse con el clúster y otros servicios de AWS a través de una puerta de enlace NAT.
> - Si las subredes no tienen acceso a Internet, asegúrese de que conoce las consideraciones y los pasos adicionales en Implementación de clústeres privados con acceso limitado a Internet.
> - Si selecciona las subredes de AWS Outposts, Wavelength o zonas locales, las subredes no se deben haber pasado cuando creó el clúster.

8. Confirme que la pila pueda crear recursos de IAM y, a continuación, seleccione Crear pila.
9. Una vez completada la creación de la pila, selecciónela en la consola y elija Salidas.
10. Anote el valor de NodeInstanceRoles correspondiente al grupo de nodos creado. Lo necesitará al configurar los nodos de Windows para Amazon EKS.

Paso 2: habilitación para que los nodos se unan al clúster

1. Verifique si ya tiene el `ConfigMap` de `aws-auth`.

```
kubectl describe configmap -n kube-system aws-auth
```

2. Si se le muestra un `ConfigMap` de `aws-auth`, actualícelo según sea necesario.
   a. Abra el icono `ConfigMap` para editar.

```
kubectl edit -n kube-system configmap/aws-auth
```

   b. Añada nuevas entradas de `mapRoles` según sea necesario. Establezca los valores de `rolearn` en los valores de NodeInstanceRole que registró en los procedimientos anteriores.

```
[...]
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. Guarde el archivo y salga del editor de texto.
3. Si recibe un error que indica “`Error from server (NotFound): configmaps "aws-auth" not found`, aplique el `ConfigMap` bursátil.
   a. Descargue el mapa de configuración.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/
aws-auth-cm-windows.yaml
```

   b. En el archivo `aws-auth-cm-windows.yaml`, establezca los valores de `rolearn` a los valores NodeInstanceRole que ha registrado en el procedimiento anterior. Puede hacerlo con un editor de texto o reemplazando los valores de ejemplo y ejecutando el siguiente comando:

```
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**
>
> - No modifique ninguna otra línea de este archivo.
> - No utilice el mismo rol de IAM para los nodos de Windows y Linux.

c. Aplique la configuración. Este comando podría tardar varios minutos en finalizar.

```
kubectl apply -f aws-auth-cm-windows.yaml
```

4. Observe el estado de los nodos y espere a que aparezca el estado `Ready`.

```
kubectl get nodes --watch
```

Ingrese `Ctrl+C` para obtener un símbolo del intérprete de comandos.

> **Note**
>
> Si recibe cualquier error de tipo de recurso o autorización, consulte the section called "Acceso denegado o no autorizado (`kubectl`)" en el tema de solución de problemas.

Si los nodos no se unen al clúster, consulte the section called "Los nodos no pueden unirse al clúster" en el capítulo de solución de problemas.

Paso 3: acciones adicionales

1. (Opcional) Implemente una aplicación de muestra para probar el clúster y los nodos de Windows.
2. (Opcional) Si la política de IAM administrada AmazonEKS_CNI_Policy (si tiene un clúster `IPv4`) o la política *`AmazonEKS_CNI_IPv6_Policy`* (que haya creado si tiene un clúster `IPv6`) están adjuntas a su rol de IAM de nodos en Amazon EKS, le recomendamos asignarlas a un rol de IAM que asocie a la cuenta de servicio de `aws-node` de Kubernetes como alternativa. Para obtener más información, consulte the section called "Configuración para IRSA".

3. Se recomienda bloquear el acceso al pod a IMDS si se cumplen las siguientes condiciones:
   - Tiene previsto asignar roles de IAM a todas sus cuentas de servicio de Kubernetes para que los pods solo tengan los permisos mínimos que necesitan.
   - Ninguno de los pods del clúster requiere acceso al servicio de metadatos de instancias (IMDS) de Amazon EC2 por otros motivos, como la recuperación de la región de AWS actual.

   Para obtener más información, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.

Creación de nodos autoadministrados de Linux para Ubuntu

> **Note**
>
> Los grupos de nodos administrados podrían ofrecer algunas ventajas para su caso de uso. Para obtener más información, consulte the section called “Grupos de nodos administrados”.

En este tema, se describe cómo lanzar grupos de escalado automático de nodos de Ubuntu en Amazon Elastic Kubernetes Service (EKS) o Ubuntu Pro en Amazon Elastic Kubernetes Service (EKS) que se registran con el clúster de Amazon EKS. Ubuntu y Ubuntu Pro para EKS se basan en Ubuntu Minimal LTS oficial, incluyen el kernel de AWS personalizado que se desarrolla junto con AWS y se han creado específicamente para EKS. Ubuntu Pro agrega una cobertura de seguridad adicional, ya que es compatible con los periodos de soporte ampliados de EKS, el parche activo del kernel, la compatibilidad con FIPS y la capacidad de ejecutar contenedores Pro ilimitados.

Una vez que los nodos se hayan unido al clúster, puede implementar aplicaciones de contenedores en ellos. Para obtener más información, consulte la documentación sobre Ubuntu en AWS y la Compatibilidad con AMI personalizada en la documentación de `eksctl`.

> **Important**
>
> - Los nodos de Amazon EKS son instancias estándar de Amazon EC2 y se les facturarán conforme a los precios ordinarios de las instancias de Amazon EC2. Para obtener más información, consulte Precios de Amazon EC2.
> - Puede lanzar nodos de Ubuntu en clústeres extendidos de Amazon EKS en AWS Outposts, pero no puede lanzarlos en clústeres locales en AWS Outposts. Para obtener más información, consulte Amazon EKS en AWS Outposts.

- Puede implementar en instancias de Amazon EC2 con procesadores x86 o Arm. Sin embargo, es posible que las instancias que tienen chips de Inferentia deban instalar primero el SDK de Neuron.

En este procedimiento, se requiere la versión `0.215.0` o posterior de `eksctl`. Puede verificar la versión con el siguiente comando:

```
eksctl version
```

Para obtener instrucciones acerca de cómo instalar o actualizar `eksctl`, consulte Instalación en la documentación de `eksctl`. NOTA: Este procedimiento solo funciona en los clústeres que se crearon con `eksctl`.

1. Copie los siguientes contenidos en su dispositivo. Reemplace `my-cluster` por el nombre del clúster. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones. Debe comenzar con un carácter alfabético y no puede tener más de 100 caracteres. Reemplace `ng-ubuntu` por un nombre para su grupo de nodos. El nombre del grupo de nodos no puede tener más de 63 caracteres. Debe empezar por una letra o un dígito, pero también puede incluir guiones y guiones bajos como caracteres no iniciales. Para implementar en instancias Arm, reemplace `m5.large` por un tipo de instancia Arm. Sustituya `my-ec2-keypair-name` por el nombre de un par de claves SSH de Amazon EC2 que pueda utilizar para conectar mediante SSH con los nodos después de haberlos lanzado. Si aún no tiene un par de claves de Amazon EC2, puede crear uno en la Consola de administración de AWS. Para obtener más información, consulte Pares de claves de Amazon EC2 en la Guía del usuario de Amazon EC2. Sustituya todos los valores de ejemplo restantes por sus propios valores. Una vez que haya llevado a cabo las sustituciones, ejecute el comando modificado para crear el archivo `ubuntu.yaml`.

   Important

   Para implementar un grupo de nodos en las subredes de AWS Outposts, AWS Wavelength o zonas locales de AWS, no pase las subredes de AWS Outposts, AWS Wavelength o Zonas locales de AWS al crear el clúster. Debe especificar las subredes en el siguiente ejemplo. Para obtener más información, consulte Crear un grupo de nodos a partir de un archivo de Config y el Esquema de archivo de configuración en la

documentación de `eksctl`. Reemplace *`region-code`* por la región de AWS en la que se encuentra el clúster.

```
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
```

Para crear un grupo de nodos de Ubuntu Pro, solo tiene que cambiar el valor `amiFamily` a `UbuntuPro2204`.

2. Implemente los nodos con el siguiente comando.

```
eksctl create nodegroup --config-file=ubuntu.yaml
```

Un ejemplo de salida sería el siguiente.

Se generan varias líneas mientras se crean los nodos. Una de las últimas líneas de salida es la siguiente línea de ejemplo.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Opcional) Implemente una aplicación de muestra para probar los nodos de Ubuntu.
4. Se recomienda bloquear el acceso al pod a IMDS si se cumplen las siguientes condiciones:
   - Tiene previsto asignar roles de IAM a todas sus cuentas de servicio de Kubernetes para que los pods solo tengan los permisos mínimos que necesitan.
   - Ninguno de los pods del clúster requiere acceso al servicio de metadatos de instancias (IMDS) de Amazon EC2 por otros motivos, como la recuperación de la región de AWS actual.

   Para obtener más información, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.

Actualización de los nodos autoadministrados para un clúster

Cuando se lanza una nueva AMI optimizada para Amazon EKS, considere la posibilidad de reemplazar los nodos del grupo de nodos autoadministrados con la nueva AMI. Asimismo, si ha actualizado la versión de Kubernetes del clúster de Amazon EKS, actualice los nodos para utilizarlos con la misma versión de Kubernetes.

> **Important**
>
> En este tema, se explican las actualizaciones de los nodos autoadministrados. Si utiliza grupos de nodos administrados, consulte the section called “Actualización”.

Existen dos formas básicas de actualizar grupos de nodos autoadministrados en los clústeres para utilizar una nueva AMI:

Migre sus aplicaciones a un nuevo grupo de nodos

Cree un nuevo grupo de nodos y migre los pods a ese grupo. La migración a un nuevo grupo de nodos es más sencilla que simplemente actualizar el ID de la AMI en una pila de AWS CloudFormation existente. Esto se debe a que el proceso de migración marca al antiguo grupo

de nodos como `NoSchedule` y drena los nodos después de que una nueva pila esté lista para aceptar la carga de trabajo del pod existente.

Actualizar una pila de nodos de AWS CloudFormation

Actualice la pila de AWS CloudFormation para que un grupo de nodos existente utilice la nueva AMI. Este método no es compatible con los grupos de nodos creados con `eksctl`.

Migración de aplicaciones a un nuevo grupo de nodos

En este tema, se describe cómo crear un nuevo grupo de nodos, migrar de forma correcta las aplicaciones existentes al nuevo grupo y eliminar el antiguo grupo de nodos del clúster. Puede migrar a un nuevo grupo de nodos mediante `eksctl` o la Consola de administración de AWS.

- the section called “`eksctl`”
- the section called “Consola de administración de AWS y AWS CLI”

`eksctl`

Migre sus aplicaciones a un nuevo grupo de nodos con **`eksctl`**

Para obtener más información sobre el uso de eksctl para la migración, consulte Nodos no administrados en la documentación de `eksctl`.

En este procedimiento, se requiere la versión `0.215.0` o posterior de `eksctl`. Puede verificar la versión con el siguiente comando:

```
eksctl version
```

Para obtener instrucciones sobre cómo instalar o actualizar `eksctl`, consulte Instalación en la documentación de `eksctl`.

> **Note**
>
> Este procedimiento solo funciona para los clústeres y grupos de nodos que se crearon con `eksctl`.

1. Recupere el nombre de los grupos de nodos existentes al sustituir *`mi clúster`* por el nombre del clúster.

```
eksctl get nodegroups --cluster=my-cluster
```

Un ejemplo de salida sería el siguiente.

```
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. Lance un nuevo grupo de nodos con `eksctl` mediante el siguiente comando. En el comando, sustituya cada *`valor de ejemplo`* por sus valores propios. El número de versión no puede ser posterior a la versión de Kubernetes del plano de control. Además, no puede ser más de dos versiones secundarias anteriores a la versión de Kubernetes para el plano de control. Recomendamos que utilice la misma versión que el plano de control.

   Se recomienda bloquear el acceso al pod a IMDS si se cumplen las siguientes condiciones:

   - Tiene previsto asignar roles de IAM a todas sus cuentas de servicio de Kubernetes para que los pods solo tengan los permisos mínimos que necesitan.
   - Ninguno de los pods del clúster requiere acceso al servicio de metadatos de instancias (IMDS) de Amazon EC2 por otros motivos, como la recuperación de la región de AWS actual.

     Para obtener más información, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.

     Si desea bloquear el acceso del pod a IMDS, agregue la opción `--disable-pod-imds` al siguiente comando.

     > **Note**
     >
     > Para obtener más marcadores disponibles y sus descripciones, consulte https://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. Cuando se complete el comando anterior, verifique con el siguiente comando que todos los nodos tengan el estado `Ready` (Listo):

```
kubectl get nodes
```

4. Elimine el grupo de nodos original con el siguiente comando. En el comando, sustituya cada *`valor de ejemplo`* con los nombres del clúster y el grupo de nodos:

```
eksctl delete nodegroup --cluster my-cluster --name standard-nodes-old
```

Consola de administración de AWS y AWS CLI

Migre sus aplicaciones a un nuevo grupo de nodos con la Consola de administración de AWS y la AWS CLI

1. Lance un nuevo grupo de nodos mediante los pasos que se indican en Creación de nodos autoadministrados de Amazon Linux.
2. Una vez completada la creación de la pila, selecciónela en la consola y elija Salidas.
3. Anote el valor de NodeInstanceRoles correspondiente al grupo de nodos creado. Lo necesita para agregar los nuevos nodos de Amazon EKS al clúster.

> **Note**
>
> Si ha adjuntado políticas de IAM adicionales al rol de IAM del grupo de nodos anterior, debe adjuntar esas mismas políticas al rol de IAM del nuevo grupo de nodos para mantener esa funcionalidad en el nuevo grupo. Esto se aplica si ha agregado permisos para el escalador automático del clúster de Kubernetes, por ejemplo.

4. Actualice los grupos de seguridad de ambos grupos de nodos para que puedan comunicarse entre sí. Para obtener más información, consulte the section called “Requisitos del grupo de seguridad”.
   a. Anote los ID de grupo de seguridad de ambos grupos de nodos. Esto se muestra como el valor NodeSecurityGroup en los resultados de la pila de AWS CloudFormation.

Puede utilizar los siguientes comandos de la AWS CLI para obtener los ID de grupo de seguridad de los nombres de pilas. En estos comandos, `oldNodes` es el nombre de pila de AWS CloudFormation de la pila de nodos antigua y `newNodes` es el nombre de la pila a la que migrará. Reemplace todos los *`valores de ejemplo`* por sus propios valores.

```
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. Agregue reglas de entrada a cada grupo de seguridad de nodos para que acepten tráfico de los otros grupos.

Los siguientes comandos de la AWS CLI agregan reglas de entrada a cada grupo de seguridad que permiten todo el tráfico en todos los protocolos del otro grupo de seguridad. Esta configuración permite que los pods de cada grupo de nodos se comuniquen entre ellos mientras migra la carga de trabajo al nuevo grupo.

```
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. Edite el mapa de configuración de `aws-auth` para asignar el rol de la instancia del nuevo nodo en RBAC.

```
kubectl edit configmap -n kube-system aws-auth
```

Agregue una nueva entrada `mapRoles` para el nuevo grupo de nodos.

```
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
```

Sustituya el fragmento *ARN of instance role (not instance profile)* por el valor de NodeInstanceRole anotado en el procedimiento anterior. A continuación, guarde y cierre el archivo para aplicar el configmap actualizado.

6. Examine el estado de los nodos y espere a que los nuevos nodos se unan al clúster y tengan el estado `Ready` (Listo).

```
kubectl get nodes --watch
```

7. (Opcional) Si utiliza el escalador automático del clúster de Kubernetes, escale la implementación a cero (0) réplicas para evitar acciones de escalado en conflicto.

```
kubectl scale deployments/cluster-autoscaler --replicas=0 -n kube-system
```

8. Utilice el siguiente comando para agregar taints en cada uno de los nodos que desee eliminar con `NoSchedule`. Esto es para que los nuevos pods no se programen ni se vuelvan a programar en los nodos que va a reemplazar. Para obtener más información, consulte Taints y toleraciones en la documentación de Kubernetes.

```
kubectl taint nodes node_name key=value:NoSchedule
```

Si va a actualizar los nodos a una nueva versión de Kubernetes, puede identificar todos los nodos de una determinada versión de Kubernetes (en este caso, `1.31`) y aplicarles una taint con el siguiente fragmento de código. El número de versión no puede ser posterior a la versión de Kubernetes del plano de control. Además, no puede ser más de dos versiones secundarias anteriores a la versión de Kubernetes del plano de control. Recomendamos que utilice la misma versión que el plano de control.

```
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. Identifique el proveedor de DNS del clúster.

```
kubectl get deployments -l k8s-app=kube-dns -n kube-system
```

Un ejemplo de salida sería el siguiente. Este clúster utiliza CoreDNS para la resolución de DNS, pero el clúster puede devolver `kube-dns` en su lugar:

```
NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
coredns   1         1         1            1           31m
```

10. Si su implementación actual está ejecutando menos de dos réplicas, escale la implementación a dos réplicas. Cambie *`coredns`* por `kubedns` si el resultado del comando anterior ha devuelto ese valor.

```
kubectl scale deployments/coredns --replicas=2 -n kube-system
```

11. Vacíe cada uno de los nodos que desea eliminar de su clúster con el siguiente comando:

```
kubectl drain node_name --ignore-daemonsets --delete-local-data
```

Si va a actualizar los nodos a una nueva versión de Kubernetes, identifique y drene todos los nodos de una determinada versión de Kubernetes (en este caso *`1.31`*) con el siguiente fragmento de código.

```
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. Una vez que haya terminado de vaciar los nodos antiguos, revoque las reglas de entrada del grupo de seguridad que autorizó anteriormente. A continuación, elimine la pila de AWS CloudFormation para terminar las instancias.

> **Note**
>
> Si ha adjuntado políticas de IAM adicionales al rol de IAM del grupo de nodos anterior, como agregar permisos para el escalador automático del clúster de Kubernetes, desconecte esas políticas adicionales del rol antes de eliminar la pila de AWS CloudFormation.

a. Revoque las reglas de entrada que creó anteriormente para los grupos de seguridad de nodos. En estos comandos, `oldNodes` es el nombre de pila de AWS CloudFormation de la pila de nodos antigua y `newNodes` es el nombre de la pila a la que migrará.

```
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. Abra la AWS consola de CloudFormation.

c. Seleccione la pila de nodos antigua.

d. Elija Eliminar.

e. En el cuadro de diálogo de confirmación Eliminar pila, elija Eliminar pila.

13. Edite el mapa de configuración de `aws-auth` para eliminar el rol de la instancia del nodo anterior de RBAC.

```
kubectl edit configmap -n kube-system aws-auth
```

Elimine la entrada `mapRoles` del grupo de nodos antiguo.

```
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>
```

Guarde y cierre el archivo para aplicar el mapa de configuración actualizado.

14. (Opcional) Si utiliza el Cluster Autoscaler de Kubernetes, vuelva a escalar la implementación a una réplica.

> **Note**
>
> También debe etiquetar el nuevo grupo de Auto Scaling de forma correcta (por ejemplo, `k8s.io/cluster-autoscaler/enabled,k8s.io/cluster-autoscaler/my-cluster`) y actualizar el comando de implementación del escalador automático del clúster para que señale el grupo de Auto Scaling recién etiquetado. Para obtener más información, consulte Escalador automático de clústeres en AWS.

```
kubectl scale deployments/cluster-autoscaler --replicas=1 -n kube-system
```

15. (Opcional) Verifique que utiliza la última versión del complemento CNI de Amazon VPC para Kubernetes. Es posible que tenga que actualizar la versión de CNI para utilizar los tipos de instancia admitidos más recientes. Para obtener más información, consulte the section called “CNI de Amazon VPC”.
16. Si el clúster utiliza `kube-dns` para la resolución DNS (consulte [migrate-determine-dns-step]), reduzca horizontalmente la implementación de `kube-dns` a una réplica.

```
kubectl scale deployments/kube-dns --replicas=1 -n kube-system
```

Actualización de una pila de nodos de AWS CloudFormation

En este tema, se describe cómo puede actualizar una pila existente de nodos autoadministrados de AWS CloudFormation con una nueva AMI. Puede utilizar este procedimiento para actualizar los nodos a una nueva versión de Kubernetes después de la actualización de un clúster. De lo contrario, puede actualizar a la última AMI optimizada de Amazon EKS para una versión existente de Kubernetes.

> **Important**
>
> En este tema, se explican las actualizaciones de los nodos autoadministrados. Para obtener información sobre el uso de la Simplificación del ciclo de vida de los nodos con grupos de nodos administrados, consulte the section called “Actualización”.

La última plantilla de AWS CloudFormation de nodos de Amazon EKS predeterminada está configurada para lanzar una instancia con la nueva AMI en el clúster antes de eliminar una antigua, una por vez. Esta configuración garantiza que siempre tenga el número deseado de instancias activas del grupo de Auto Scaling en el clúster durante la actualización progresiva.

> **Note**
>
> Este método no es compatible con los grupos de nodos creados con `eksctl`. Si ha creado su clúster o un grupo de nodos con `eksctl`, consulte the section called “Migración”.

1. Determine el proveedor de DNS para el clúster.

```
kubectl get deployments -l k8s-app=kube-dns -n kube-system
```

Un ejemplo de salida sería el siguiente. Este clúster utiliza CoreDNS para la resolución de DNS, pero el clúster puede devolver `kube-dns` en su lugar. El resultado puede tener un aspecto diferente según la versión de `kubectl` que utilice.

```
NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
coredns   1         1         1            1           31m
```

2. Si su implementación actual está ejecutando menos de dos réplicas, escale la implementación a dos réplicas. Cambie *`coredns`* por `kube-dns` si el resultado del comando anterior ha devuelto ese valor.

```
kubectl scale deployments/coredns --replicas=2 -n kube-system
```

3. (Opcional) Si utiliza el escalador automático del clúster de Kubernetes, escale la implementación a cero (0) réplicas para evitar acciones de escalado en conflicto.

```
kubectl scale deployments/cluster-autoscaler --replicas=0 -n kube-system
```

4. Determine el tipo de instancia y el número de instancias deseado del grupo de nodos actual. Ingrese estos valores más tarde cuando actualice la plantilla de AWS CloudFormation del grupo.
   a. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
   b. En el panel de navegación izquierdo, elija Configuraciones de lanzamiento (Configuraciones de lanzamiento) y anote el tipo de instancia de la configuración de lanzamiento de nodos existente.
   c. En el panel de navegación izquierdo, elija Auto Scaling Groups (Grupos de Auto Scaling) y anote el recuento de instancias deseado para el grupo de Auto Scaling de nodos existente.
5. Abra la AWS consola de CloudFormation.
6. Seleccione la pila del grupo de nodos y, a continuación, seleccione Update (Actualizar).
7. Seleccione Replace current template (Reemplazar plantilla actual) y seleccione Amazon S3 URL.
8. Para URL de Amazon S3, pegue la siguiente URL en el área de texto a fin de asegurarse de que utiliza la versión más reciente de la plantilla de AWS Cloud Formation de nodos. A continuación, elija Siguiente:

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2022-12-23/amazon-eks-nodegroup.yaml
```

9. En la página Especificar detalles, rellene los parámetros siguientes y seleccione Siguiente:

   - NodeAutoScalingGroupDesiredCapacity: ingrese el recuento de instancias deseado que registró en un paso anterior. O bien, ingrese el nuevo número deseado de nodos para escalar cuando se actualice la pila.
   - NodeAutoScalingGroupMaxSize: ingrese el número máximo de nodos al que pueda llegar el grupo de Auto Scaling de nodos. Este valor debe ser al menos un nodo más que la capacidad deseada. Es para que pueda realizar una actualización continua de los nodos sin que se reduzca el número durante la actualización.
   - NodeInstanceType: elija el tipo de instancia que registró en un paso anterior. Por otra parte, puede elegir un tipo de instancia diferente para los nodos. Antes de elegir un tipo de instancia diferente, vea Elección de un tipo de instancia de nodo de Amazon EC2 óptimo. Cada tipo de instancia de Amazon EC2 admite un número máximo de interfaces de redes elásticas (interfaz de red) y cada interfaz de red admite un número máximo de direcciones IP. Dado que a cada nodo de trabajo y pod se les asigna su propia dirección IP, es importante elegir un tipo de instancia que admita el número máximo de pods que desea ejecutar en cada nodo de Amazon EC2. Para obtener una lista del número de interfaces de red y direcciones IP admitidas por los tipos de instancias, consulte Direcciones IP por interfaz de red por tipo de instancia. Por ejemplo, el tipo de instancia `m5.large` admite un máximo de 30 direcciones IP para el nodo de trabajo y los pods.

     > **Note**
     >
     > Los tipos de instancias compatibles con la versión más reciente del complemento CNI de Amazon VPC para Kubernetes se muestran en vpc_ip_resource_limit.go en GitHub. Es posible que tenga que actualizar la versión del complemento CNI de Amazon VPC para Kubernetes a fin de utilizar los tipos de instancia admitidos más recientes. Para obtener más información, consulte the section called “CNI de Amazon VPC”.

     > **Important**
     >
     > Algunos tipos de instancias podrían no estar disponibles en todas las regiones de AWS.

   - NodeImageIdSSMParam: el parámetro de Amazon EC2 Systems Manager del ID de AMI al que desee actualizar. El siguiente valor utiliza la última AMI optimizada para Amazon EKS para la versión `1.33` de Kubernetes.

```
/aws/service/eks/optimized-ami/1.33/amazon-linux-2/recommended/image_id
```

Puede reemplazar *1.33* por una versión de la plataforma que sea igual. O bien, debería ser hasta una versión anterior a la versión de Kubernetes que se ejecuta en el plano de control. Le recomendamos que los nodos tengan la misma versión que el plano de control. También puede sustituir *amazon-linux-2* por un tipo de AMI diferente. Para obtener más información, consulte the section called “Obtención de los ID más recientes”.

> **Note**
>
> El uso del parámetro de Amazon EC2 Systems Manager lo habilita a actualizar los nodos en el futuro sin tener que buscar y especificar un ID de AMI. Si la pila de AWS CloudFormation utiliza este valor, cualquier actualización de la pila lanzará siempre la última AMI optimizada para Amazon EKS recomendada en función de la versión de Kubernetes especificada. Este es el caso incluso si no cambia ningún valor en la plantilla.

- NodeImageId: para utilizar su propia AMI personalizada, introduzca el ID de la AMI que va a utilizar.

> **Important**
>
> Este valor anula cualquier valor especificado para NodeImageIdSSMParam. Si desea utilizar el valor NodeImageIdSSMParam asegúrese de que el valor de NodeImageId esté vacío.

- DisableIMDSv1: cada nodo admite de forma predeterminada la versión 1 (IMDSv1) e IMDSv2 del servicio de metadatos de la instancia. Sin embargo, puede desactivar IMDSv1. Seleccione verdadero si no desea que ningún nodo o ningún pod programado en el grupo de nodos utilice IMDSv1. Para obtener más información, consulte Configuración del servicio de metadatos de instancia. Si ha implementado roles de IAM para cuentas de servicio, asigne los permisos necesarios de forma directa a todos los pods que requieren acceso a servicios de AWS. De esta manera, ningún pod del clúster requiere el acceso a IMDS por otros motivos, como la recuperación de la región de AWS actual. A continuación, también puede deshabilitar el acceso a IMDSv2 para los pods que no utilizan redes de host. Para obtener más información, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.

10. (Opcional) En la página Options (Opciones), marque los recursos de la pila. Elija Siguiente.
11. En la página Review (Revisar), revise la información, confirme la advertencia de que la pila puede crear recursos de IAM y elija Update stack (Actualizar pila).

> **Note**
>
> La actualización de cada nodo del clúster tarda varios minutos. Espere a que se complete la actualización de todos los nodos antes de realizar los siguientes pasos.

12. Si su proveedor DNS del clúster es `kube-dns`, reduzca horizontalmente la implementación de `kube-dns` a una réplica.

```
kubectl scale deployments/kube-dns --replicas=1 -n kube-system
```

13. (Opcional) Si utiliza el Cluster Autoscaler de Kubernetes, vuelva a escalar la implementación a la cantidad de réplicas deseada.

```
kubectl scale deployments/cluster-autoscaler --replicas=1 -n kube-system
```

14. (Opcional) Verifique que utiliza la última versión del complemento CNI de Amazon VPC para Kubernetes. Es posible que tenga que actualizar la versión del complemento CNI de Amazon VPC para Kubernetes a fin de utilizar los tipos de instancia admitidos más recientes. Para obtener más información, consulte the section called “CNI de Amazon VPC”.

Simplificación de la administración de computación con AWS Fargate

En este tema se explica cómo utilizar Amazon EKS para ejecutar pods de Kubernetes en AWS Fargate. Fargate es una tecnología que proporciona capacidad informática bajo demanda correctamente dimensionada para contenedores. Con Fargate ya no tendrá que aprovisionar, configurar ni escalar grupos de máquinas virtuales por su cuenta para ejecutar los contenedores. Tampoco tendrá que elegir tipos de servidores, decidir cuándo escalar los grupos de nodos u optimizar el empaquetado de clústeres.

Puede controlar qué pods se comienzan en Fargate y cómo se ejecutan con perfiles de Fargate. Los perfiles de Fargate se definen como parte de su clúster de Amazon EKS. Amazon EKS integra Kubernetes con Fargate mediante controladores creados por AWS con el modelo ascendente y

extensible proporcionado por Kubernetes. Estos controladores funcionan como parte del plano de control de Kubernetes administrado por Amazon EKS y son responsables de programar los pods de Kubernetes nativos en Fargate. Los controladores de Fargate incluyen un nuevo programador que se ejecuta junto con el programador predeterminado de Kubernetes, además de varios controladores de admisión de mutación y validación. Cuando comienza un pod que cumple los criterios para ejecutarse en Fargate, los controladores de Fargate que se ejecutan en el clúster reconocen, actualizan y programan el pod en Fargate.

En este tema se describen los diferentes componentes de los pods que se ejecutan en Fargate y se ofrecen consideraciones especiales sobre cómo utilizar Fargate con Amazon EKS.

Consideraciones sobre Fargate de AWS

Aquí se incluyen algunos aspectos que debe tener en cuenta sobre la utilización de Fargate en Amazon EKS.

- Cada pod que se ejecuta en Fargate tiene su propio límite de computación. No comparten el kernel subyacente, los recursos de CPU, los recursos de memoria o la interfaz de red elástica con otro pod.
- Los Network Load Balancers y los Application Load Balancers (ALB) solo se pueden utilizar con Fargate con destinos IP. Para obtener más información, consulte the section called “Crear un equilibrador de carga de red” y the section called “Equilibrio de carga de aplicaciones”.
- Los servicios expuestos de Fargate solo se ejecutan en modo IP de tipo de destino y no en modo IP de nodo. La forma recomendada de verificar la conectividad de un servicio que se ejecuta en un nodo administrado y un servicio que se ejecuta en Fargate es conectarse a través del nombre del servicio.
- Los pods deben coincidir con un perfil de Fargate en el momento de su programación para ejecutarse en Fargate. Los pods que no coinciden con un perfil de Fargate podrían quedarse atascados como `Pending`. Si existe un perfil de Fargate coincidente, puede eliminar los pods pendientes que haya creado para reprogramarlos en Fargate.
- No se admiten DaemonSets en Fargate. Si su aplicación requiere un daemon, reconfigure ese daemon para que se ejecute como un contenedor asociado en sus pods.
- No se admiten contenedores con privilegios en Fargate.
- Los pods que se ejecutan en Fargate no pueden especificar `HostPort` ni `HostNetwork` en el manifiesto del pod.

- El límite flexible predeterminado de `nofile` y `nproc` es 1024 y el límite invariable es 65 535 para los pods de Fargate.
- Las GPU no están disponibles actualmente en Fargate.
- Los pods que se ejecutan en Fargate solo se admiten en subredes privadas (con acceso de una puerta de enlace NAT a servicios de AWS, pero sin una ruta directa a una puerta de enlace de Internet), por lo que la VPC del clúster debe tener subredes privadas disponibles. Para los clústeres sin acceso a Internet saliente, consulte the section called “Clústeres privados”.
- Puede utilizar el Ajuste de los recursos de pods con el Escalador automático vertical de pods para establecer el tamaño correcto inicial de la CPU y la memoria de sus pods de Fargate y, a continuación, utilizar el Escalado de las implementaciones de pods con el Escalador automático horizontal de pods para ajustar la escala de esos pods. Si desea que el Escalador automático vertical de pods vuelva a implementar automáticamente los pods en Fargate con combinaciones de CPU y memoria más grandes, configure el modo del Escalador automático vertical de pods en `Auto` o `Recreate` para garantizar la funcionalidad correcta. Para obtener más información, consulte el documento Escalador automático vertical de pods en GitHub.
- La resolución de DNS y los nombres de host DNS deben estar habilitados en la VPC. Para obtener más información, consulte Ver y actualizar el soporte de DNS para su VPC.
- Fargate de Amazon EKS agrega defensa en profundidad para las aplicaciones de Kubernetes aislando cada pod dentro de una máquina virtual (VM). Este límite de VM impide el acceso a los recursos basados en host utilizados por otros pods en caso de escape de contenedor, que es un método común para atacar aplicaciones en contenedores y obtener acceso a recursos fuera del contenedor.

  El uso de Amazon EKS no modifica sus responsabilidades en virtud del modelo de responsabilidad compartida. Debe evaluar detenidamente la configuración de los controles de seguridad y gobernanza del clúster. La forma más segura de aislar una aplicación es ejecutarla siempre en un clúster independiente.
- Los perfiles de Fargate admiten la especificación de subredes de bloques de CIDR secundarios de VPC. Puede que desee especificar un bloque de CIDR secundario. Esto es debido a que hay un número limitado de direcciones IP disponibles en una subred. Como resultado, hay también un número limitado de pods que se pueden crear en el clúster. Si usa subredes diferentes para los pods, puede aumentar el número de direcciones IP disponibles. Para obtener más información, consulte Adición de bloques de CIDR IPv4 a una VPC.
- El servicio de metadatos de instancias (IMDS) de Amazon EC2 no está disponible para pods implementados en nodos de Fargate. Si tiene pods implementados en Fargate que necesitan

credenciales de IAM, asígnelos a sus pods mediante roles de IAM para cuentas de servicio. Si los pods necesitan acceso a otra información disponible a través de IMDS, debe llevar a cabo una codificación rígida de esta información en la especificación del pod. Esto incluye la región de AWS o zona de disponibilidad en la que se implementa un pod.

- No se pueden implementar pods de Fargate en AWS Outposts, AWS Wavelength o Zonas locales de AWS.
- Amazon EKS debe aplicar parches periódicamente de pods de Fargate para mantenerlos seguros. Intentamos las actualizaciones de forma que disminuya el impacto, pero hay ocasiones en que los pods deben eliminarse si no se expulsan correctamente. Hay algunas acciones que puede emprender para minimizar las interrupciones. Para obtener más información, consulte the section called “Eventos de aplicación de revisiones del sistema operativo”.
- El complemento CNI de Amazon VPC para Amazon EKS se encuentra instalado en los nodos de Fargate. No puede utilizar complementos CNI alternativos para clústeres de Amazon EKS con nodos de Fargate.
- Un pod que se ejecuta en Fargate monta automáticamente un sistema de archivos de Amazon EFS, sin necesidad de seguir pasos manuales para la instalación de controladores. No se puede utilizar el aprovisionamiento dinámico de volúmenes persistentes con nodos de Fargate, pero se puede utilizar el aprovisionamiento estático.
- Amazon EKS no es compatible con Fargate Spot.
- No puede montar volúmenes de Amazon EBS en los pods de Fargate.
- Puede ejecutar el controlador CSI de Amazon EBS en nodos de Fargate, pero el DaemonSet del nodo CSI de Amazon EBS solo se puede ejecutar en instancias de Amazon EC2.
- Después de marcar un trabajo de Kubernetes como `Completed` o `Failed`, los pods que el trabajo crea normalmente siguen existiendo. Este comportamiento le permite ver sus registros y resultados, pero se pueden generar costos adicionales con Fargate si después no limpia el trabajo.

  Para eliminar automáticamente los pods relacionados después de que se complete o falle un trabajo, puede especificar un periodo de tiempo mediante el controlador de tiempo de vida (TTL). En el siguiente ejemplo, se muestra la especificación `.spec.ttlSecondsAfterFinished` en el manifiesto del trabajo.

```
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
```

Tabla de comparación de Fargate

Criterios	AWS Fargate
Se puede implementar en AWS Outposts	No
Se puede implementar en AWS Local Zones.	No
Puede ejecutar contenedores que requieren Windows	No
Puede ejecutar contenedores que requieran Linux.	Sí
Puede ejecutar cargas de trabajo que requieren el chip de inferencias.	No
Puede ejecutar cargas de trabajo que requieren una GPU.	No
Puede ejecutar cargas de trabajo que requieren procesadores Arm.	No
Puede ejecutar AWS Bottlerocket.	No
Los pods comparten un entorno en tiempo de ejecución del kernel con otros pods.	No. Cada pod tiene un kernel dedicado.
Los pods comparten CPU, memoria, almacenamiento y recursos de red con otros pods.	No. Cada pod tiene recursos dedicados y su tamaño se puede adaptar de forma independi

Criterios	AWS Fargate
	ente para maximizar la utilización de los recursos.
Los pods pueden utilizar más hardware y memoria que lo que se indica en las especific aciones del pod.	No. El pod se puede volver a implementar mediante una vCPU y una configuración de memoria más grandes.
Debe implementar y administrar instancias de Amazon EC2.	No
Debe proteger, mantener y aplicar parches al sistema operativo de las instancias de Amazon EC2.	No
Puede proporcionar argumentos de arranque en la implementación de un nodo, como argumentos kubelet adicionales.	No
Puede asignar direcciones IP a pods desde un bloque de CIDR diferente de la dirección IP asignada al nodo.	No
Se puede utilizar SSH en el nodo.	No: no hay ningún sistema operativo host de nodos para utilizar SSH.
Puede implementar su propia AMI personali zada en los nodos.	No
Puede implementar su propia CNI personali zada en los nodos.	No
Debe actualizar la AMI de nodos por su cuenta	No
Debe actualizar la versión de Kubernetes de los nodos por su cuenta.	No: no administra nodos.
Puede utilizar el almacenamiento de Amazon EBS con pods.	No

Criterios	AWS Fargate
Puede utilizar el almacenamiento de Amazon EFS con pods.	Sí
Puede utilizar el almacenamiento de Amazon FSx para Lustre con pods.	No
Puede utilizar el Network Load Balancer para servicios.	Sí, cuando se utiliza la Creación de un equilibrador de carga de red
Los pods pueden ejecutarse en una subred pública.	No
Puede asignar diferentes grupos de seguridad de VPC a pods individuales.	Sí
Puede ejecutar DaemonSets de Kubernetes.	No
Soporte de `HostPort` y `HostNetwork` en el manifiesto del pod.	No
Disponibilidad regional de AWS	Algunas regiones admitidas por Amazon EKS
Puede ejecutar contenedores en hosts dedicados de Amazon EC2	No
Precios	Costo de una memoria de Fargate individual y configuración de CPU. Cada pod tiene su propio costo. Para obtener más información, consulte AWS Precios de Fargate.

Introducción a AWS Fargate para un clúster

En este tema, se explica cómo comenzar a ejecutar pods en AWS Fargate con su clúster de Amazon EKS.

Si restringe el acceso al punto de conexión público del clúster mediante bloques de CIDR, recomendamos habilitar también el acceso al punto de conexión privado. De este modo, los pods de

Fargate pueden comunicarse con el clúster. Si el punto de conexión privado no está habilitado, los bloques de CIDR que especifique para el acceso público deben incluir los orígenes de salida de su VPC. Para obtener más información, consulte the section called “Acceso al punto de conexión del clúster”.

Requisito previo

Un clúster existente. Si no dispone de un clúster de Amazon EKS, consulte Introducción.

Paso 1: garantía de que los nodos existentes se puedan comunicar con los pods de Fargate

Si está trabajando con un nuevo clúster sin nodos o con un clúster solo con grupos de nodos administrados (consulte the section called “Grupos de nodos administrados”), puede ir directamente a the section called “Paso 2: creación de un rol de ejecución de pods de Fargate”.

Supongamos que está trabajando con un clúster existente que ya tiene nodos asociados. Asegúrese de que los pods de estos nodos puedan comunicarse libremente con los pods que se ejecutan en Fargate. Los pods que se ejecutan en Fargate se configuran automáticamente para utilizar el grupo de seguridad del clúster al que están asociados. Asegúrese de que los nodos existentes en su clúster puedan enviar y recibir tráfico hacia el grupo de seguridad del clúster y recibirlo desde este. Los grupos de nodos administrados se configuran de manera automática para utilizar también el grupo de seguridad del clúster, por lo que no es necesario modificarlos ni verificar si admiten esta función (consulte the section called “Grupos de nodos administrados”).

Para los grupos de nodos existentes que se crearon con `eksctl` o con las plantillas de AWS CloudFormation administradas de Amazon EKS, puede agregar manualmente el grupo de seguridad del clúster a los nodos. O bien, puede modificar la plantilla de lanzamiento del grupo de Auto Scaling para el grupo de nodos a fin de adjuntar el grupo de seguridad del clúster a las instancias. Para obtener más información, consulte Cambio de los grupos de seguridad de una instancia en la Guía del usuario de Amazon VPC.

Puede buscar un grupo de seguridad para su clúster en la Consola de administración de AWS, en la sección Networking (Redes) del clúster. O puede hacerlo con el siguiente comando de AWS CLI. Cuando utilice este comando, sustituya `<my-cluster>` por el nombre del clúster.

```
aws eks describe-cluster --name <my-cluster> --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId
```

Paso 2: creación de un rol de ejecución de pods de Fargate

Cuando su clúster crea pods en AWS Fargate, los componentes que se ejecutan en la infraestructura de Fargate deben hacer llamadas a las API de AWS en su nombre. El rol de ejecución de pods de Amazon EKS proporciona los permisos de IAM para esta tarea. Para crear un rol de ejecución de pods de AWS Fargate, consulte the section called “Rol de IAM de ejecución de pods”.

> **Note**
>
> Si creó el clúster con `eksctl` a través de la opción `--fargate`, el clúster ya tiene un rol de ejecución de pods que puede encontrar en la consola de IAM con el patrón `eksctl-my-cluster-FargatePodExecutionRole-ABCDEFGHIJKL`. Del mismo modo, si utiliza `eksctl` para crear sus perfiles de Fargate, `eksctl` crea su rol de ejecución de pods si aún no se ha creado uno.

Paso 3: creación de un perfil de Fargate para el clúster

Para poder programar los pods que se ejecuten en Fargate en su clúster, debe definir un perfil de Fargate que especifique qué pods deben utilizar Fargate cuando se lancen. Para obtener más información, consulte the section called “Definición de perfiles”.

> **Note**
>
> Si creó el clúster con `eksctl` mediante la opción `--fargate`, ya se habrá creado un perfil de Fargate para el clúster con selectores para todos los pods de los espacios de nombres `kube-system` y `default`. Utilice el siguiente procedimiento para crear perfiles de Fargate para cualquier otro espacio de nombres que desee utilizar con Fargate.

Puede crear un perfil de Fargate con una de las siguientes herramientas:

- the section called “`eksctl`”
- the section called “Consola de administración de AWS”

eksctl

En este procedimiento, se requiere la versión `0.215.0` o posterior de `eksctl`. Puede verificar la versión con el siguiente comando:

```
eksctl version
```

Para obtener instrucciones sobre cómo instalar o actualizar `eksctl`, consulte Instalación en la documentación de `eksctl`.

Cómo crear un perfil de Fargate con **eksctl**

Cree el perfil de Fargate con el siguiente comando de `eksctl` y reemplace cada `<example value>` con valores propios. Debe especificar un espacio de nombres. Sin embargo, la opción `--labels` no es obligatoria.

```
eksctl create fargateprofile \
    --cluster <my-cluster> \
    --name <my-fargate-profile> \
    --namespace <my-kubernetes-namespace> \
    --labels <key=value>
```

Puede usar ciertos comodines para las etiquetas `<my-kubernetes-namespace>` y `<key=value>`. Para obtener más información, consulte the section called “Comodines de perfil de Fargate”.

Consola de administración de AWS

Para crear un perfil de Fargate con Consola de administración de AWS

1. Abra la consola de Amazon EKS.
2. Elija el clúster para el que desea crear un perfil de Fargate.
3. Elija la pestaña Computación.
4. En Perfiles de Fargate, elija Agregar perfil de Fargate.
5. En la página Configurar perfil de Fargate, haga lo siguiente:
   a. En Name (Nombre), ingrese un nombre único para el perfil de Fargate. El nombre debe ser único.
   b. En Rol de ejecución de pods, elija el rol de ejecución de pods que se va a utilizar con el perfil de Fargate. Solo se muestran los roles de IAM con la entidad principal del servicio de `eks-`

`fargate-pods.amazonaws.com`. Si no ve ningún rol, debe crear uno. Para obtener más información, consulte the section called “Rol de IAM de ejecución de pods”.

c. Modifique las Subredes seleccionadas según sea necesario.

> **Note**
>
> Solo las subredes privadas son compatibles con los pods que se ejecutan en Fargate.

d. En Etiquetas, puede etiquetar su perfil de Fargate si lo desea. Estas etiquetas no se propagan a otros recursos asociados con el perfil, como los pods.

e. Elija Siguiente.

6. En la página Configurar la selección de pods, haga lo siguiente:

a. En Espacio de nombres, ingrese un espacio de nombres que coincida con los pods.

- Puede usar espacios de nombres específicos para que coincidan, como `kube-system` o `default`.
- Puede usar ciertos comodines (por ejemplo, `prod-*`) para que coincidan con varios espacios de nombres (por ejemplo, `prod-deployment` y `prod-test`). Para obtener más información, consulte the section called “Comodines de perfil de Fargate”.

b. (Opcional) Agregue etiquetas de Kubernetes al selector. Agréguelos específicamente al selector con el que deben coincidir los pods del espacio de nombres especificado.

- Puede agregar la etiqueta `infrastructure: fargate` al selector para que solo los pods del espacio de nombres especificado que también tengan la etiqueta `infrastructure: fargate` de Kubernetes coincidan con el selector.
- Puede usar ciertos comodines (por ejemplo, `key?: value?`) para que coincidan con varios espacios de nombres (por ejemplo, `keya: valuea` y `keyb: valueb`). Para obtener más información, consulte the section called “Comodines de perfil de Fargate”.

c. Elija Siguiente.

7. En la página Revisar y crear, revise la información de su perfil de Fargate y elija Crear.

Paso 4: actualización de CoreDNS

De forma predeterminada, CoreDNS está configurado para ejecutarse en la infraestructura de Amazon EC2 en clústeres de Amazon EKS. Si desea ejecutar solo los pods de Fargate en el clúster, complete los pasos que se describen a continuación.

> **Note**
>
> Si ha creado el clúster con `eksctl` mediante la opción `--fargate`, entonces puede ir directamente a the section called “Siguientes pasos”.

1. Cree un perfil de Fargate para CoreDNS con el siguiente comando. Reemplace `<my-cluster>` con su nombre de clúster, `<111122223333>` con su ID de cuenta, `<AmazonEKSFargatePodExecutionRole>` con el nombre de su rol de ejecución del pod y `<000000000000000a>`, `<000000000000000b>` y `<000000000000000c>` con los ID de las subredes privadas. Si no tiene un rol de ejecución de pods, debe crear uno antes (consulte the section called “Paso 2: creación de un rol de ejecución de pods de Fargate”).

   > **Important**
   >
   > El ARN de rol no puede incluir una ruta de acceso que no sea `/`. Por ejemplo, si el nombre de su rol es `development/apps/AmazonEKSFargatePodExecutionRole`, tendrá que cambiarlo a `AmazonEKSFargatePodExecutionRole` cuando especifique el ARN del rol. El formato del ARN del rol debe ser `arn:aws:iam::<111122223333>:role/<AmazonEKSFargatePodExecutionRole>`.

   ```
   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. Active un despliegue de la implementación `coredns`.

   ```
   kubectl rollout restart -n kube-system deployment coredns
   ```

Siguientes pasos

- Puede comenzar a migrar las aplicaciones existentes para que se ejecuten en Fargate con el siguiente flujo de trabajo.
  a. the section called “Creación de un perfil de Fargate” que coincida con el espacio de nombres de Kubernetes y las etiquetas de Kubernetes de su aplicación.
  b. Elimine y vuelva a crear los pods existentes para que se programen en Fargate. Modifique `<namespace>` y `<deployment-type>` para actualizar sus pods específicos.

```
kubectl rollout restart -n <namespace> deployment <deployment-type>
```

- Implemente the section called “Equilibrio de carga de aplicaciones” para permitir que los objetos de entrada de los pods se ejecuten en Fargate.
- Puede utilizar el the section called “Escalador automático vertical de pods” para medir correctamente inicialmente el tamaño de la CPU y la memoria de los pods de Fargate y, a continuación, utilizar el the section called “Escalador automático de pods horizontales” para escalar esos pods. Si desea que el Escalador automático vertical de pods vuelva a implementar automáticamente los pods en Fargate con combinaciones de CPU y memoria mayores, configure el modo del Escalador automático vertical de pods en `Auto` o `Recreate`. Esto es para garantizar una funcionalidad correcta. Para obtener más información, consulte el documento Escalador automático vertical de pods en GitHub.
- Puede configurar el recopilador AWS Distro for OpenTelemetry (ADOT) para el monitoreo de aplicaciones siguiendo estas instrucciones.

Cómo definir los pods que deben lanzarse en AWS Fargate

Antes de programar pods en Fargate en el clúster, debe definir al menos un perfil de Fargate que especifique qué pods utilizan Fargate cuando se lanzan.

Como administrador, puede utilizar un perfil de Fargate para declarar qué pods se ejecutan en Fargate. Puede hacerlo a través de los selectores del perfil. Puede agregar hasta cinco selectores a cada perfil. Cada selector debe contener un espacio de nombres. El selector también puede incluir etiquetas. El campo de etiqueta consta de varios pares de clave-valor opcionales. Los pods que coinciden con un selector se programan en Fargate. Los pods se comparan mediante un espacio de nombres y las etiquetas que se especifican en el selector. Si se define un selector de espacio de nombres sin etiquetas, Amazon EKS intenta programar todos los pods que se ejecutan en ese

espacio de nombres en Fargate mediante el perfil. Si un pod programado coincide con alguno de los selectores del perfil de Fargate, ese pod se programa en Fargate.

Si un pod coincide con varios perfiles de Fargate, puede especificar qué perfil utiliza un pod al agregar la siguiente etiqueta de Kubernetes a la especificación del pod: `eks.amazonaws.com/fargate-profile: my-fargate-profile`. El pod debe coincidir con un selector en ese perfil para que se pueda programar en Fargate. Las reglas de afinidad y antiafinidad de Kubernetes no se aplican y no son necesarias con los pods de Fargate de Amazon EKS.

Cuando se crea un perfil de Fargate, se debe especificar un rol de ejecución de pods. Este rol de ejecución es para los componentes de Amazon EKS que se ejecutan en la infraestructura de Fargate mediante el perfil. Se agrega al control de acceso basado en roles (RBAC) de Kubernetes del clúster para la autorización. De este modo, el `kubelet` que se ejecuta en la infraestructura de Fargate puede registrarse en su clúster de Amazon EKS y aparecer en su clúster como un nodo. El rol de ejecución de pods también proporciona permisos de IAM a la infraestructura de Fargate para permitir el acceso de lectura a los repositorios de imágenes de Amazon ECR. Para obtener más información, consulte the section called “Rol de IAM de ejecución de pods”.

Los perfiles de Fargate no se pueden cambiar. Sin embargo, puede crear un nuevo perfil actualizado para reemplazar un perfil existente y, a continuación, eliminar el original.

> **Note**
>
> Los pods que se están ejecutando con un perfil de Fargate se detienen y pasan a un estado pendiente cuando se elimina el perfil.

Si el estado de alguno de los perfiles de Fargate de un clúster es `DELETING`, hay que esperar a que se borre el perfil de Fargate antes de crear otros perfiles en ese clúster.

> **Note**
>
> Fargate actualmente no admite topologySpreadConstraints de Kubernetes.

Amazon EKS y Fargate distribuyen los pods en cada una de las subredes definidas en el perfil de Fargate. Sin embargo, es posible que acabe con una propagación desigual. Si debe tener una propagación uniforme, utilice dos perfiles de Fargate. Es importante una propagación uniforme en los

escenarios en los que se desea desplegar dos réplicas y no se desea ningún tiempo de inactividad. Se recomienda que cada perfil tenga solo una subred.

Componentes de un perfil de Fargate

Un perfil de Fargate consta de los siguientes componentes.

Rol de ejecución de pods

Cuando el clúster crea pods en AWS Fargate, el `kubelet` que se ejecuta en la infraestructura de Fargate debe hacer llamadas a las API de AWS en su nombre. Por ejemplo, necesita hacer llamadas para extraer imágenes del contenedor de Amazon ECR. El rol de ejecución de pods de Amazon EKS proporciona los permisos de IAM para esta tarea.

Al crear un perfil de Fargate, debe especificar un rol de ejecución de pods para utilizarlo con los pods. Este rol se agrega al control de acceso basado en roles (RBAC) de Kubernetes del clúster para su autorización. De este modo, el `kubelet` que se está ejecutando en la infraestructura de Fargate puede registrarse en el clúster de Amazon EKS y aparecer en el clúster como un nodo. Para obtener más información, consulte the section called “Rol de IAM de ejecución de pods”.

Subredes

Los identificadores de las subredes en las que se lanzarán los pods que utilizan este perfil. En este momento, a los pods que se ejecutan en Fargate no se les asignan direcciones IP públicas. Por lo tanto, para este parámetro solo se aceptan subredes privadas que no tengan una ruta directa a una puerta de enlace de Internet.

Selectores

Los selectores que deben coincidir para que los pods utilicen este perfil de Fargate. Puede especificar hasta cinco selectores en un perfil Fargate. Los selectores tienen los siguientes componentes:

- Espacio de nombres: debe especificar un espacio de nombres para un selector. El selector solo coincide con los pods que se crean en este espacio de nombres. Sin embargo, puede crear varios selectores para orientar varios espacios de nombres.
- Etiquetas: si lo desea, puede especificar etiquetas de Kubernetes que coincidan con el selector. El selector solo coincide con los pods que tienen todas las etiquetas especificadas en el selector.

Comodines de perfil de Fargate

Además de los caracteres permitidos por Kubernetes, se permite utilizar * y ? en los criterios del selector para los espacios de nombres, las claves de las etiquetas y los valores de las etiquetas:

- * representa ninguno, uno o varios caracteres. Por ejemplo, `prod*` puede representar `prod` y `prod-metrics`.
- ? representa un solo carácter (por ejemplo, `value?` puede representar `valuea`). Sin embargo, no puede representar `value` y `value-a`, porque ? solo puede representar exactamente un carácter.

Estos caracteres comodín se pueden usar en cualquier posición y en combinación (por ejemplo, `prod*`, `*dev` y `frontend*?`). No se admiten otros comodines ni formas de coincidencia de patrones, como las expresiones regulares.

Si hay varios perfiles que coinciden con el espacio de nombres y las etiquetas en la especificación del pod, Fargate elige el perfil según la clasificación alfanumérica por nombre de perfil. Por ejemplo, si tanto el perfil A (con el nombre `beta-workload`) como el perfil B (con el nombre `prod-workload`) tienen selectores coincidentes para los pods que se lanzarán, Fargate elige el perfil A (`beta-workload`) para los pods. Los pods tienen etiquetas con el perfil A en los pods (por ejemplo, `eks.amazonaws.com/fargate-profile=beta-workload`).

Si desea migrar los pods de Fargate existentes a los nuevos perfiles que utilizan comodines, hay dos maneras de hacerlo:

- Cree un nuevo perfil con los selectores correspondientes y, a continuación, elimine los perfiles antiguos. Los pods etiquetados con perfiles antiguos se reprograman con nuevos perfiles coincidentes.
- Si quiere migrar cargas de trabajo, pero no sabe con seguridad qué etiquetas de Fargate hay en cada pod de Fargate, puede utilizar el siguiente método. Cree un nuevo perfil con un nombre que se ordene alfanuméricamente en primer lugar entre los perfiles del mismo clúster. A continuación, recicle los pods de Fargate que deban migrar a nuevos perfiles.

Creación de un perfil de Fargate

En esta sección se explica cómo crear un perfil de Fargate. También debe haber creado un rol de ejecución de pods para utilizarlo en su perfil de Fargate. Para obtener más información, consulte the section called “Rol de IAM de ejecución de pods”. Los pods que se ejecutan en Fargate solo se admiten en subredes privadas con acceso a la puerta de enlace de NAT a AWS, pero no una ruta

directa a una puerta de enlace de Internet. Esto es para que la VPC de su clúster tenga subredes privadas disponibles.

Para crear un perfil puede realizar lo siguiente:

- the section called “`eksctl`”
- the section called “Consola de administración de AWS”

`eksctl`

Para crear un perfil de Fargate con **`eksctl`**

Cree el perfil de Fargate con el siguiente comando de `eksctl` y reemplace cada valor de ejemplo con valores propios. Debe especificar un espacio de nombres. Sin embargo, la opción `--labels` no es obligatoria.

```
eksctl create fargateprofile \
    --cluster my-cluster \
    --name my-fargate-profile \
    --namespace my-kubernetes-namespace \
    --labels key=value
```

Puede usar ciertos comodines para las etiquetas `my-kubernetes-namespace` y `key=value`. Para obtener más información, consulte the section called “Comodines de perfil de Fargate”.

Consola de administración de AWS

Para crear un perfil de Fargate con Consola de administración de AWS

1. Abra la consola de Amazon EKS.
2. Elija el clúster para el que desea crear un perfil de Fargate.
3. Elija la pestaña Computación.
4. En Perfiles de Fargate, elija Agregar perfil de Fargate.
5. En la página Configurar perfil de Fargate, haga lo siguiente:
   a. En Name (Nombre), ingrese un nombre único para su perfil de Fargate; por ejemplo, `my-profile`.
   b. En Rol de ejecución de pods, elija el rol de ejecución de pods que se va a utilizar con el perfil de Fargate. Solo se muestran los roles de IAM con la entidad principal del servicio de `eks-`

`fargate-pods.amazonaws.com`. Si no ve ningún rol, debe crear uno. Para obtener más información, consulte the section called “Rol de IAM de ejecución de pods”.

c. Modifique las Subredes seleccionadas según sea necesario.

> **Note**
>
> Solo las subredes privadas son compatibles con los pods que se ejecutan en Fargate.

d. En Etiquetas, puede etiquetar su perfil de Fargate si lo desea. Estas etiquetas no se propagan a otros recursos asociados con el perfil, como los pods.

e. Elija Siguiente.

6. En la página Configurar la selección de pods, haga lo siguiente:

a. En Espacio de nombres, ingrese un espacio de nombres que coincida con los pods.

- Puede usar espacios de nombres específicos para que coincidan, como `kube-system` o `default`.
- Puede usar ciertos comodines (por ejemplo, `prod-*`) para que coincidan con varios espacios de nombres (por ejemplo, `prod-deployment` y `prod-test`). Para obtener más información, consulte the section called “Comodines de perfil de Fargate”.

b. (Opcional) Agregue etiquetas de Kubernetes al selector. Agréguelos específicamente al selector con el que deben coincidir los pods del espacio de nombres especificado.

- Puede agregar la etiqueta `infrastructure: fargate` al selector para que solo los pods del espacio de nombres especificado que también tengan la etiqueta `infrastructure: fargate` de Kubernetes coincidan con el selector.
- Puede usar ciertos comodines (por ejemplo, `key?: value?`) para que coincidan con varios espacios de nombres (por ejemplo, `keya: valuea` y `keyb: valueb`). Para obtener más información, consulte the section called “Comodines de perfil de Fargate”.

c. Elija Siguiente.

7. En la página Revisar y crear, revise la información de su perfil de Fargate y elija Crear.

Eliminación de un perfil de Fargate

En este tema se explica cómo eliminar un perfil de Fargate. Al eliminar un perfil de Fargate, se eliminan todos los pods que se programaron en Fargate con el perfil. Si esos pods coinciden con otro

perfil de Fargate, se programan en Fargate con ese perfil. Si ya no coinciden con ningún perfil de Fargate, significa que no están programados en Fargate y pueden permanecer como pendientes.

Solo un perfil de Fargate de un clúster puede tener el estado `DELETING` a la vez. Espere a que un perfil de Fargate termine de eliminarse para poder eliminar cualquier otro perfil de ese clúster.

Puede eliminar perfiles con cualquiera de las siguientes herramientas:

- the section called “`eksctl`”
- the section called “Consola de administración de AWS”
- the section called “AWS CLI”

`eksctl`

Elimine un perfil de Fargate con **`eksctl`**

Utilice el siguiente comando para eliminar un perfil de un clúster. Reemplace los *`valores de ejemplo`* por sus propios valores.

```
eksctl delete fargateprofile  --name my-profile --cluster my-cluster
```

Consola de administración de AWS

Elimine un perfil de Fargate con Consola de administración de AWS

1. Abra la consola de Amazon EKS.
2. En el panel de navegación izquierdo, elija Clusters (Clústeres). En la lista de clústeres, elija el clúster del que desea eliminar el perfil de Fargate.
3. Elija la pestaña Computación.
4. Elija el perfil de Fargate que desea eliminar y, a continuación, elija Delete (Eliminar).
5. En la página Delete Fargate Profile (Eliminar perfil de Fargate), ingrese el nombre del perfil y, a continuación, elija Delete (Eliminar).

AWS CLI

Elimine un perfil de Fargate con la AWS CLI

Utilice el siguiente comando para eliminar un perfil de un clúster. Reemplace los *valores de ejemplo* por sus propios valores.

```
aws eks delete-fargate-profile --fargate-profile-name my-profile --cluster-name my-cluster
```

Descripción de los detalles de configuración de pods de Fargate

En esta sección se describen algunos de los detalles de configuración únicos de los pods para ejecutar pods de Kubernetes en AWS Fargate.

Memoria y CPU de los pods

Con Kubernetes, puede definir solicitudes, una cantidad mínima de vCPU y recursos de memoria que se asignan a cada contenedor en un pod. Kubernetes programa los pods para garantizar que al menos los recursos solicitados para cada pod estén disponibles en el recurso informático. Para obtener más información, consulte la administración de recursos de computación para contenedores en la documentación de Kubernetes.

> **Note**
>
> Dado que Amazon EKS Fargate ejecuta solo un pod por nodo, no se produce el escenario de expulsión de pods en caso de menos recursos. Todos los pods de Amazon EKS Fargate se ejecutan con prioridad garantizada, por lo que la CPU y la memoria solicitadas deben ser iguales al límite de todos los contenedores. A fin de obtener más información, consulte Configurar la calidad del servicio para los pods en la documentación de Kubernetes.

Cuando los pods están programados en Fargate, las reservas de vCPU y memoria dentro de la especificación del pod determinan cuánta CPU y memoria se debe aprovisionar para el pod.

- La solicitud máxima de cualquier contenedor Init se utiliza para determinar los requisitos de vCPU y de memoria de la solicitud Init.
- Las solicitudes para todos los contenedores de larga duración se suman para determinar los requisitos de memoria y vCPU de las solicitudes de larga duración.
- El mayor de los dos valores anteriores se elige para la solicitud de vCPU y de memoria que se utilizará para el pod.

- Fargate agrega 256 MB a la reserva de memoria de cada pod para los componentes requeridos de Kubernetes (`kubelet`, `kube-proxy` y `containerd`).

Fargate redondea hasta la siguiente configuración de computación que más se acerque a la suma de las solicitudes de vCPU y memoria para garantizar que los pods siempre tengan los recursos que necesitan para ejecutarse.

Si no especifica una combinación de vCPU y memoria, se utilizará la combinación más chica disponible (0,25 vCPU y 0,5 GB de memoria).

En la siguiente tabla se muestran las combinaciones de vCPU y memoria disponibles para los pods que se ejecutan en Fargate.

Valor de vCPU	Valor de memoria
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	Entre 4 GB y 16 GB en incrementos de 1 GB
4 vCPU	Entre 8 GB y 30 GB en incrementos de 1 GB
8 vCPU	Entre 16 GB y 60 GB en incrementos de 4 GB
16 vCPU	Entre 32 GB y 120 GB en incrementos de 8 GB

La memoria adicional reservada para los componentes de Kubernetes puede generar una tarea de Fargate con más vCPU de las que se solicitaron aprovisionar. Por ejemplo, una solicitud de 1 vCPU y 8 GB de memoria tendrá 256 MB agregados a su solicitud de memoria y aprovisionará una tarea de Fargate con 2 vCPU y 9 GB de memoria, ya que no hay ninguna tarea con 1 vCPU y 9 GB de memoria disponible.

No hay correlación entre el tamaño del pod que se ejecuta en Fargate y el tamaño del nodo informado por Kubernetes con `kubectl get nodes`. El tamaño del nodo informado suele ser mayor que la capacidad del pod. Puede verificar la capacidad de los pods con el siguiente comando. Reemplace *`default`* por el espacio de nombres del pod y *`pod-name`* por el nombre del pod.

```
kubectl describe pod --namespace default pod-name
```

Un ejemplo de salida sería el siguiente.

```
[...]
annotations:
    CapacityProvisioned: 0.25vCPU 0.5GB
[...]
```

La anotación `CapacityProvisioned` representa la capacidad del pod forzada y determina el costo del pod que se ejecuta en Fargate. Para obtener información sobre los precios de las configuraciones informáticas, consulte Precios de AWS Fargate.

Almacenamiento de Fargate

Un pod que se ejecuta en Fargate monta automáticamente un sistema de archivos de Amazon EFS, sin necesidad de seguir pasos manuales para la instalación de controladores. No se puede utilizar el aprovisionamiento dinámico de volúmenes persistentes con nodos de Fargate, pero se puede utilizar el aprovisionamiento estático. Para obtener más información, consulte Controlador de CSI de Amazon EFS en GitHub.

Cuando se aprovisiona, cada pod que se ejecuta en Fargate recibe un almacenamiento efímero predeterminado de 20 GiB. Este tipo de almacenamiento se elimina después de que un pod se detenga. Los nuevos pods lanzados en Fargate tienen el cifrado del volumen de almacenamiento efímero habilitado de forma predeterminada. El almacenamiento de pod efímero se cifra con un algoritmo de cifrado AES-256 mediante claves administradas por AWS Fargate.

> **Note**
>
> El almacenamiento utilizable predeterminado para pods de Amazon EKS que se ejecuta en Fargate tiene menos de 20 GiB. Esto se debe a que parte del espacio lo utilizan `kubelet` y otros módulos de Kubernetes que se cargan dentro del pod.

La cantidad total de almacenamiento efímero se puede aumentar hasta un máximo de 175 GiB. Para configurar el tamaño con Kubernetes, especifique las solicitudes del recurso de `ephemeral-storage` para cada contenedor en un pod. Cuando Kubernetes programa pods, asegura que la suma de las solicitudes de recursos para cada pod es inferior a la capacidad de la tarea de Fargate.

Para obtener más información, consulte Resource Management for Pods and Containers en la documentación de Kubernetes.

Amazon EKS Fargate proporciona más almacenamiento efímero del solicitado para el uso del sistema. Por ejemplo, una solicitud de 100 GiB aprovisionará una tarea de Fargate con 115 GiB de almacenamiento efímero.

Definición de acciones para los eventos de aplicación de revisiones del sistema operativo de AWS Fargate

Amazon EKS debe aplicar parches periódicamente del sistema operativo para los nodos de AWS Fargate a fin de mantenerlos seguros. Como parte del proceso de aplicación de parches, reciclamos los nodos para instalar los parches del sistema operativo. Las actualizaciones se intentan de tal manera que se produzca el menor impacto en sus servicios. Sin embargo, si los pods no se expulsan correctamente, hay ocasiones en que deben eliminarse. A continuación, se muestran las acciones que puede realizar para minimizar las posibles interrupciones:

- Establezca los presupuestos de interrupción de pods (PDB) adecuados para controlar el número de pods que están inactivos simultáneamente.
- Cree reglas de Amazon EventBridge para controlar las expulsiones fallidas antes de que se eliminen los pods.
- Reinicie manualmente los pods afectados antes de la fecha de expulsión que se indica en la notificación que reciba.
- Cree una configuración de notificaciones en Notificaciones de usuario de AWS.

Amazon EKS trabaja en estrecha colaboración con la comunidad de Kubernetes para que las correcciones de errores y los parches de seguridad estén disponibles lo antes posible. Todos los pods de Fargate empiezan en la versión de parche de Kubernetes más recientes, que está disponible en Amazon EKS para la versión de Kubernetes de su clúster. Si tiene un pod con una versión de parche anterior, Amazon EKS podría reciclarlo para actualizarlo a la última versión. Esto garantiza que los pods estén equipados con las últimas actualizaciones de seguridad. De esa forma, si hay un problema de Vulnerabilidades y exposiciones comunes (CVE) crítico, estará al tanto para reducir los riesgos de seguridad.

Cuando se actualice el sistema operativo de AWS Fargate, Amazon EKS enviará una notificación que incluirá los recursos afectados y la fecha de las próximas expulsiones de pods. Si la fecha de expulsión prevista resulta inoportuna, es posible reiniciar manualmente los pods afectados antes

de la fecha de expulsión indicada en la notificación. Todos los pods creados antes del momento en que reciba la notificación están sujetos a expulsión. Consulte la documentación de Kubernetes para obtener más instrucciones sobre cómo reiniciar manualmente los pods.

Para limitar el número de pods que están inactivos al mismo tiempo cuando se reciclan pods, puede establecer presupuestos de interrupción de pods (PDB). Puede utilizar PDB para definir la disponibilidad mínima en función de los requisitos de cada una de las aplicaciones y, al mismo tiempo, permitir que se produzcan actualizaciones. La disponibilidad mínima de PDB debe ser inferior al 100 %. Para obtener más información, consulte Specifying a Disruption Budget for your Application en la documentación de Kubernetes.

Amazon EKS utiliza la API de expulsión para drenar el pod de forma segura a la vez que se respetan los PDB que configuró para la aplicación. La zona de disponibilidad expulsa los pods para minimizar el impacto. Si la expulsión tiene éxito, el nuevo pod obtiene el parche más reciente y no será necesario llevar a cabo más acciones.

Cuando falla la expulsión de un pod, Amazon EKS envía un evento a su cuenta con detalles sobre los pods que han fallado la expulsión. Puede actuar en función del mensaje antes de la hora de finalización programada. El tiempo específico varía según la urgencia del parche. Cuando llega el momento, Amazon EKS intenta expulsar de nuevo los pods. Sin embargo, esta vez no se envía un nuevo evento si la expulsión falla. Si la expulsión vuelve a fallar, los pods existentes se eliminan periódicamente para que los nuevos pods puedan tener el último parche.

A continuación, se muestra un evento de ejemplo recibido cuando falla la expulsión de pods. Contiene detalles sobre el clúster, el nombre del pod, el espacio de nombres del pod, el perfil de Fargate y la hora de finalización programada.

```
{
    "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]"
    }
}
```

Además, tener varios PDB asociados a un pod puede provocar un error de expulsión. Este evento devuelve el siguiente mensaje de error.

```
"evictErrorMessage": "This pod has multiple PodDisruptionBudget, which the eviction
 subresource does not support",
```

Puede crear una acción deseada basada en este evento. Por ejemplo, puede ajustar el presupuesto de interrupción de pods (PDB) para controlar cómo se expulsan los pods. Más concretamente, supongamos que empieza con un PDB que especifica el porcentaje objetivo de pods que están disponibles. Antes de que los pods finalicen forzosamente durante una actualización, puede ajustar el PDB a un porcentaje diferente de pods. Para recibir este evento, debe crear una regla de Amazon EventBridge en la cuenta de AWS y la región de AWS a la que pertenece el clúster. La regla debe utilizar el siguiente patrón personalizado. Para obtener más información, consulte Creación de reglas de EventBridge que reaccionan a eventos en la Guía del usuario de Amazon EventBridge.

```
{
  "source": ["aws.eks"],
  "detail-type": ["EKS Fargate Pod Scheduled Termination"]
}
```

Se puede establecer un objetivo adecuado para que el evento lo capture. Para obtener una lista completa de los destinos admitidos, consulte Destinos de Amazon EventBridge en la Guía del usuario de Amazon EventBridge. También puede crear una configuración de notificaciones en Notificaciones de usuario de AWS. Al utilizar la Consola de administración de AWS para crear la notificación, en Reglas de eventos, seleccione Elastic Kubernetes Service (EKS) para el AWS nombre del servicio y Terminación programada de EKS Fargate Pod para el tipo de evento. Para obtener más información, consulte Introducción a las notificaciones de usuario de AWS en la Guía de usuario de notificaciones de usuario de AWS.

Consulte Preguntas frecuentes: notificación de expulsión del pod de Fargate en AWS re:Post para consultar las preguntas más frecuentes sobre las expulsiones del pod de EKS.

Recopilación de métricas de uso y aplicación de AWS Fargate

Puede recopilar métricas del sistema y métricas de uso de CloudWatch para AWS Fargate.

Métricas de aplicación

Para aplicaciones que se ejecutan en Amazon EKS y AWS Fargate, puede utilizar AWS Distro para OpenTelemetry (ADOT). ADOT le permite recopilar métricas del sistema y enviarlas a los paneles de CloudWatch Container Insights. Para empezar a utilizar ADOT para aplicaciones que se ejecutan en Fargate, consulte Using CloudWatch Container Insights with AWS Distro for OpenTelemetry (Uso de CloudWatch Container Insights con Distro for OpenTelemetry) en la documentación de ADOT.

Métricas de uso

Puede utilizar las métricas de uso de CloudWatch para proporcionar visibilidad sobre el uso de los recursos de su cuenta. Utilice estas métricas para visualizar el uso actual del servicio en paneles y gráficos de CloudWatch.

Las métricas de uso de AWS Fargate se corresponden con las Service Quotas de servicio de AWS. Puede configurar alarmas que le avisen cuando su uso se acerque a una Service Quota. Para obtener más información acerca de las cuotas de servicio de Fargate, consulte the section called "Service Quotas".

AWS Fargate publica las siguientes métricas en el espacio de nombres `AWS/Usage`.

Métrica	Descripción
`ResourceCount`	El número total de los recursos especificados que se ejecutan en su cuenta. Los recursos se definen por las dimensiones asociadas a la métrica.

Las siguientes dimensiones se utilizan para ajustar las métricas de uso que publica AWS Fargate.

Dimensión	Descripción
`Service`	El nombre del servicio de AWS que contiene el recurso. Para las métricas de uso de

Dimensión	Descripción
	AWS Fargate, el valor de esta dimensión es `Fargate`.
`Type`	El tipo de entidad que se notifica. Actualmente, el único valor válido para las métricas de uso de AWS Fargate es `Resource`.
`Resource`	El tipo de recurso que se ejecuta. En la actualidad, AWS Fargate devuelve información sobre la utilización bajo demanda de Fargate. El valor de recurso para la utilizaci ón bajo demanda de Fargate es `OnDemand`. [NOTA] ==== El uso bajo demanda de Fargate combina los pods de Amazon EKS que utilizan Fargate, las tareas de Amazon ECS que utilizan el tipo de lanzamiento de Fargate y las tareas de Amazon ECS que utilizan el proveedor de capacidad de `FARGATE`. ====
`Class`	La clase de recurso a la que se realiza el seguimiento. En la actualidad, AWS Fargate no utiliza la dimensión de clase.

Creación de una alarma de CloudWatch para monitorear las métricas de uso de recursos de Fargate

AWS Fargate proporciona métricas de uso de CloudWatch que corresponden a las Service Quotas de AWS para el uso de recursos bajo demanda de Fargate. En la consola de Service Quotas, puede ver el uso en un gráfico. También puede configurar alarmas que le avisen cuando su uso se acerque a una cuota de servicio. Para obtener más información, consulte the section called “Recopilación de métricas”.

Siga estos pasos para crear una alarma de CloudWatch basada en las métricas de uso de recursos de Fargate.

1. Abra la consola de Service Quotas en https://console.aws.amazon.com/servicequotas/.
2. En el panel de navegación de la izquierda, elija Servicios de AWS.
3. En la lista Servicios de AWS, busque y seleccione AWS Fargate.
4. En la lista Service quotas (Cuotas de servicio), elija la cuota de utilización de Fargate para la que desee crear una alarma.
5. En la sección Amazon CloudWatch alarms (Alarmas de Amazon CloudWatch), elija Create (Crear).
6. En Alarm threshold (Umbral de alarma), elija el porcentaje del valor de la cuota aplicada que desee establecer como valor de la alarma.
7. En Alarm name (Nombre de la alarma), escriba el nombre de la alarma y elija Create (Crear).

Inicio del registro de AWS Fargate para un clúster

Amazon EKS en Fargate ofrece un enrutador de registros integrado basado en Fluent Bit. Esto significa que no ejecuta explícitamente un contenedor de Fluent Bit como archivo sidecar, sino que Amazon lo ejecuta. Todo lo que tiene que hacer es configurar el enrutador de registros. La configuración se realiza a través de un `ConfigMap` dedicado que debe cumplir los siguientes criterios:

- Tener un `aws-logging` con nombre.
- Haber sido creado en un espacio de nombres dedicado llamado `aws-observability`.
- No puede superar los 5300 caracteres.

Una vez que haya creado el `ConfigMap`, Amazon EKS en Fargate lo detecta automáticamente y configura el enrutador de registros con él. Fargate utiliza una versión de AWS para Fluent Bit, una distribución conforme del cliente al servidor de Fluent Bit administrada por AWS. Para obtener más información, consulte AWS para Fluent Bit en GitHub.

El enrutador de registros le permite utilizar la amplia gama de servicios de AWS para el análisis y el almacenamiento de registros. Puede transmitir registros desde Fargate directamente a Amazon CloudWatch, Amazon OpenSearch Service. También puede transmitir registros a destinos como

Amazon S3, Amazon Kinesis Data Streams y herramientas de socios a través de Amazon Data Firehose.

- Un perfil de Fargate existente que especifica un espacio de nombres de Kubernetes existente en el que se implementan pods de Fargate. Para obtener más información, consulte the section called "Paso 3: creación de un perfil de Fargate para el clúster".
- Un rol de ejecución de pods de Fargate existente. Para obtener más información, consulte the section called "Paso 2: creación de un rol de ejecución de pods de Fargate".

Configuración del enrutador de registros

> **Important**
>
> Para que los registros se publiquen correctamente, debe haber acceso de red desde la VPC en la que se encuentra el clúster hasta el destino de los registros. Esto se aplica principalmente a los usuarios que personalizan las reglas de salida de la VPC. Para ver un ejemplo con CloudWatch, consulte la sección Uso de Registros de CloudWatch con puntos de conexión de VPC de interfaz en la Guía del usuario de los Registros de Amazon CloudWatch.

En los siguientes pasos, reemplace cada *valor de ejemplo* por valores propios.

1. Cree un espacio de nombres de Kubernetes dedicado llamado `aws-observability`.
   a. Guarde el siguiente contenido en un archivo llamado `aws-observability-namespace.yaml` en el equipo. El valor de `name` debe ser `aws-observability` y la etiqueta `aws-observability: enabled` es obligatoria.

```
kind: Namespace
apiVersion: v1
metadata:
  name: aws-observability
  labels:
    aws-observability: enabled
```

   b. Cree el espacio de nombres.

```
kubectl apply -f aws-observability-namespace.yaml
```

2. Cree un `ConfigMap` con un valor de datos de `Fluent Conf` para enviar los registros de contenedores a un destino. Fluent Conf es Fluent Bit, que es un lenguaje de configuración del procesador de registros rápido y ligero que se utiliza para dirigir los registros del contenedor a un destino de registro de su elección. Para obtener más información, consulte Archivo de configuración en la documentación de Fluent Bit.

   > **Important**
   >
   > En un `Fluent Conf` típico, las secciones principales incluidas son `Service`, `Input`, `Filter` y `Output`. Sin embargo, el enrutador de registros de Fargate solo acepta:
   >
   > - Las secciones `Filter` y `Output`.
   > - Una sección `Parser`.
   >
   > Si proporciona otras secciones, se rechazarán.

   El enrutador de registros de Fargate administra las secciones `Service` e `Input`. Tiene la siguiente sección `Input`, la cual no se puede modificar y no es necesaria en su `ConfigMap`. Sin embargo, puede obtener información a partir de ella, como el límite del búfer de memoria y la etiqueta aplicada a los registros.

   ```
   [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.*
   ```

   Al crear el `ConfigMap`, debe tener en cuenta las siguientes reglas que Fargate utiliza para validar campos:

   - Se supone que `[FILTER]`, `[OUTPUT]` y `[PARSER]` deben especificarse en cada clave correspondiente. Por ejemplo, `[FILTER]` debe estar en `filters.conf`. Puede tener uno o más `[FILTER]` en `filters.conf`. Las secciones `[OUTPUT]` y `[PARSER]` también

deben estar en sus claves correspondientes. Mediante la especificación de varias secciones `[OUTPUT]`, puede dirigir sus registros a diferentes destinos al mismo tiempo.

- Fargate valida las claves requeridas para cada sección. `Name` y `match` son necesarias para cada `[FILTER]` y `[OUTPUT]`. `Name` y `format` son necesarias para cada `[PARSER]`. Las claves distinguen entre mayúsculas y minúsculas.
- Las variables de entorno, como `${ENV_VAR}` no se permiten en `ConfigMap`.
- La sangría tiene que ser la misma para la política o el par clave-valor dentro de cada `filters.conf`, `output.conf` y `parsers.conf`. Los pares clave-valor deben tener más sangría que las políticas.
- Fargate valida con los siguientes filtros compatibles: `grep`, `parser`, `record_modifier`, `rewrite_tag`, `throttle`, `nest`, `modify` y `kubernetes`.
- Fargate verifica las siguientes salidas compatibles: `es`, `firehose`, `kinesis_firehose`, `cloudwatch`, `cloudwatch_logs` y `kinesis`.
- Se debe proporcionar al menos un complemento de `Output` compatible en el `ConfigMap` a fin de habilitar el registro. `Filter` y `Parser` no son necesarios para habilitar el registro.

También puede ejecutar Fluent Bit en Amazon EC2 con la configuración deseada para solucionar cualquier problema que surja de la validación. Cree su `ConfigMap` con uno de los siguientes ejemplos.

> **Important**
>
> El registro de Fargate de Amazon EKS no admite la configuración dinámica del `ConfigMap`. Cualquier cambio en `ConfigMap` solo se aplica a los pods nuevos. Los cambios no se aplican a los pods existentes.

Cree un `ConfigMap` con el ejemplo para el destino de registro deseado.

> **Note**
>
> También puede utilizar Amazon Kinesis Data Streams como destino para su registro. Si usa Kinesis Data Streams, asegúrese de que la función de ejecución del pod tenga otorgado el permiso `kinesis:PutRecords`. Para obtener más información, consulte Permisos de Amazon Kinesis Data Streams en Fluent Bit: Manual oficial.

Example

CloudWatch

Tiene dos opciones de salida al utilizar CloudWatch:

- Un complemento de salida escrito en C
- Un complemento de salida escrito en Golang

En el siguiente ejemplo se muestra cómo utilizar el complemento de `cloudwatch_logs` para enviar registros a CloudWatch.

a. Guarde los siguientes contenidos en un archivo llamado `aws-logging-cloudwatch-configmap.yaml`. Reemplace *`region-code`* por la región de AWS en la que se encuentra el clúster. Se requieren los parámetros en `[OUTPUT]`.

```
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. Aplique el manifiesto al clúster.

```
kubectl apply -f aws-logging-cloudwatch-configmap.yaml
```

Amazon OpenSearch Service

Si desea enviar registros a Amazon OpenSearch Service, puede utilizar la salida es, que es un complemento escrito en C. En el ejemplo siguiente se muestra cómo utilizar el complemento para enviar registros a OpenSearch.

a. Guarde los siguientes contenidos en un archivo llamado `aws-logging-opensearch-configmap.yaml`. Reemplace los *valores de ejemplo* por sus propios valores.

```
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. Aplique el manifiesto al clúster.

```
kubectl apply -f aws-logging-opensearch-configmap.yaml
```

Firehose

Tiene dos opciones de salida al enviar registros a Firehose:

- kinesis_firehose: un complemento de salida escrito en C.
- firehose: un complemento de salida escrito en Golang.

En el siguiente ejemplo se muestra cómo utilizar el complemento de `kinesis_firehose` para enviar registros a Firehose.

a. Guarde los siguientes contenidos en un archivo llamado `aws-logging-firehose-configmap.yaml`. Reemplace *`region-code`* por la región de AWS en la que se encuentra el clúster.

```
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. Aplique el manifiesto al clúster.

```
kubectl apply -f aws-logging-firehose-configmap.yaml
```

3. Configure los permisos del rol de ejecución de pods en Fargate para enviar los registros al destino correspondiente.

a. Descargue la política de IAM para el destino en su computadora.

Example

CloudWatch

Descargue la política de IAM de CloudWatch en su equipo. También puede ver la política en GitHub.

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-fluent-
logging-examples/mainline/examples/fargate/cloudwatchlogs/permissions.json
```

Amazon OpenSearch Service

Descargue la política de IAM de OpenSearch en su ordenador. También puede ver la política en GitHub.

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-
fluent-logging-examples/mainline/examples/fargate/amazon-elasticsearch/
permissions.json
```

Asegúrese de que el control de acceso de OpenSearch Dashboards esté configurado correctamente. El `all_access role` de OpenSearch Dashboards debe tener el rol de ejecución de pods de Fargate y el rol de IAM asignado. Se debe hacer el mismo mapeo para el rol de `security_manager`. Puede agregar los mapeos anteriores al seleccionar `Menu`, `Security` y `Roles` y, a continuación, seleccionar los roles respectivos. Para obtener más información, consulte ¿Cómo puedo solucionar los problemas de CloudWatch Logs para que transmita a mi dominio de Amazon ES?.

Firehose

Descargue la política de IAM de Firehose en su ordenador. También puede ver la política en GitHub.

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-fluent-
logging-examples/mainline/examples/fargate/kinesis-firehose/permissions.json
```

b. Cree una política de IAM a partir del archivo de política que descargó.

```
aws iam create-policy --policy-name eks-fargate-logging-policy --policy-document
 file://permissions.json
```

c. Adjunte la política de IAM al rol de ejecución de pods especificado para el perfil de Fargate con el siguiente comando. Reemplace *111122223333* por el ID de su cuenta. Reemplace *AmazonEKSFargatePodExecutionRole* por el rol de ejecución de pods (para obtener más información, consulte the section called “Paso 2: creación de un rol de ejecución de pods de Fargate”).

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::111122223333:policy/eks-fargate-logging-policy \
  --role-name AmazonEKSFargatePodExecutionRole
```

Compatibilidad del filtro de Kubernetes

El filtro de Kubernetes de Fluent Bit permite agregar metadatos de Kubernetes a los archivos de registro. Para obtener más información acerca del filtro, consulte Kubernetes en la documentación de Fluent Bit. Puede aplicar un filtro mediante el punto de conexión del servidor de la API.

```
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` y `Kube_Token_File` son parámetros de configuración propiedad del servicio y no deben especificarse. Amazon EKS Fargate completa estos valores.
- `Kube_Meta_Cache_TTL` es el tiempo que Fluent Bit espera hasta que se comunica con el servidor de la API para obtener los metadatos más recientes. Si `Kube_Meta_Cache_TTL` no se especifica, Amazon EKS Fargate agrega un valor predeterminado de 30 minutos para reducir la carga en el servidor de la API.

Envío de registros de procesos de Fluent Bit a su cuenta

Opcionalmente, puede enviar registros de procesos de Fluent Bit a Amazon CloudWatch mediante el siguiente `ConfigMap`. El envío de registros de procesos de Fluent Bit a CloudWatch requiere costos adicionales de ingesta y almacenamiento de registros. Reemplace *`region-code`* por la región de AWS en la que se encuentra el clúster.

```
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
```

Los registros se encuentran en CloudWatch, en la misma región de AWS que el clúster. El nombre del grupo de registros es *`my-cluster`*`-fluent-bit-logs` y el nombre del flujo de registros de Fluent Bit es `fluent-bit-`*`podname`*`-`*`pod-namespace`* .

> **Note**
>
> - Los registros de proceso se envían solo cuando el proceso de Fluent Bit se inicia de forma correcta. Si se produce un error al iniciar Fluent Bit, se pierden los registros del proceso. Solo puede enviar los registros del proceso a CloudWatch.
> - Para depurar los registros del proceso de envío en la cuenta, puede aplicar el `ConfigMap` anterior a fin de obtener los registros del proceso. El hecho de que Fluent Bit no se inicie suele deberse a que Fluent Bit no analiza ni acepta el `ConfigMap` durante el inicio.

Detención del envío de registros de procesos de Fluent Bit

El envío de registros de procesos de Fluent Bit a CloudWatch requiere costos adicionales de ingesta y almacenamiento de registros. Para excluir los registros de procesos de una configuración de `ConfigMap` existente, siga estos pasos.

1. Busque el grupo de registro de CloudWatch creado automáticamente para los registros de procesos de Fluent Bit del clúster de Amazon EKS después de habilitar el registro de Fargate. Sigue el formato *`my-cluster`*`-fluent-bit-logs`.
2. Elimine los flujos de registro de CloudWatch existentes creados para los registros de procesos de cada pod en el grupo de registros de CloudWatch.
3. Edite el `ConfigMap` y configure `flb_log_cw: "false"`.
4. Reinicie todos los pods existentes en el clúster.

Probar la aplicación

1. Implemente un pod de ejemplo.
   a. Guarde el siguiente contenido en un archivo llamado `sample-app.yaml` en el equipo.

```
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. Aplique el manifiesto al clúster.

```
kubectl apply -f sample-app.yaml
```

2. Visualización de los registros NGINX con los destinos que configuró en el `ConfigMap`.

Consideraciones sobre el tamaño

Le sugerimos que planee utilizar hasta 50 MB de memoria para el enrutador de registros. Si anticipa que su aplicación generará registros con un rendimiento muy alto, entonces debe planificar utilizar hasta 100 MB.

Solución de problemas

Para confirmar si la característica de registro está habilitada o desactivada por algún motivo, como un `ConfigMap` que no es válido y desea saber por qué no es válido, verifique los eventos de pods con `kubectl describe pod` *`pod-name`* . La salida puede incluir eventos del pod que aclaran si el registro está habilitado o no, como la siguiente salida de ejemplo.

```
[...]
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
```

Los eventos de pod son efímeros con un periodo de tiempo en función de la configuración. También puede ver las anotaciones de un pod con `kubectl describe pod` *`pod-name`* . En la anotación del pod, hay información sobre si la característica de registro está habilitada o desactivada y el motivo.

Elección de un tipo de instancia de nodo de Amazon EC2 óptimo

Amazon EC2 proporciona una amplia selección de tipos de instancias para nodos de trabajo. Cada tipo de instancia ofrece diferentes capacidades de computación, memoria y almacenamiento. Cada instancia se agrupa también en una familia de instancias en función de dichas características. Para obtener una lista, consulte Tipos de instancias disponibles en la Guía del usuario de Amazon EC2. Amazon EKS publica diferentes variaciones de las AMI de Amazon EC2 para habilitar el soporte. Para asegurarse de que el tipo de instancia que seleccione es compatible con Amazon EKS, tenga en cuenta los siguientes criterios.

- En la actualidad, las AMI de Amazon EKS no admiten la familia `mac`.
- Las AMI de Arm y las no aceleradas de Amazon EKS no admiten las familias `g3`, `g4`, `inf` y `p`.
- Las AMI aceleradas de Amazon EKS no admiten las familias `a`, `c`, `hpc`, `m` y `t`.
- Para las instancias basadas en ARM, Amazon Linux 2023 (AL2023) solo admite tipos de instancias que utilizan procesadores Graviton2 o posteriores. AL2023 no admite las instancias `A1`.

Al elegir entre los tipos de instancias admitidos por Amazon EKS, tenga en cuenta las siguientes capacidades de cada tipo.

Número de instancias de un grupo de nodos

> En general, que haya menos instancias y que sean más grandes es mejor, especialmente si tiene muchos DaemonSets. Cada instancia requiere llamadas a la API para el servidor de API, por lo que cuantas más instancias tenga, más carga tendrá el servidor de API.

Sistema operativo

> Revise los tipos de instancias admitidos para Linux, Windows y Bottlerocket. Antes de crear instancias de Windows, consulte Deploy Windows nodes on EKS clusters.

Arquitectura de hardware

> ¿Necesita x86 o Arm? Antes de implementar instancias de Arm, consulte Amazon EKS optimized Arm Amazon Linux AMIs. ¿Necesita instancias integradas en Nitro System (Linux o Windows) o que tengan capacidades aceleradas? Si necesita capacidades aceleradas, solo puede utilizar Linux con Amazon EKS.

Número máximo de pods

Dado que a cada pod se le asigna su propia dirección IP, la cantidad de direcciones IP admitidas por un tipo de instancia es un factor que se considera a la hora de determinar el número de pods que se pueden ejecutar en la instancia. Para determinar manualmente cuántos pods admite un tipo de instancia, consulte the section called “Número máximo de pods recomendado por Amazon EKS para cada tipo de instancia de Amazon EC2”.

> **Note**
>
> Si utiliza una AMI de Amazon Linux 2 optimizada para Amazon EKS, `v20220406` o posterior, puede utilizar un nuevo tipo de instancia sin actualizar a la última AMI. Para estas AMI, la AMI calcula automáticamente el valor de `max-pods` necesario si no se incluye en el archivo eni-max-pods.txt. Es posible que Amazon EKS no admita los tipos de instancias que se encuentran en vista previa de forma predeterminada. Aún se deben agregar valores para `max-pods` para estos tipos a `eni-max-pods.txt` en nuestra AMI.

Los tipos de instancia AWS Nitro System admiten opcionalmente más direcciones IP que los tipos de instancias que no son Nitro System. Sin embargo, no todas las direcciones IP asignadas a una instancia están disponibles para los pods. Para asignar un número significativamente mayor de direcciones IP a sus instancias, debe tener la versión `1.9.0` o posterior del complemento Amazon VPC CNI instalada en el clúster y configurada de forma adecuada. Para obtener más información, consulte the section called “Aumento de las direcciones IP”. Para asignar el mayor número de direcciones IP a sus instancias, debe tener la versión `1.10.1` o posterior del complemento Amazon VPC CNI instalada en su clúster, e implementar este con la familia `IPv6`.

Familia de IP

Puede usar cualquier tipo de instancia compatible cuando utilice la familia `IPv4` para un clúster, que permite que su clúster asigne direcciones `IPv4` privadas a sus pods y servicios. Pero si desea usar la familia `IPv6` para su clúster, entonces debe usar tipos de instancias AWS Nitro System o tipos de ejemplares bare metal. Solo se admite `IPv4` en las instancias de Windows. Su clúster debe ejecutar la versión `1.10.1` o posterior del complemento Amazon VPC CNI. Para obtener más información acerca del uso de `IPv6`, consulte the section called “IPv6”.

Versión del complemento CNI de Amazon VPC que ejecuta

La versión más reciente del complemento CNI de Amazon VPC para Kubernetes es compatible con estos tipos de instancias. Es posible que tenga que actualizar la versión del complemento

CNI de Amazon VPC para aprovechar los últimos tipos de instancia admitidos. Para obtener más información, consulte the section called “CNI de Amazon VPC”. La última versión admite las características más recientes para el uso con Amazon EKS. Las versiones anteriores no admiten todas las características. Puede ver las características compatibles con las distintas versiones en Changelog en GitHub.

Región de AWS en la que va a crear los nodos

No todos los tipos de instancias están disponibles en todas las regiones de AWS.

Si utiliza grupos de seguridad para pods

Si utiliza grupos de seguridad para pods, solo se admiten tipos de instancia específicos. Para obtener más información, consulte the section called “Grupos de seguridad de pods”.

Número máximo de pods recomendado por Amazon EKS para cada tipo de instancia de Amazon EC2

Dado que a cada pod se le asigna su propia dirección IP, la cantidad de direcciones IP admitidas por un tipo de instancia es un factor que se considera a la hora de determinar el número de pods que se pueden ejecutar en la instancia. Amazon EKS proporciona un script que puede descargar y ejecutar para determinar el número máximo de pods recomendado por Amazon EKS para ejecutar en cada tipo de instancia. El script utiliza los atributos de hardware de cada instancia y las opciones de configuración para determinar el número máximo de pods. Puede utilizar el número devuelto en estos pasos para habilitar capacidades como la asignación de direcciones IP a pods desde una subred diferente a la de la instancia y el aumento significativo del número de direcciones IP de la instancia. Si utiliza un grupo de nodos administrado con varios tipos de instancias, utiliza un valor que funcione para todos los tipos de instancias.

1. Descargue un script que pueda utilizar para calcular el número máximo de pods para cada tipo de instancia.

```
curl -O https://raw.githubusercontent.com/awslabs/amazon-eks-ami/master/templates/
al2/runtime/max-pods-calculator.sh
```

2. Marque el script como ejecutable en el equipo.

```
chmod +x max-pods-calculator.sh
```

3. Ejecute el script, mediante el reemplazo de *`m5.large`* por el tipo de instancia que planea implementar y *`1.9.0-eksbuild.1`* por su versión del complemento CNI de Amazon VPC. Para determinar la versión del complemento, consulte los procedimientos de actualización en Asignar direcciones IP a pods con CNI de Amazon VPC.

```
./max-pods-calculator.sh --instance-type m5.large --cni-version 1.9.0-eksbuild.1
```

Un ejemplo de salida sería el siguiente.

```
29
```

Puede agregar las siguientes opciones al script para ver el número máximo de pods admitido cuando se utilizan capacidades opcionales.

- `--cni-custom-networking-enabled`: utilice esta opción si desea asignar direcciones IP desde una subred distinta a la de su instancia. Para obtener más información, consulte the section called “Redes personalizadas”. La adición de esta opción al script anterior con los mismos valores de ejemplo produce `20`.
- `--cni-prefix-delegation-enabled`: utilice esta opción cuando desee asignar un número significativamente mayor de direcciones IP a cada interfaz de red elástica. Esta capacidad requiere una instancia de Amazon Linux que se ejecute en Nitro System y en la versión `1.9.0` o posterior del complemento CNI de Amazon VPC. Para obtener más información, consulte the section called “Aumento de las direcciones IP”. La adición de esta opción al script anterior con los mismos valores de ejemplo produce `110`.

También puede ejecutar el script con la opción `--help` para ver todas las opciones disponibles.

> **Note**
>
> El script para el cálculo del máximo de pods limita el valor devuelto a `110` en función de los umbrales de escalabilidad de Kubernetes y la configuración recomendada. Si su tipo de instancia tiene más de 30 vCPU, este límite aumenta a `250`, un número basado en las pruebas internas del equipo de escalabilidad de Amazon EKS. Para obtener más información, consulte la entrada del blog Complemento CNI de Amazon VPC que aumenta los límites de pods por nodo.

Consideraciones para el modo automático de EKS

El modo automático de EKS limita la cantidad de pods en los nodos al menor de los siguientes valores:

- Límite fijo de 110 pods
- El resultado del cálculo de pods máximos descrito anteriormente.

Creación de nodos con imágenes optimizadas precompiladas

Puede implementar nodos con imágenes de máquina de Amazon (AMI) optimizadas para Amazon EKS prediseñadas o con AMI personalizadas propias al utilizar grupos de nodos administrados o nodos autoadministrados. Si ejecuta nodos híbridos, consulte the section called “Cómo preparar el sistema operativo”. A fin de obtener información acerca de cada tipo de AMI optimizada para Amazon EKS, consulte uno de los siguientes temas. Para obtener instrucciones sobre cómo crear su propia AMI personalizada, consulte the section called “Creación de una AMI de Amazon Linux optimizada para EKS personalizada”.

Con el modo automático de Amazon EKS, EKS administra la instancia de EC2, incluida la selección y actualización de la AMI.

Temas

- Guía sobre las características de transición de las AMI EKS AL2 y AL2-Accelerated
- Creación de nodos con AMI de Amazon Linux optimizadas
- Creación de nodos con AMI de Bottlerocket optimizadas
- Creación de nodos con AMI de Ubuntu Linux optimizadas
- Creación de nodos con AMI de Windows optimizadas

Guía sobre las características de transición de las AMI EKS AL2 y AL2-Accelerated

> **Warning**
>
> Amazon EKS dejó de publicar las AMI optimizadas para EKS de Amazon Linux 2 (AL2) el 26 de noviembre de 2025. Las AMI basadas en AL2023 y Bottlerocket para Amazon EKS

están disponibles para todas las versiones de Kubernetes compatibles, lo que incluye la versión 1.33 y posteriores.

AWS finalizará el soporte para las AMI EKS AL2 y AL2-Accelerated a partir del 26 de noviembre de 2025. Aunque aún podrá utilizar las AMI EKS AL2 después de la fecha de fin de soporte (EOS) (26 de noviembre de 2025), EKS dejará de publicar nuevas versiones de Kubernetes o actualizaciones de las AMI AL2, incluso versiones secundarias, revisiones y correcciones de errores después de esta fecha. Recomendamos actualizar a las AMI de Amazon Linux 2023 (AL2023) o Bottlerocket:

- AL2023 ofrece un enfoque de seguridad desde el diseño con políticas de seguridad previamente configuradas, SELinux en modo permisivo, modo exclusivo IMDSv2 habilitado de forma predeterminada, tiempos de arranque optimizados y administración de paquetes mejorada para aumentar la seguridad y el rendimiento, lo que resulta idóneo para infraestructuras que exigen personalizaciones significativas, como acceso directo a nivel de sistema operativo o cambios amplios en los nodos. Para obtener más información, consulte las Preguntas frecuentes sobre AL2023 o nuestra guía de migración detallada en the section called “Actualización a AL2023”.
- Bottlerocket aumenta la seguridad, acelera los tiempos de arranque y reduce la superficie de ataque a fin de mejorar la eficacia gracias a su diseño específico y optimizado para contenedores, que resulta idóneo para enfoques de uso nativo de contenedores con muy pocas personalizaciones de nodos. Para obtener más información, consulte las Preguntas frecuentes sobre Bottlerocket o nuestra guía de migración detallada en the section called “Bottlerocket”.

Como alternativa, puede utilizar the section called “Creación de una AMI de Amazon Linux optimizada para EKS personalizada” hasta la fecha de finalización del soporte (26 de noviembre de 2025). Además, puede crear una AMI personalizada con una instancia base de Amazon Linux 2 hasta la fecha de finalización del soporte de Amazon Linux 2 (30 de junio de 2026).

Preguntas frecuentes sobre migración y soporte

¿Cómo se realiza la migración de AL2 a una AMI de AL2023?

Recomendamos crear e implementar un plan de migración que incluya pruebas exhaustivas de las cargas de trabajo de las aplicaciones y procedimientos de reversión documentados, y posteriormente seguir las instrucciones paso a paso que se indican en Actualización de Amazon Linux 2 a Amazon Linux 2023 en la documentación oficial de EKS.

¿Podré crear una AMI de AL2 personalizada pasada la fecha de fin de soporte (EOS) de EKS para las AMI de AL2 optimizadas para EKS?

Aunque recomendamos adoptar las AMI optimizadas para EKS, oficialmente admitidas y publicadas para AL2023 o Bottlerocket, puede crear AMI personalizadas optimizadas y aceleradas para EKS AL2 hasta la fecha de finalización del soporte de las AMI de AL2 (26 de noviembre de 2025). Como alternativa, puede crear una AMI personalizada con una instancia base de Amazon Linux 2 hasta la fecha de finalización del soporte de Amazon Linux 2 (30 de junio de 2026). Para obtener instrucciones paso a paso sobre cómo crear una AMI personalizada optimizada y acelerada para EKS AL2, consulte la sección Creación de una AMI personalizada de Amazon Linux en la documentación oficial de EKS.

¿La política de soporte de versiones de Kubernetes en EKS aplica a las distribuciones de Amazon Linux?

No. La fecha de finalización del soporte para las AMI optimizadas y aceleradas para EKS AL2 es independiente de los periodos de soporte estándar y extendido de las versiones de Kubernetes por parte de EKS. Debe migrar a AL2023 o Bottlerocket incluso si utiliza el soporte extendido de EKS.

¿Cómo afecta a la migración el cambio de cgroupv1 a cgroupv2?

La comunidad de Kubernetes trasladó el soporte para `cgroupv1` (utilizado por AL2) a modo de mantenimiento, lo que significa que no se agregarán nuevas características y solo se proporcionarán correcciones críticas de seguridad y errores importantes. Para adoptar `cgroupv2` en Kubernetes, debe garantizar la compatibilidad entre el sistema operativo, el kernel, el tiempo de ejecución de contenedores y los componentes de Kubernetes. Esto requiere una distribución de Linux que habilite `cgroupv2` de forma predeterminada, como AL2023, Bottlerocket, Red Hat Enterprise Linux (RHEL) 9 o superior, Ubuntu 22.04 o superior, o Debian 11 o superior. Estas distribuciones se entregan con versiones del kernel iguales o superiores a la 5.8, que es el requisito mínimo para el soporte de `cgroupv2` en Kubernetes. Para obtener más información, consulte Acerca de cgroup v2.

¿Qué se debe hacer si se necesita Neuron en la AMI personalizada de AL2?

No puede ejecutar de forma nativa las aplicaciones completas basadas en Neuron en AMI basadas en AL2. Para aprovechar AWS Neuron en una AMI basada en AL2, debe contenerizar las aplicaciones con un contenedor compatible con Neuron que utilice una distribución de Linux distinta a AL2 (por ejemplo, Ubuntu 22.04, Amazon Linux 2023, etc.) y luego implementar esos contenedores en una AMI basada en AL2 que tenga instalado el controlador de Neuron (`aws-neuronx-dkms`).

¿Debo cambiarme a una instancia básica de Amazon Linux 2 después de la fecha de finalización del soporte de la AMI de AL2 de EKS (26 de noviembre de 2025)?

El cambio a una instancia básica de Amazon Linux 2 carece de las optimizaciones, configuraciones de tiempo de ejecución de contenedores y personalizaciones específicas que proporcionan las AMI optimizadas y aceleradas para AL2 de EKS. Como alternativa, si debe continuar utilizando una solución basada en AL2, le recomendamos que cree una AMI personalizada con las recetas de AMI de EKS que aparecen en the section called “Creación de una AMI de Amazon Linux optimizada para EKS personalizada” o la especificación de creación de AMI de Amazon EKS. De este modo, se garantiza la compatibilidad con sus cargas de trabajo existentes y se incluyen las actualizaciones del kernel de AL2 hasta la fecha de finalización del soporte de Amazon Linux 2 (30 de junio de 2026).

Al crear una AMI de AL2 personalizada con el repositorio de GitHub de la AMI de EKS después de la fecha de finalización del soporte de la AMI de AL2 de EKS (26 de noviembre de 2025), ¿qué soporte está disponible para los paquetes de repositorios como amzn2-core y amzn2extra-docker?

La receta de la AMI de EKS que se encuentra en la especificación de creación de AMI de Amazon EKS extrae paquetes mediante YUM del software estándar de Amazon Linux 2, como amzn2-core y amzn2extra-docker. Después de la fecha de finalización del soporte de la AMI de AL2 de EKS (26 de noviembre de 2025), este software continuará siendo compatible hasta la fecha de finalización del soporte de Amazon Linux 2 (30 de junio de 2026). Tenga en cuenta que el soporte se limita a las actualizaciones del kernel durante este periodo, lo que significa que tendrá que administrar y aplicar manualmente otras actualizaciones de paquetes, parches de seguridad y cualquier dependencia ajena al kernel para mantener la seguridad y la compatibilidad.

¿Por qué es posible que las aplicaciones Java que utilizan versiones anteriores de JDK8 en Amazon EKS con AL2023 experimenten excepciones por falta de memoria (OOM) y se reinicien los pods? ¿Cómo se puede resolver esto?

Cuando se ejecutan en nodos de Amazon EKS con AL2023, las aplicaciones Java que dependen de versiones de JDK 8 anteriores a `jdk8u372` pueden provocar excepciones de OOM y reinicios de pods porque JVM no es compatible con `cgroupv2`. Este problema surge específicamente de la incapacidad de JVM de detectar los límites de memoria del contenedor mediante `cgroupv2`, la opción predeterminada en Amazon Linux 2023. Como resultado, basa la asignación de pilas en la memoria total del nodo en lugar del límite definido por el pod. Esto se debe a que `cgroupv2` cambia la ubicación de almacenamiento de los datos con límite de memoria, lo que provoca que las versiones anteriores de Java malinterpreten la memoria disponible y utilicen recursos del nodo. Algunas opciones posibles son las siguientes:

- Actualización de la versión de JDK: la actualización a `jdk8u372` o una versión posterior, o a una versión más reciente de JDK con soporte para `cgroupv2` completo, puede resolver este problema. Para obtener una lista de las versiones de Java completamente compatibles con `cgroupv2`, consulte About cgroup v2.
- Creación de una AMI personalizada: si debe continuar utilizando una solución basada en AL2, puede crear una AMI personalizada basada en AL2 (hasta el 26 de noviembre de 2025) con the section called “Creación de una AMI de Amazon Linux optimizada para EKS personalizada” o la especificación de creación de AMI de Amazon EKS. Por ejemplo, puede crear una AMI v1.33 basada en AL2 (hasta el 26 de noviembre de 2025). Amazon EKS proporcionará AMI basadas en AL2 hasta la fecha de finalización del soporte de AL2 de EKS (26 de noviembre de 2025). Después de la fecha de finalización del soporte (26 de noviembre de 2025), tendrá que crear su propia AMI.
- Activación de cgroupv1: si debe continuar utilizando `cgroupv1`, puede activar `cgroupv1` en una AMI de AL2023 de EKS. Para activarlo, ejecute `sudo grubby --update-kernel=ALL --args="systemd.unified_cgroup_hierarchy=0"` y reinicie el sistema (por ejemplo, un nodo o una instancia de EC2 que ejecute Amazon Linux 2023). De este modo, se modificarán los parámetros de arranque del sistema (por ejemplo, al agregar el parámetro del kernel “systemd.unified_cgroup_hierarchy=0” a la configuración de GRUB, que indica a systemd que debe utilizar la jerarquía de `cgroupv1` heredad) y activará `cgroupv1`. Tenga en cuenta que, cuando ejecuta este comando grubby, está reconfigurando el kernel para que se lance con `cgroupv1` activado y `cgroupv2` desactivado. Solo una de estas versiones de cgroup se utiliza para la administración activa de los recursos en un nodo. No es lo mismo que ejecutar `cgroupv2` con compatibilidad con versiones anteriores para la API de `cgroupv1`.

> **Warning**
>
> No recomendamos el uso continuo de `cgroupv1`. En su lugar, recomendamos migrar a `cgroupv2`. La comunidad de Kubernetes trasladó el soporte para `cgroupv1` (utilizado por AL2) a modo de mantenimiento, lo que significa que no se agregarán nuevas características ni actualizaciones y solo se proporcionarán correcciones críticas de seguridad y errores importantes. Se espera la eliminación total del soporte para `cgroupv1` en una versión futura, aunque aún no se ha anunciado ninguna fecha específica para esta eliminación total. Si tiene problemas con `cgroupv1`, AWS no podrá proporcionarle soporte y le recomendará que actualice a `cgroupv2`.

Compatibilidad y versiones

Versiones de Kubernetes compatibles con las AMI de AL2

La versión 1.32 de Kubernetes será la última para la cual Amazon EKS publicará AMI basadas en AL2 (Amazon Linux 2). Para las versiones de Kubernetes compatibles hasta la 1.32, EKS continuará publicando AMI de AL2 (AL2_ARM_64, AL2_x86_64) y AMI aceleradas de AL2 (AL2_x86_64_GPU) hasta el 26 de noviembre de 2025. Después de esta fecha, EKS dejará de publicar AMI optimizadas y aceleradas de AL2 para todas las versiones de Kubernetes. Tenga en cuenta que la fecha de finalización del soporte para las AMI optimizadas y aceleradas de AL2 en EKS es independiente de los periodos de soporte estándar y extendido de las versiones de Kubernetes por parte de EKS.

Comparación de controladores compatibles y versiones del kernel de Linux para las AMI de AL2, AL2023 y Bottlerocket

Componente	AMI de AL2 de EKS	AMI de AL2023 de EKS	AMI de Bottlerocket de EKS
Compatibilidad con el sistema operativo base	RHEL7/CentOS 7	Fedora/CentOS 9	N/A
Controlador de modo de usuario CUDA	12.x	12.x,13.x	12.x,13.x
Controlador de GPU NVIDIA	R570	R580	R570, R580
Controlador de AWS Neuron	2.20+	2.20+	2.20+
Kernel de Linux	5.10	6.1, 6.12	6.1, 6.12

Para obtener más información sobre la compatibilidad de CUDA y los controladores de NVIDIA, consulte la documentación de NVIDIA.

Compatibilidad de AWS Neuron con las AMI de AL2

A partir de la versión 2.20 de AWS Neuron, el entorno de ejecución de Neuron (`aws-neuronx-runtime-lib`) utilizado por las AMI basadas en AL para EKS ya no es compatible con Amazon Linux 2 (AL2). El controlador de Neuron (`aws-neuronx-dkms`) es ahora el único paquete de AWS Neuron que admite Amazon Linux 2. Esto significa que no puede ejecutar las aplicaciones con tecnología de Neuron de forma nativa en una AMI basada en AL2. Para configurar Neuron en AMI de AL2023, consulte la guía de configuración de AWS Neuron.

Compatibilidad de Kubernetes con las AMI de AL2

La comunidad de Kubernetes ha puesto el soporte para `cgroupv1` (utilizado por AL2) en modo de mantenimiento. Esto significa que no se agregarán nuevas características y solo se proporcionarán correcciones críticas de seguridad y de errores importantes. Cualquier característica de Kubernetes que dependa de cgroupv2, como MemoryQoS y el aislamiento mejorado de recursos, no está disponible en AL2. Además, la versión 1.32 de Kubernetes en Amazon EKS fue la última en admitir AMI basadas en AL2. Para mantener la compatibilidad con las versiones más recientes de Kubernetes, recomendamos migrar a AL2023 o Bottlerocket, que habilitan `cgroupv2` de forma predeterminada.

Compatibilidad de versiones de Linux con las AMI de AL2

Amazon Linux 2 (AL2) cuenta con soporte de AWS hasta su fecha de finalización del soporte (EOS), el 30 de junio de 2026. Sin embargo, con el paso del tiempo, el respaldo de la comunidad de Linux para nuevas aplicaciones y funcionalidades en AL2 se ha reducido. Las AMI de AL2 se basan en el kernel de Linux 5.10, mientras que AL2023 utiliza el kernel de Linux 6.1. A diferencia de AL2023, AL2 cuenta con un soporte limitado por parte de la comunidad de Linux en general. Esto significa que muchos paquetes y herramientas de Linux de origen deben adaptarse para funcionar con la versión antigua del kernel de AL2; además, algunas características modernas de Linux y mejoras de seguridad no están disponibles debido a ese kernel, y muchos proyectos de código abierto han quedado obsoletos o han limitado el soporte para versiones antiguas como la 5.10.

Paquetes obsoletos no incluidos en AL2023

Algunos de los paquetes más comunes que no están incluidos o que han cambiado en AL2023 incluyen:

- Algunos paquetes binarios de origen disponibles en Amazon Linux 2 ya no lo están en Amazon Linux 2023.

- Cambios en cómo Amazon Linux admite diferentes versiones de paquetes (por ejemplo, el sistema amazon-linux-extras) en AL2023
- Los paquetes adicionales para Enterprise Linux (EPEL) no son compatibles con AL2023
- Las aplicaciones de 32 bits no son compatibles con AL2023

Para obtener más información, consulte la Comparación entre AL2 y AL2023.

Comparación de la validación FIPS entre AL2, AL2023 y Bottlerocket

Amazon Linux 2 (AL2), Amazon Linux 2023 (AL2023) y Bottlerocket ofrecen compatibilidad con el cumplimiento de los Estándares Federales de Procesamiento de Información (FIPS).

- AL2 cuenta con certificación FIPS 140-2 y AL2023 con certificación FIPS 140-3. Para habilitar el modo FIPS en AL2023, instale los paquetes necesarios en la instancia de Amazon EC2 y siga las instrucciones de configuración que se indican en Habilitación del modo FIPS en AL2023. Para obtener más información, consulte las Preguntas frecuentes de AL2023.
- Bottlerocket ofrece variantes diseñadas específicamente para FIPS, que restringen el kernel y los componentes del espacio de usuario al uso de módulos criptográficos que han sido enviados al Programa de validación de módulos criptográficos FIPS 140-3.

Registro de cambios de controladores y versiones de las AMI de EKS

Para ver la lista completa de todos los componentes de las AMI de EKS y sus versiones, consulte las Notas de la versión de las AMI de Amazon EKS en GitHub.

Creación de nodos con AMI de Amazon Linux optimizadas

Amazon Elastic Kubernetes Service (Amazon EKS) proporciona imágenes de máquina de Amazon (AMI) especializadas optimizadas para ejecutar nodos de trabajo de Kubernetes. Estas AMI de Amazon Linux (AL) optimizadas para EKS vienen preconfiguradas con componentes esenciales, como `kubelet`, Autenticador de AWS IAM y `containerd` para garantizar la seguridad y una integración sin problemas en sus clústeres. En esta guía, se detallan las versiones de AMI disponibles y describe las opciones especializadas para la computación acelerada y las arquitecturas basadas en Arm.

Consideraciones

- Puede realizar un seguimiento de los eventos de seguridad o privacidad de Amazon Linux en el Centro de seguridad de Amazon Linux seleccionando la pestaña de la versión que desee. También puede suscribirse a la fuente RSS correspondiente. Los eventos de seguridad y privacidad incluyen información general del problema, qué paquetes están afectados y cómo actualizar las instancias para corregir el problema.
- Antes de implementar una AMI de Arm o acelerada, revise la información de AMI de Amazon Linux aceleradas optimizadas para Amazon EKS y the section called “AMI de Amazon Linux de Arm optimizadas para Amazon EKS”.
- Las instancias `P2` de Amazon EC2 no son compatibles con Amazon EKS ya que requieren la versión 470 del controlador `NVIDIA` o anterior.
- A partir de la versión `1.30` o posterior, todos los grupos de nodos administrados recién creados utilizarán automáticamente AL2023 como sistema operativo de nodos de forma predeterminada.

AMI de Amazon Linux aceleradas optimizadas para Amazon EKS

Las AMI de Amazon Linux (AL) aceleradas optimizadas para Amazon EKS se crean sobre las AMI de Amazon Linux optimizadas para EKS estándar. Están configuradas para actuar como imágenes opcionales para que los nodos de Amazon EKS admitan cargas de trabajo basadas en GPU, Inferentia y Trainium.

Para obtener más información, consulte the section called “Uso de AMI de GPU de Linux de EKS”.

AMI de Amazon Linux de Arm optimizadas para Amazon EKS

Las instancias Arm ofrecen un importante ahorro de costos para aplicaciones de escalado horizontal y aplicaciones basadas en Arm, como servidores web, microservicios en contenedores, flotas de almacenamiento en caché y almacenes de datos distribuidos. Al agregar nodos de Arm al clúster, tenga en cuenta las siguientes consideraciones.

- Si el clúster se implementó antes del 17 de agosto de 2020, debe realizar una actualización única de los manifiestos complementarios de clúster críticos. Esto es para que Kubernetes pueda extraer la imagen correcta para cada arquitectura de hardware que se utilice en el clúster. Para obtener más información acerca de la actualización de complementos de clúster, consulte the section called “Paso 1: preparación para la actualización”. Si implementó el clúster a partir del 17 de agosto de 2020, significa que su CoreDNS, `kube-proxy` y los complementos CNI de Amazon VPC para los complementos de Kubernetes ya son aptos en múltiples arquitecturas.

- Las aplicaciones implementadas en los nodos de Arm deben compilarse para Arm.
- Si tiene algún DaemonSet que está implementado en un clúster existente o desea implementarlos en un clúster nuevo en el que también quiera implementar nodos de Arm, compruebe que el DaemonSet se pueda ejecutar en todas las arquitecturas de hardware del clúster.
- Puede ejecutar grupos de nodos de Arm y grupos de nodos x86 en el mismo clúster. Si lo hace, considere la posibilidad de implementar imágenes de contenedor de varias arquitecturas en un repositorio de contenedores como Amazon Elastic Container Registry y, a continuación, agregar selectores de nodo a los manifiestos para que Kubernetes sepa en qué arquitectura de hardware se puede implementar un pod. Para obtener más información, consulte Introducir una imagen de varias arquitecturas en la Guía del usuario de Amazon ECR y en la publicación del blog Introducing multi-architecture container images for Amazon ECR.

Más información

Para obtener más información sobre el uso de las AMI de Amazon Linux optimizadas para Amazon EKS, consulte las siguientes secciones:

- Para usar Amazon Linux con grupos de nodos administrados, consulte the section called “Grupos de nodos administrados”.
- Para lanzar nodos autoadministrados de Amazon Linux, consulte the section called “Obtención de los ID más recientes”.
- Para obtener información sobre la versión, consulte the section called “Obtención de información de la versión”.
- Para recuperar los ID más recientes de las AMI de Amazon Linux optimizadas para Amazon EKS, consulte the section called “Obtención de los ID más recientes”.
- Para los scripts de código abierto que se utilizan para crear las AMI optimizadas para Amazon EKS, consulte the section called “Creación de una AMI de Amazon Linux optimizada para EKS personalizada”.

Actualización de Amazon Linux 2 a Amazon Linux 2023

> **Warning**
>
> Amazon EKS dejó de publicar las AMI optimizadas para EKS de Amazon Linux 2 (AL2) el 26 de noviembre de 2025. Las AMI basadas en AL2023 y Bottlerocket para Amazon EKS

están disponibles para todas las versiones de Kubernetes compatibles, lo que incluye la versión 1.33 y posteriores.

AL2023 es un sistema operativo basado en Linux diseñado para proporcionar un entorno seguro, estable y de alto rendimiento para las aplicaciones en la nube. Es la próxima generación de Amazon Linux de Amazon Web Services y está disponible en todas las versiones compatibles de Amazon EKS.

AL2023 ofrece varias mejoras con respecto al AL2. Para obtener una comparación completa, consulte Comparación de AL2 y Amazon Linux 2023 en la Guía del usuario de Amazon Linux 2023. Se han añadido, actualizado y eliminado varios paquetes de AL2. Se recomienda encarecidamente probar las aplicaciones con AL2023 antes de realizar la actualización. Para ver una lista de todos los cambios de paquetes en AL2023, consulte Cambios de paquetes en Amazon Linux 2023 en las Notas de la versión de Amazon Linux 2023.

Además de estos cambios, debe tener en cuenta lo siguiente:

- AL2023 presenta un nuevo proceso de inicialización de nodos `nodeadm` que utiliza un esquema de configuración YAML. Si utiliza grupos de nodos autoadministrados o una AMI con una plantilla de lanzamiento, ahora tendrá que proporcionar metadatos del clúster adicionales de forma explícita cuando cree un nuevo grupo de nodos. A continuación, se muestra un ejemplo de los parámetros mínimos necesarios, en los que ahora se necesitan `apiServerEndpoint`, `certificateAuthority` y el servicio de `cidr`:

  ```
  ---
  apiVersion: node.eks.aws/v1alpha1
  kind: NodeConfig
  spec:
    cluster:
      name: my-cluster
      apiServerEndpoint: https://example.com
      certificateAuthority: Y2VydGlmaWNhdGVBdXRob3JpdHk=
      cidr: 10.100.0.0/16
  ```

  En AL2, los metadatos de estos parámetros se descubrieron a partir de la llamada a la API `DescribeCluster` de Amazon EKS. Con AL2023, este comportamiento ha cambiado, ya que la llamada a la API adicional corre el riesgo de limitarse durante los escalados verticales de nodos a gran escala. Este cambio no le afecta si utiliza grupos de nodos administrados sin una plantilla de

lanzamiento o si utiliza Karpenter. Para obtener más información sobre `certificateAuthority` y el servicio `cidr`, consulte `DescribeCluster` en la Referencia de la API de Amazon EKS.

- En el caso de AL2023, `nodeadm` también cambia el formato para aplicar los parámetros al `kubelet` para cada nodo que utilice `NodeConfigSpec`. En AL2, esto se hizo con el parámetro `--kubelet-extra-args`. Se suele utilizar para agregar etiquetas y taints a los nodos. En el siguiente ejemplo, se muestra cómo aplicar `maxPods` y `--node-labels` al nodo.

```
---
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
```

- Se requiere la versión `1.16.2` o posterior de CNI de Amazon VPC para AL2023.
- De forma predeterminada, AL2023 requiere `IMDSv2`. `IMDSv2` tiene varios beneficios que ayudan a mejorar la postura de seguridad. Utiliza un método de autenticación orientado a la sesión que requiere la creación de un token secreto en una solicitud sencilla de HTTP PUT para iniciar la sesión. El tiempo de validez de un token de sesión puede oscilar entre 1 segundo y 6 horas. Para obtener más información sobre cómo realizar la transición de `IMDSv1` a `IMDSv2`, consulte Transición a la versión 2 del servicio de metadatos de instancias y Cómo aprovechar todos los beneficios de IMDSv2 e inhabilitar IMDSv1 en toda la infraestructura de AWS. Si desea utilizar `IMDSv1`, puede hacerlo si anula de manera manual la configuración mediante las propiedades de inicio de la opción de metadatos de la instancia.

> **Note**
>
> Para el `IMDSv2` con AL2023, el recuento de saltos predeterminado para los grupos de nodos administrados puede variar:
>
> - Cuando no se utiliza una plantilla de lanzamiento, el valor predeterminado es 1. Esto significa que los contenedores no tendrán acceso a las credenciales del nodo mediante

IMDS. Si necesita acceso del contenedor a las credenciales del nodo, puede hacerlo usando una plantilla de lanzamiento personalizada de Amazon EC2.

- Cuando se utiliza una AMI personalizada en una plantilla de lanzamiento, el valor predeterminado de `HttpPutResponseHopLimit` es 2. Puede anular manualmente el `HttpPutResponseHopLimit` en la plantilla de lanzamiento.

Como alternativa, puede utilizar Pod Identity de Amazon EKS para proporcionar credenciales en lugar de `IMDSv2`.

- AL2023 presenta la siguiente generación de jerarquías de grupos de control unificados (`cgroupv2`). `cgroupv2` se utiliza para implementar un tiempo de ejecución de contenedores y por `systemd`. Si bien AL2023 sigue incluyendo un código que puede hacer que el sistema funcione mediante `cgroupv1`, esta configuración no se recomienda ni se admite. Esta configuración se eliminará por completo en una futura versión importante de Amazon Linux.
- Se requiere una versión de `eksctl` `0.176.0` o superior para que `eksctl` sea compatible con AL2023.

En el caso de los grupos de nodos administrados que existían con anterioridad, puede realizar una actualización local o una actualización azul/verde, según cómo utilice la plantilla de lanzamiento:

- Si utiliza una AMI personalizada con un grupo de nodos administrado, puede realizar una actualización local si intercambia el ID de la AMI en la plantilla de lanzamiento. Debe asegurarse de que las aplicaciones y cualquier dato de usuario se transfieran primero a AL2023 antes de llevar a cabo esta estrategia de actualización.
- Si utiliza grupos de nodos administrados con la plantilla de lanzamiento estándar o con una plantilla de lanzamiento personalizada que no especifica el ID de la AMI, deberá actualizar mediante una estrategia azul/verde. Una actualización azul/verde suele ser más compleja e implica la creación de un grupo de nodos completamente nuevo en el que se especificará AL2023 como tipo de AMI. Luego, será necesario configurar con cuidado el nuevo grupo de nodos para garantizar que todos los datos personalizados del grupo de nodos de AL2 sean compatibles con el nuevo sistema operativo. Una vez que el nuevo grupo de nodos se haya probado y validado con sus aplicaciones, podrá migrar pods del grupo de nodos anterior al nuevo grupo de nodos. Una vez completada la migración, puede eliminar el grupo de nodos anterior.

Si utiliza Karpenter y quiere utilizar AL2023, deberá modificar el campo de `EC2NodeClass` `amiFamily` con AL2023. De forma predeterminada, la desviación está habilitada en Karpenter.

Esto significa que, una vez que se haya cambiado el campo `amiFamily`, Karpenter actualizará automáticamente los nodos de trabajo a la AMI más reciente cuando esté disponible.

Obtención de información acerca de la versión de la AMI de Amazon Linux

Las AMI de Amazon Linux optimizadas para Amazon EKS están versionadas por la versión de Kubernetes y la fecha de lanzamiento de la AMI en el siguiente formato:

```
k8s_major_version.k8s_minor_version.k8s_patch_version-release_date
```

Cada versión de AMI incluye varias versiones de kubelet, el kernel de Linux y containerd. Las AMI aceleradas también incluyen varias versiones del controlador de NVIDIA. Puede encontrar la información de esta versión en Changelog en GitHub.

Obtención de los ID de AMI de Amazon Linux recomendados

Al implementar nodos, puede especificar un ID de una imagen de máquina de Amazon (AMI) optimizada para Amazon EKS previamente creada. Para recuperar un ID de AMI que se ajuste a la configuración deseada, consulte la API del almacén de parámetros de AWS Systems Manager. Al utilizar esta API, se elimina la necesidad de buscar manualmente los ID de AMI optimizados para Amazon EKS. Para obtener más información, consulte GetParameter. La entidad principal de IAM que utiliza debe tener el permiso `ssm:GetParameter` de IAM para recuperar los metadatos de la AMI optimizada para Amazon EKS.

Puede recuperar el ID de imagen de la última AMI de Amazon Linux optimizada para Amazon EKS recomendada con el siguiente comando, que usa el parámetro secundario `image_id`. Realice las siguientes modificaciones en el comando según sea necesario y, a continuación, ejecute el comando modificado:

- Reemplace `<kubernetes-version>` por una versión compatible de Amazon EKS.
- Reemplace *`ami-type`* por una de las siguientes opciones: Para obtener más información sobre los tipos de instancias de Amazon EC2, consulte Tipos de instancias de Amazon EC2.
  - Use *`amazon-linux-2023/x86_64/standard`* para instancias basadas en Amazon Linux 2023 (AL2023) x86.
  - Use *`amazon-linux-2023/arm64/standard`* para instancias ARM AL2023, como las instancias basadas en AWS Graviton.
  - Use *`amazon-linux-2023/x86_64/nvidia`* para las instancias aprobadas más recientes de NVIDIA AL2023 basadas en x86.

  - Use *amazon-linux-2023/arm64/nvidia* para las instancias aprobadas más recientes de NVIDIA AL2023 basadas en arm64.
  - Use *amazon-linux-2023/x86_64/neuron* para las instancias más recientes de AWS Neuron AL2023.
- Reemplace <region-code> por una región de AWS compatible con Amazon EKS para la que desee el ID de la AMI.

```
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
```

A continuación, se muestra un comando de ejemplo después de reemplazar los marcadores de posición.

```
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
```

Un ejemplo de salida sería el siguiente.

```
ami-1234567890abcdef0
```

Creación de una AMI de Amazon Linux optimizada para EKS personalizada

> **Warning**
>
> Amazon EKS dejó de publicar las AMI optimizadas para EKS de Amazon Linux 2 (AL2) el 26 de noviembre de 2025. Las AMI basadas en AL2023 y Bottlerocket para Amazon EKS están disponibles para todas las versiones de Kubernetes compatibles, lo que incluye la versión 1.33 y posteriores.

Amazon EKS proporciona scripts de creación de código abierto en el repositorio de especificaciones de creación de la AMI de Amazon EKS que podrá utilizar para ver las configuraciones de kubelet, el tiempo de ejecución y el autenticador de AWS IAM para Kubernetes, así como crear su propia AMI basada en AL desde cero.

Este repositorio contiene el script de arranque para AL2 especializado y la herramienta nodeadm para AL2023 que se ejecuta en el momento del arranque. Estos scripts configuran los datos de certificado de la instancia, el punto de conexión del plano de control, el nombre del clúster, etcétera. Estos scripts se consideran el origen de información verdadera para las creaciones de la AMI optimizada para Amazon EKS, de modo que puede seguir el repositorio de GitHub para supervisar los cambios en nuestras AMI.

Al crear AMI personalizadas con las AMI optimizadas para EKS como base, no se recomienda ni se admite ejecutar una actualización del sistema operativo (por ejemplo, `dnf upgrade`) ni actualizar cualquiera de los paquetes de Kubernetes o GPU que se incluyen en las AMI optimizadas para EKS, ya que se corre el riesgo de interrumpir la compatibilidad de los componentes. Si actualiza el sistema operativo o los paquetes que se incluyen en las AMI optimizadas para EKS, se recomienda llevar a cabo pruebas exhaustivas en un entorno de desarrollo o ensayo antes de implementarlas en producción.

Al crear AMI personalizadas para instancias de GPU, se recomienda crear AMI personalizadas independientes para cada familia y generación del tipo de instancia que vaya a ejecutar. Las AMI aceleradas optimizadas para EKS instalan de forma selectiva los controladores y paquetes en tiempo de ejecución en función de la familia y generación del tipo de instancia subyacentes. Para obtener más información, consulte los scripts de AMI de EKS para la instalación y el tiempo de ejecución.

Requisitos previos

- Instalar la CLI de AWS
- Instalar HashiCorp Packer v1.9.4+
- Instalar GNU Make

Inicio rápido

En este inicio rápido, se muestran los comandos para crear una AMI personalizada en su cuenta de AWS. Para obtener más información sobre las configuraciones disponibles para personalizar la AMI, consulte las variables de plantilla en la página Amazon Linux 2023.

Requisitos previos

Instale el complemento de Amazon necesario. Por ejemplo:

```
packer plugins install github.com/hashicorp/amazon
```

Paso 1. Configure su entorno

Clone o bifurque el repositorio de AMI oficial de Amazon EKS. Por ejemplo:

```
git clone https://github.com/awslabs/amazon-eks-ami.git
cd amazon-eks-ami
```

Compruebe que Packer esté instalado:

```
packer --version
```

Paso 2. Para crear una AMI de personalizada

A continuación, se muestran ejemplos de comandos para varias AMI personalizadas.

AMI de NVIDIA AL2 básica:

```
make k8s=1.31 os_distro=al2 \
  enable_accelerator=nvidia \
  nvidia_driver_major_version=560 \
  enable_efa=true
```

AMI de NVIDIA AL2023 básica:

```
make k8s=1.31 os_distro=al2023 \
  enable_accelerator=nvidia \
  nvidia_driver_major_version=560 \
  enable_efa=true
```

AMI de Neuron AL2023 compatible con STIG:

```
make k8s=1.31 os_distro=al2023 \
  enable_accelerator=neuron \
  enable_fips=true \
  source_ami_id=ami-0abcd1234efgh5678 \
  kms_key_id=alias/aws-stig
```

Tras ejecutar estos comandos, Packer hará lo siguiente: * Lanzará una instancia temporal de Amazon EC2. * Instalará los componentes, los controladores y las configuraciones de Kubernetes. * Creará la AMI en su cuenta de AWS.

El resultado esperado debe tener el siguiente aspecto:

```
==> 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
```

Paso 3. Vea los valores predeterminados

Para ver los valores predeterminados y las opciones adicionales, ejecute el siguiente comando:

```
make help
```

Creación de nodos con AMI de Bottlerocket optimizadas

Bottlerocket es una distribución de Linux de código abierto patrocinada y respaldada por AWS. Bottlerocket está diseñado específicamente para alojar cargas de trabajo de contenedores. Con Bottlerocket, puede mejorar la disponibilidad de las implementaciones en contenedores y reducir los costos operativos mediante la automatización de las actualizaciones de la infraestructura de contenedores. Bottlerocket incluye solo el software esencial para ejecutar los contenedores, lo que mejora el uso de los recursos, reduce las amenazas a la seguridad y reduce los gastos de administración. La AMI de Bottlerocket incluyes `containerd`, `kubelet` y el Autenticador de AWS IAM. Además de los grupos de nodos administrados y los nodos autoadministrados, Bottlerocket también es compatible con Karpenter.

Ventajas

Al utilizar Bottlerocket con el clúster de Amazon EKS, tiene las siguientes ventajas:

- Mayor tiempo de actividad con un menor costo operativo y una menor complejidad de administración: Bottlerocket ocupa menos recursos, tiene tiempos de arranque más cortos y es menos vulnerable a las amenazas de seguridad que otras distribuciones de Linux. Una huella más pequeña de Bottlerocket ayuda a reducir los costos al utilizar menos recursos de almacenamiento, computación y redes.

- Seguridad mejorada gracias a las actualizaciones automáticas del sistema operativo: las actualizaciones de Bottlerocket se aplican como una sola unidad y se pueden anular si es necesario. Esto elimina el riesgo de actualizaciones dañadas o fallidas que pueden dejar el sistema inutilizable. Con Bottlerocket, las actualizaciones de seguridad se pueden aplicar automáticamente tan pronto como estén disponibles de forma mínimamente disruptiva y revertirse si se producen errores.
- Soporte premium: las versiones proporcionadas por AWS de Bottlerocket en Amazon EC2 están cubiertas por los mismos planes de AWS Support que también cubren servicios de AWS como Amazon EC2, Amazon EKS y Amazon ECR.

Consideraciones

Tenga en cuenta lo siguiente cuando utilice Bottlerocket para su tipo de AMI:

- Bottlerocket admite instancias de Amazon EC2 con procesadores `x86_64` y `arm64`.
- Bottlerocket admite instancias de Amazon EC2 con GPU. Para obtener más información, consulte the section called “Uso de AMI de GPU de Linux de EKS”.
- Las imágenes de Bottlerocket no incluyen un servidor SSH ni un intérprete de comandos. Puede utilizar métodos de acceso fuera de banda para permitir SSH. Estos métodos permiten el contenedor de administrador y superar algunos pasos de configuración de arranque con datos de usuario. Para obtener más información, consulte las siguientes secciones de Bottlerocket OS en GitHub:
  - Exploration (Exploración
  - Contenedor de administrador
  - Configuración de Kubernetes
- Bottlerocket utiliza diferentes tipos de contenedores:
  - De forma predeterminada, se habilita un contenedor de control. Este contenedor ejecuta el agente de AWS Systems Manager que puede utilizar para ejecutar comandos o iniciar sesiones de intérprete de comandos en instancias de Bottlerocket de Amazon EC2. Para obtener más información, consulte Configuración del Administrador de sesiones en la Guía del usuario de AWS.
  - Si se proporciona una clave SSH al crear el grupo de nodos, se habilita un contenedor de administración. Recomendamos utilizar el contenedor de administración solo para escenarios de desarrollo y pruebas. No recomendamos utilizarlo en entornos de producción. Para obtener más información, consulte Contenedor de administración en GitHub.

Más información

Para obtener más información sobre el uso de las AMI de Bottlerocket optimizadas para Amazon EKS, consulte las siguientes secciones:

- Para obtener más información sobre Bottlerocket, consulte la documentación de Bottlerocket.
- Para obtener recursos de información sobre la versión, consulte the section called “Obtención de información de la versión”.
- Para usar Bottlerocket con grupos de nodos administrados, consulte the section called “Grupos de nodos administrados”.
- Para lanzar nodos de Bottlerocket autoadministrados, consulte the section called “Bottlerocket”.
- Para recuperar los ID más recientes de las AMI de Bottlerocket optimizadas para Amazon EKS, consulte the section called “Obtención de los ID más recientes”.
- Para obtener más información sobre el soporte de cumplimiento, consulte the section called “Soporte de cumplimiento”.

Recuperación de información sobre la versión de la AMI de Bottlerocket

Cada versión de la AMI de Bottlerocket incluye varias versiones de kubelet, el kernel de Bottlerocket y containerd. Las variantes de AMI aceleradas también incluyen varias versiones del controlador de NVIDIA. Encontrará más información sobre esta versión en el tema sobre el sistema operativo en la documentación de Bottlerocket. En esta página, navegue hasta el subtema Información sobre la versión correspondiente.

A veces, la documentación de Bottlerocket puede retrasarse respecto a las versiones que están disponibles en GitHub. Puede encontrar una lista de los cambios de las últimas versiones en Releases en GitHub.

Recuperación de los ID de AMI de Bottlerocket recomendados

Al implementar nodos, puede especificar un ID de una imagen de máquina de Amazon (AMI) optimizada para Amazon EKS previamente creada. Para recuperar un ID de AMI que se ajuste a la configuración deseada, consulte la API del almacén de parámetros de AWS Systems Manager. Al utilizar esta API, se elimina la necesidad de buscar manualmente los ID de AMI optimizados para Amazon EKS. Para obtener más información, consulte GetParameter. La entidad principal de IAM que utiliza debe tener el permiso `ssm:GetParameter` de IAM para recuperar los metadatos de la AMI optimizada para Amazon EKS.

Puede recuperar el ID de imagen de la última AMI de Bottlerocket optimizada para Amazon EKS recomendada con el siguiente comando de la AWS CLI, que usa el parámetro secundario `image_id`. Realice las siguientes modificaciones en el comando según sea necesario y, a continuación, ejecute el comando modificado:

- Reemplace *kubernetes-version* por cualquier versión de la plataforma compatible.
- Reemplace *-flavor* por una de las siguientes opciones:
  - Elimine *-flavor* para las variantes sin GPU.
  - Use *-nvidia* para las variantes con GPU habilitadas.
  - Utilice *-fips* para las variantes habilitadas para FIPS.
- Reemplace *architecture* por una de las siguientes opciones:
  - Use *x86_64* para instancias basadas en x86.
  - Use *arm64* para instancias de ARM.
- Sustituya *region-code* por una región de AWS compatible con Amazon EKS para la que desea el ID de AMI.

```
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
```

A continuación, se muestra un comando de ejemplo después de reemplazar los marcadores de posición.

```
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
```

Un ejemplo de salida sería el siguiente.

```
ami-1234567890abcdef0
```

Cómo cumplir con los requisitos de conformidad mediante Bottlerocket

Bottlerocket cumple con las recomendaciones definidas por varias organizaciones:

- Hay un punto de referencia del CIS definido por Bottlerocket. En una configuración predeterminada, la imagen de Bottlerocket tiene la mayoría de los controles requeridos por el perfil de configuración de nivel 1 del CIS. Puede implementar los controles requeridos para un perfil de configuración de nivel 2 de CIS. Para obtener más información, consulte Validar la AMI de Bottlerocket optimizada de Amazon EKS con el punto de referencia del CIS en el blog de AWS.
- El conjunto de características optimizado y la reducción de la superficie expuesta a ataques significan que las instancias de Bottlerocket requieren menos configuración para cumplir con los requisitos de PCI DSS. El Punto de referencia de CIS para Bottlerocket es un recurso excelente para reforzar las directrices y cumple con los requisitos de estándares de configuración segura según el requisito 2.2 de PCI DSS. También puede aprovechar Fluent Bit para cumplir con sus requisitos de registro de auditorías a nivel de sistema operativo según el requisito 10.2 de PCI DSS. AWS publica instancias nuevas (con parches) de Bottlerocket periódicamente para ayudarlo a cumplir con los requisitos 6.2 de PCI DSS (para la versión 3.2.1) y 6.3.3 (para la versión 4.0).
- Bottlerocket es una característica que cumple los requisitos de la HIPAA y está autorizada para su uso con cargas de trabajo reguladas tanto para Amazon EC2 como para Amazon EKS. Para obtener más información, consulte la Referencia de servicios compatibles con HIPAA.
- Existen AMI de Bottlerocket preconfiguradas para utilizar módulos criptográficos validados por FIPS 140-3. Entre otras, el módulo criptográfico Kernel Crypto API de Amazon Linux 2023 y el módulo criptográfico AWS-LC. Para obtener más información, consulte the section called “AMI aprobadas por el FIPS de Bottlerocket”.

Prepare sus nodos de trabajo para el FIPS con las AMI aprobadas por el FIPS de Bottlerocket

El Estándar de procesamiento de la información federal (FIPS), publicación 140-3, es un estándar de los gobiernos de Estados Unidos y Canadá que especifica los requisitos de seguridad de los módulos criptográficos que protegen información confidencial. Bottlerocket facilita el cumplimiento del FIPS porque ofrece AMI con un kernel de FIPS.

Estas AMI están preconfiguradas para utilizar módulos criptográficos validados por FIPS 140-3. Entre otras, el módulo criptográfico Kernel Crypto API de Amazon Linux 2023 y el módulo criptográfico AWS-LC.

El uso de las AMI aprobadas por el FIPS de Bottlerocket hace que sus nodos de trabajo estén “preparados para el FIPS”, pero no hace que automáticamente “cumplan con el FIPS”. Para obtener más información, consulte Estándar de procesamiento de la información federal (FIPS) 140-3.

Consideraciones

- Si el clúster utiliza subredes aisladas, es posible que no se pueda acceder al punto de conexión del FIPS de Amazon ECR. Esto puede provocar que el arranque del nodo falle. Asegúrese de que la configuración de red permita el acceso a los puntos de conexión del FIPS necesarios. Para obtener más información, consulte Acceso a un recurso a través un punto de conexión de VPC en la Guía de AWS PrivateLink.
- Si su clúster usa una subred con PrivateLink, las extracciones de imágenes fallarán porque los puntos de conexión del FIPS de Amazon ECR no están disponibles a través de PrivateLink.

Creación de un grupo de nodos administrados con una AMI aprobada por el FIPS de Bottlerocket

La AMI aprobada por el FIPS de Bottlerocket viene en dos variantes para respaldar sus cargas de trabajo:

- `BOTTLEROCKET_x86_64_FIPS`
- `BOTTLEROCKET_ARM_64_FIPS`

Para crear un grupo de nodos administrado con una AMI aprobada por el FIPS de Bottlerocket, elija el tipo de AMI correspondiente durante el proceso de creación. Para obtener más información, consulte the section called “Creación”.

Para obtener más información sobre cómo seleccionar variantes compatibles con FIPS, consulte the section called “Obtención de los ID más recientes”.

Deshabilite el punto de conexión del FIPS para las regiones de AWS no compatibles

Las AMI aprobadas por el FIPS de Bottlerocket se admiten directamente en los Estados Unidos, incluidas las regiones de AWS GovCloud (EE. UU.). En las regiones de AWS en las que las AMI están disponibles, pero no se admiten directamente, puede seguir usándolas mediante la creación de un grupo de nodos administrados con una plantilla de lanzamiento.

La AMI aprobada por el FIPS de Bottlerocket se basa en el punto de conexión del FIPS de Amazon ECR durante el arranque, que generalmente no está disponible fuera de los Estados Unidos. Para usar la AMI para su kernel de FIPS en las regiones de AWS que no tienen disponible el punto de conexión del FIPS de Amazon ECR, siga estos pasos para inhabilitar el punto de conexión del FIPS:

1. Cree un nuevo archivo de configuración con el siguiente contenido o incorpórelo a su archivo de configuración existente.

```
[default]
use_fips_endpoint=false
```

1. Codifique el contenido del archivo en formato Base64.
2. En el `UserData` de la plantilla de lanzamiento, añada la siguiente cadena codificada en formato TOML:

```
[settings.aws]
config = "<your-base64-encoded-string>"
```

Para ver otros ajustes, consulte la descripción de los ajustes de Bottlerocket en GitHub.

A continuación, se muestra un ejemplo de `UserData` en una plantilla de lanzamiento:

```
[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>
```

Para obtener más información acerca de la creación de una plantilla de lanzamiento, consulte the section called “Plantillas de inicialización”.

Creación de nodos con AMI de Ubuntu Linux optimizadas

Canonical se ha asociado con Amazon EKS para crear AMI de nodo que puede utilizar en sus clústeres.

Canonical distribuye una imagen de sistema operativo de nodo de Kubernetes personalizada. Esta imagen de Ubuntu minimizada está optimizada para Amazon EKS e incluye el kernel de AWS personalizado desarrollado conjuntamente con AWS. Para obtener más información, consulte Ubuntu

en Amazon Elastic Kubernetes Service (EKS) y the section called “Ubuntu Linux”. Para obtener más información sobre la compatibilidad, consulte la sección de Software de terceros de las Preguntas frecuentes sobre AWS Premium Support.

Creación de nodos con AMI de Windows optimizadas

Las AMI de Windows optimizadas para Amazon EKS se basan en Windows Server 2019 y Windows Server 2022. Están configuradas de modo que sirvan de imagen base para los nodos de Amazon EKS. De forma predeterminada, las AMI incluyen los siguientes componentes:

- kubelet
- kube-proxy
- Autenticador de AWS IAM para Kubernetes
- csi-proxy
- containerd

> **Note**
>
> Puede realizar un seguimiento de eventos de seguridad o privacidad para Windows Server con la guía de actualización de seguridad de Microsoft.

Amazon EKS ofrece las siguientes variantes de AMI optimizadas para contenedores de Windows:

- La AMI de Windows Server 2019 Core optimizada para Amazon EKS
- AMI de Windows Server 2019 Full optimizada para Amazon EKS
- AMI de Windows Server 2022 Core optimizada para Amazon EKS
- AMI de Windows Server 2022 Full optimizada para Amazon EKS

> **Important**
>
> - La AMI de Windows Server 20H2 Core optimizada para Amazon EKS está obsoleta. No se lanzarán nuevas versiones de esta AMI.
> - Para asegurarse de que dispone de las actualizaciones de seguridad más recientes de forma predeterminada, Amazon EKS mantiene optimizada las AMI de Windows durante

los últimos cuatro meses. Cada nueva AMI estará disponible durante cuatro meses a partir del momento de su lanzamiento inicial. Después de este período, las AMI más antiguas pasan a ser privadas y ya no se podrá acceder a ellas. Recomendamos utilizar las AMI más recientes para evitar vulnerabilidades de seguridad y perder el acceso a las AMI más antiguas que hayan llegado al final de su vida útil. Si bien no podemos garantizar que podamos proporcionar acceso a las AMI que se han convertido en privadas, puede solicitar el acceso enviando un ticket a AWS Support.

Calendario de versiones

En la siguiente tabla se enumeran las fechas de lanzamiento y finalización de la compatibilidad para las versiones de Windows de Amazon EKS. Si una fecha de finalización está en blanco, significa que la versión aún es compatible.

Versión de Windows	Versión de Amazon EKS	Fin de la compatibilidad de Amazon EKS
Windows Server 2022 Core	17/10/2022	
Windows Server 2022 Full	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 Core	07/10/2019	
Windows Server 2019 Full	07/10/2019	
Windows Server 1909 Core	07/10/2019	08/12/2020

Parámetros de configuración del script de arranque

Al crear un nodo de Windows, hay un script en el nodo que permite la configuración de diferentes parámetros. Según la configuración, este script se puede encontrar en el nodo en una ubicación similar a: `C:\Program Files\Amazon\EKS\Start-EKSBootstrap.ps1`. Para especificar valores de parámetros personalizados, puede especificarlos como argumentos del script de

arranque. Por ejemplo, puede actualizar los datos de usuario en la plantilla de lanzamiento. Para obtener más información, consulte the section called “Datos de usuario de Amazon EC2”.

El script incluye los siguientes parámetros de la línea de comandos:

- `-EKSClusterName`: especifica el nombre del clúster de Amazon EKS para que este nodo de trabajo se va a unir.
- `-KubeletExtraArgs`: especifica argumentos adicionales para `kubelet` (opcional).
- `-KubeProxyExtraArgs`: especifica argumentos adicionales para `kube-proxy` (opcional).
- `-APIServerEndpoint`: especifica el punto de conexión del servidor de API del clúster de Amazon EKS (opcional). Solo válido cuando se utiliza con `-Base64ClusterCA`. Se omite llamando a `Get-EKSCluster`.
- `-Base64ClusterCA`: especifica el contenido de CA de clúster codificado en base64 (opcional). Solo válido cuando se utiliza con `-APIServerEndpoint`. Se omite llamando a `Get-EKSCluster`.
- `-DNSClusterIP`: sustituye la dirección IP que se va a utilizar para las consultas DNS dentro del clúster (opcional). El valor predeterminado es `10.100.0.10` o `172.20.0.10` en función de la dirección IP de la interfaz principal.
- `-ServiceCIDR`: anula el rango de direcciones IP del servicio de Kubernetes desde el que se direccionan los servicios del clúster. El valor predeterminado es `172.20.0.0/16` o `10.100.0.0/16` en función de la dirección IP de la interfaz principal.
- `-ExcludedSnatCIDRs`: lista de los CIDR de `IPv4` que se deben excluir de la traducción de direcciones de red de origen (SNAT). Esto implica que la IP privada del pod, que es accesible dentro de la VPC, no se traduciría a la dirección IP de la dirección `IPv4` principal de la ENI de la instancia para el tráfico saliente. De forma predeterminada, se agrega el CIDR de `IPv4` de la VPC para el nodo de Windows de Amazon EKS. Al especificar los CIDR en este parámetro también se excluyen los CIDR especificados. Para obtener más información, consulte the section called “Tráfico de salida”.

Además de los parámetros de la línea de comandos, también puede especificar algunos parámetros de variables de entorno. Cuando se especifica un parámetro de la línea de comandos, tiene prioridad sobre la variable de entorno correspondiente. Las variables de entorno deben definirse con la máquina (o sistema) como ámbito, ya que el script de arranque solo leerá variables cuyo ámbito sea la máquina.

El script tiene en cuenta las siguientes variables de entorno:

- `SERVICE_IPV4_CIDR`: consulte el parámetro de la línea de comandos `ServiceCIDR` para ver la definición.
- `EXCLUDED_SNAT_CIDRS`: debe ser una cadena separada por comas. Consulte el parámetro de la línea de comandos `ExcludedSnatCIDRs` para ver la definición.

Soporte de autenticación de gMSA

Los pods de Windows de Amazon EKS permiten diferentes tipos de autenticación de cuenta de servicio administrada por grupo (gMSA).

- Amazon EKS admite identidades de dominio de Active Directory para la autenticación. Para obtener más información sobre la gMSA unida a un dominio, consulte Windows Authentication on Amazon EKS Windows pods en el blog de AWS.
- Amazon EKS ofrece un complemento que permite que los nodos de Windows que no están unidos a un dominio recuperen credenciales de gMSA con una identidad de usuario portátil. Para obtener más información sobre la gMSA sin dominio, consulte Domainless Windows Authentication for Amazon EKS Windows pods en el blog de AWS.

Imágenes de contenedor en caché

Las AMI de Windows optimizadas para Amazon EKS poseen algunas imágenes de contenedor en caché para los tiempos de ejecución de `containerd`. Las imágenes de los contenedores se almacenan en caché al crear AMI personalizadas mediante componentes de compilación administrados por Amazon. Para obtener más información, consulte the section called “Uso del componente de compilación administrado por Amazon”.

Las siguientes imágenes de contenedores en caché son para el tiempo de ejecución de `containerd`:

- `amazonaws.com/eks/pause-windows`
- `mcr.microsoft.com/windows/nanoserver`
- `mcr.microsoft.com/windows/servercore`

Más información

Para obtener más información sobre el uso de las AMI de Windows optimizadas para Amazon EKS, consulte las siguientes secciones:

- Para obtener más información sobre la ejecución de cargas de trabajo en AMI de Windows aceleradas y optimizadas para Amazon EKS, consulte the section called “Configuración de las AMI de GPU de Windows”.
- Para usar Windows con grupos de nodos administrados, consulte the section called “Grupos de nodos administrados”.
- Para lanzar nodos de Windows autoadministrados, consulte the section called “Windows”.
- Para obtener información sobre la versión, consulte the section called “Obtención de información de la versión”.
- Para recuperar los ID más recientes de las AMI de Windows optimizadas para Amazon EKS, consulte the section called “Obtención de los ID más recientes”.
- Para utilizar el Generador de imágenes de Amazon EC2 para crear las AMI de Windows personalizadas optimizadas para Amazon EKS, consulte the section called “Compilaciones personalizadas”.
- Para conocer las prácticas recomendadas, consulte Administración de AMI de Windows optimizadas para Amazon EKS en la Guía de prácticas recomendadas de EKS.

Creación de nodos de Windows Server 2022 autoadministrados con `eksctl`

Puede usar `test-windows-2022.yaml` como referencia para crear nodos autoadministrados de Windows Server 2022. Reemplace los *`valores de ejemplo`* por sus propios valores.

> **Note**
>
> Debe usar la versión de `eksctl` 0.116.0 o una posterior para poner en funcionamiento nodos autoadministrados de Windows Server 2022.

```
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
```

Los grupos de nodos se pueden crear con el siguiente comando.

```
eksctl create cluster -f test-windows-2022.yaml
```

Recuperación de la información sobre la versión de la AMI de Windows

En este tema, se enumeran las versiones de las AMI de Windows optimizadas para Amazon EKS y sus versiones correspondientes de kubelet, containerd y csi-proxy.

Los metadatos de la AMI optimizada para Amazon EKS, incluido el ID de la AMI, de cada variante se pueden recuperar mediante programación. Para obtener más información, consulte the section called "Obtención de los ID más recientes".

Las AMI están versionadas por la versión de Kubernetes y la fecha de lanzamiento de la AMI en el siguiente formato:

```
k8s_major_version.k8s_minor_version-release_date
```

> **Note**
>
> Los grupos de nodos administrados de Amazon EKS admiten las versiones de noviembre de 2022 y posteriores de las AMI de Windows.

Para recibir notificaciones de todos los cambios en el archivo de origen de esta página de documentación específica, puede suscribirse a la siguiente URL con un lector de RSS:

```
https://github.com/awsdocs/amazon-eks-user-guide/commits/mainline/latest/ug/nodes/eks-ami-versions-windows.adoc.atom
```

AMI de Windows Server 2022 Core optimizada para Amazon EKS

En las siguientes tablas, se enumeran las versiones actuales y anteriores de la AMI de Windows Server 2022 Core optimizada para Amazon EKS.

Example

Versión de Kubernetes 1.34

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	

Versión 1.33 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	

Versión 1.32 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containerd` a `1.7.27`.
1.32-2025.05.17	1.32.5	1.7.20	1.2.1	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	

Versión 1.31 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.31-2025.06.13	1.31.9	1.7.27	1.2.1	Se actualizó containerd a 1.7.27.
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	Incluye parches para 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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.31-2024.09.10	1.31.0	1.7.20	1.1.3	

Versión 1.30 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containerd` a `1.7.27`.
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.30-2025.03.14	1.30.9	1.7.20	1.1.3	Se actualizó containerd a 1.7.20.
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	Incluye parches para 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	Incluye parches para CVE-2024-5321 .

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.30-2024.06.17	1.30.0	1.7.14	1.1.2	Se actualizó containerd a 1.7.14.
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Versión 1.29 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó containerd a 1.7.27.
1.29-2025.05.17	1.29.15	1.7.20	1.2.1	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se actualizó containerd a 1.7.20.
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	Incluye parches para 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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.29-2024.07.10	1.29.6	1.7.11	1.1.2	Incluye parches para 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	Se actualizó containerd a 1.7.11. Se actualizó kubelet a 1.29.3.
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	Se actualizó containerd a 1.6.28. Se volvieron a crear CNI y csi-proxy mediante golang 1.22.1.
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
`1.29-2024.02.06`	`1.29.0`	`1.6.25`	`1.1.2`	Se corrigió el error de que la imagen de pausa se borraba incorrectamente durante el proceso de recogida de basura de `kubelet`.

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.29-2024.01.11	1.29.0	1.6.18	1.1.2	Se excluye la actualización independiente de Windows KB5034439 en las AMI principales de Windows Server 2022. La base de conocimientos se aplica solo a las instalaciones de Windows con una partición de WinRE independiente, las cuales no se incluyen en ninguna de las AMI de Windows optimizadas para Amazon EKS.

Versión 1.28 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.28-2025.10.18	1.28.15	1.7.28	1.2.1	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.28-2025.09.13	1.28.15	1.7.28	1.2.1	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containerd` a `1.7.27`.
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	Se actualizó `containerd` a `1.7.20`.
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.28-2025-01-01	1.28.15	1.7.14	1.1.3	Incluye parches para 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	Incluye parches para CVE-2024-5321 .
1.28-2024.06.17	1.28.8	1.7.11	1.1.2	Se actualizó containerd a 1.7.11.
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	Se actualizó containerd a 1.6.28. Se actualizó kubelet a 1.28.8.

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	Se actualizó containerd a 1.6.25. Se volvieron a crear CNI y csi-proxy mediante golang 1.22.1.
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.28-2024.01.11	1.28.5	1.6.18	1.1.2	Se excluye la actualización independiente de Windows KB5034439 en las AMI principales de Windows Server 2022. La base de conocimientos se aplica solo a las instalaciones de Windows con una partición de WinRE independiente, las cuales no se incluyen en ninguna de las AMI de Windows optimizadas para Amazon EKS.
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	Incluye parches para CVE-2023-5528 .

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	Se actualizó containerd a 1.6.18. Se agregaron nuevas variables de entorno de script de arranque (SERVICE_IPV4_CIDR y EXCLUDED_SNAT_CIDRS).
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	Se ha corregido un aviso de seguridad en kubelet.
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

AMI de Windows Server 2022 Full optimizada para Amazon EKS

En las siguientes tablas, se enumeran las versiones actuales y anteriores de la AMI de Windows Server 2022 Full optimizada para Amazon EKS.

Example

Versión de Kubernetes 1.34

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	

Versión 1.33 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.33-2025.05.17	1.33.1	1.7.27	1.2.1	

Versión 1.32 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containrd` a 1.7.27
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	

Versión 1.31 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containerd` a `1.7.27`.

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Incluye parches para 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	

Versión 1.30 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containerd` a `1.7.27`.
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	Se actualizó `contianerd` a `1.7.20`.
1.30-2025.02.15	1.30.9	1.7.14	1.1.3	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Incluye parches para 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	Incluye parches para CVE-2024-5321 .
1.30-2024.06.17	1.30.0	1.7.14	1.1.2	Se actualizó containerd a 1.7.14.
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Versión 1.29 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containerd` a `1.7.27`.
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	Se actualizó `containerd` a `1.7.20`.
1.29-2025.02.15	1.29.13	1.7.14	1.1.3	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Incluye parches para 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	Incluye parches para CVE-2024-5321 .
1.29-2024.06.17	1.29.3	1.7.11	1.1.2	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	Se actualizó containerd a 1.7.11. Se actualizó kubelet a 1.29.3.
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	Se actualizó containerd a 1.6.28. Se volvieron a crear CNI y csi-proxy mediante golang 1.22.1.
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	Se corrigió el error de que la imagen de pausa se borraba incorrectamente durante el proceso de recogida de basura de kubelet.

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.29-2024.01.09	1.29.0	1.6.18	1.1.2	

Versión 1.28 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containerd` a `1.7.27`.
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.28-2025.03.14	1.28.15	1.7.20	1.1.3	Se actualizó containerd a 1.7.20.
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	Incluye parches para 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	Incluye parches para CVE-2024-5321 .

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.28-2024.06.17	1.28.8	1.7.11	1.1.2	Se actualizó containerd a 1.7.11.
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	Se actualizó containerd a 1.6.28. Se actualizó kubelet a 1.28.8.
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	Se actualizó containerd a 1.6.25. Se volvieron a crear CNI y csi-proxy mediante golang 1.22.1.
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	Incluye parches para CVE-2023-5528 .
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	Se actualizó containerd a 1.6.18. Se agregaron nuevas variables de entorno de script de arranque (SERVICE_IPV4_CIDR y EXCLUDED_SNAT_CIDRS).
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	Se ha corregido un aviso de seguridad en kubelet.
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

AMI de Windows Server 2019 Core optimizada para Amazon EKS

En las siguientes tablas, se enumeran las versiones actuales y anteriores de la AMI de Windows Server 2019 Core optimizada para Amazon EKS.

Example

Versión de Kubernetes 1.34

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	

Versión 1.33 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.33-2025.05.17	1.33.1	1.7.27	1.2.1	

Versión 1.32 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containerd` a `1.7.27`.
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	

Versión 1.31 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containerd` a `1.7.27`.

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Incluye parches para 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	

Versión 1.30 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containerd` a `1.7.27`.
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	Se actualizó `containerd` a `1.7.20`.
1.30-2025-02-15	1.30.9	1.7.14	1.1.3	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Incluye parches para 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	Incluye parches para CVE-2024-5321 .
1.30-2024.06.17	1.30.0	1.7.14	1.1.2	Se actualizó containerd a 1.7.14.
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Versión 1.29 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containerd` a `1.7.27`.
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	Se actualizó `containerd` a `1.7.20`.
1.29-2025.02.15	1.29.13	1.7.14	1.1.3	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Incluye parches para 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	Incluye parches para CVE-2024-5321 .
1.29-2024.06.17	1.29.3	1.7.11	1.1.2	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	Se actualizó containerd a 1.7.11. Se actualizó kubelet a 1.29.3.
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	Se actualizó containerd a 1.6.28. Se volvieron a crear CNI y csi-proxy mediante golang 1.22.1.
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	
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	Se corrigió el error de que la imagen de pausa se borraba incorrectamente durante el proceso de recogida de basura de kubelet.

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.29-2024.01.09	1.29.0	1.6.18	1.1.2	

Versión 1.28 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containerd` a `1.7.27`.
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.28-2025.03.14	1.28.15	1.7.20	1.1.3	Se actualizó containerd a 1.7.20.
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	Incluye parches para 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	Incluye parches para CVE-2024-5321 .

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.28-2024.06.17	1.28.8	1.7.11	1.1.2	Se actualizó containerd a 1.7.11.
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	Se actualizó containerd a 1.6.28. Se actualizó kubelet a 1.28.8.
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	Se actualizó containerd a 1.6.25. Se volvieron a crear CNI y csi-proxy mediante golang 1.22.1.
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	Incluye parches para CVE-2023-5528 .
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	Se actualizó containerd a 1.6.18. Se agregaron nuevas variables de entorno de script de arranque (SERVICE_IPV4_CIDR y EXCLUDED_SNAT_CIDRS).
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	Se ha corregido un aviso de seguridad en kubelet.
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

AMI de Windows Server 2019 Full optimizada para Amazon EKS

En las siguientes tablas, se enumeran las versiones actuales y anteriores de la AMI de Windows Server 2019 Full optimizada para Amazon EKS.

Example

Versión de Kubernetes 1.34

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	

Versión 1.33 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.33-2025.05.17	1.33.1	1.7.27	1.2.1	

Versión 1.32 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containerd` a `1.7.27`.
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	

Versión 1.31 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containerd` a `1.7.27`.

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Incluye parches para 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	

Versión 1.30 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containerd` a `1.7.27`.
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	Se actualizó `containerd` a `1.7.20`.
1.30-2025.02.15	1.30.9	1.7.14	1.1.3	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Incluye parches para 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	Incluye parches para CVE-2024-5321 .
1.30-2024.06.17	1.30.0	1.7.14	1.1.2	Se actualizó containerd a 1.7.14.
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Versión 1.29 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containerd` a `1.7.27`.
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	Se actualizó `containerd` a `1.7.20`.
1.29-2025.02.15	1.29.13	1.7.14	1.1.3	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Incluye parches para 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	Incluye parches para CVE-2024-5321 .
1.29-2024.06.17	1.29.3	1.7.11	1.1.2	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	Se actualizó containerd a 1.7.11. Se actualizó kubelet a 1.29.3.
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	Se actualizó containerd a 1.6.28. Se volvieron a crear CNI y csi-proxy mediante golang 1.22.1.
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	
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	Se corrigió el error de que la imagen de pausa se borraba incorrectamente durante el proceso de recogida de basura de kubelet.

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.29-2024.01.09	1.29.0	1.6.18	1.1.2	

Versión 1.28 de Kubernetes

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
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	Se cambiaron los registros del complemento gMSA a Eventos de Windows.
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	Se actualizó `containerd` a `1.7.27`.
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.28-2025.03.14	1.28.15	1.7.20	1.1.3	Se actualizó containerd a 1.7.20.
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	Incluye parches para 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	Incluye parches para CVE-2024-5321 .

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.28-2024.06.17	1.28.8	1.7.11	1.1.2	Se actualizó containerd a 1.7.11.
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	Se actualizó containerd a 1.6.28. Se actualizó kubelet a 1.28.8.
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	Se actualizó containerd a 1.6.25. Se volvieron a crear CNI y csi-proxy mediante golang 1.22.1.
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	

Versión de AMI	versión de kubelet	versión de containerd	versión de csi-proxy	Notas de la versión
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	Incluye parches para CVE-2023-5528 .
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	Se actualizó containerd a 1.6.18. Se agregaron nuevas variables de entorno de script de arranque (SERVICE_IPV4_CIDR y EXCLUDED_SNAT_CIDRS).
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	Se ha corregido un aviso de seguridad en kubelet.
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

Recuperación de los ID de AMI de Microsoft Windows recomendados

Al implementar nodos, puede especificar un ID de una imagen de máquina de Amazon (AMI) optimizada para Amazon EKS previamente creada. Para recuperar un ID de AMI que se ajuste a la configuración deseada, consulte la API del almacén de parámetros de AWS Systems Manager. Al utilizar esta API, se elimina la necesidad de buscar manualmente los ID de AMI optimizados para

Amazon EKS. Para obtener más información, consulte GetParameter. La entidad principal de IAM que utiliza debe tener el permiso `ssm:GetParameter` de IAM para recuperar los metadatos de la AMI optimizada para Amazon EKS.

Puede recuperar el ID de imagen de la última AMI de Windows optimizada para Amazon EKS recomendada con el siguiente comando, que usa el parámetro secundario `image_id`. Realice las siguientes modificaciones en el comando según sea necesario y, a continuación, ejecute el comando modificado:

- Reemplace *`release`* por una de las siguientes opciones:
  - Use *`2022`* para Windows Server 2022.
  - Use *`2019`* para Windows Server 2019.
- Reemplace *`installation-option`* por una de las siguientes opciones. Para obtener más información, consulte ¿Qué es la opción de instalación Server Core en Windows Server?
  - Use *`Core`* para una instalación mínima con una superficie expuesta a ataques más pequeña.
  - Use *`Full`* para incluir la experiencia de escritorio de Windows.
- Reemplace *`kubernetes-version`* por cualquier versión de la plataforma compatible.
- Sustituya *`region-code`* por una región de AWS compatible con Amazon EKS para la que desea el ID de AMI.

```
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
```

A continuación, se muestra un comando de ejemplo después de reemplazar los marcadores de posición.

```
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
```

Un ejemplo de salida sería el siguiente.

```
ami-1234567890abcdef0
```

Creación de una AMI de Windows personalizada con el Generador de imágenes

Puede utilizar el Generador de imágenes de EC2 para crear las AMI de Windows personalizadas optimizadas para Amazon EKS con una de las siguientes opciones:

- Uso de una AMI de Windows optimizada para Amazon EKS como base
- Uso del componente de compilación administrado por Amazon

Con ambos métodos, debe crear su propia receta de Image Builder. Para obtener más información, consulte Crear una nueva versión de una receta de imagen en la Guía del usuario de Image Builder.

> **Important**
>
> Los siguientes componentes administrados por Amazon para `eks` incluyen parches para `CVE-2024-5321`.
>
> - `1.28.2` y posteriores
> - `1.29.2` y posteriores
> - `1.30.1` y posteriores
> - Todas las versiones para Kubernetes 1.31 y posteriores

Uso de una AMI de Windows optimizada para Amazon EKS como base

Esta opción es la forma recomendada de crear las AMI de Windows personalizadas. Las AMI de Windows optimizadas para Amazon EKS que ofrecemos se actualizan con más frecuencia que el componente de compilación administrado por Amazon.

1. Inicie una receta nueva de Image Builder.
   a. Abra la consola de EC2 Image Builder enhttps://console.aws.amazon.com/imagebuilder.
   b. En el panel de navegación izquierdo, elija Recetas de imágenes.
   c. Seleccione Crear receta de imagen.
2. En la sección Detalles de la receta, ingrese un nombre y una versión.
3. Especifique el ID de la AMI de Windows optimizada para Amazon EKS en la sección Imagen base.
   a. Elija Escribir el ID personalizado de la AMI.

b. Recupere el ID de la AMI de la versión del sistema operativo de Windows que necesita. Para obtener más información, consulte the section called “Obtención de los ID más recientes”.

c. Ingrese el ID de la AMI personalizada. Si no encuentra el ID de la AMI, asegúrese de que la región de AWS del ID de la AMI coincida con la región de AWS que se muestra en la esquina superior derecha de la consola.

4. (Opcional) Para obtener las actualizaciones de seguridad más recientes, agrega el componente `update-windows` en la sección Componentes de compilación: .

a. En la lista desplegable situada a la derecha del cuadro de búsqueda Buscar componentes por nombre, seleccione Administrado por Amazon.

b. En el cuadro de búsqueda Buscar componentes por nombre, ingrese `update-windows`.

c. Seleccione la casilla de verificación del resultado de búsqueda **`update-windows`**. Este componente incluye los parches de Windows más recientes para el sistema operativo.

5. Complete las entradas restantes de la receta de imágenes con las configuraciones necesarias. Para obtener más información, consulte Crear una nueva versión de una receta de imagen (consola) en la Guía del usuario de Image Builder.

6. Elija Crear receta.

7. Utilice la nueva receta de imagen en una canalización de imágenes nueva o existente. Una vez que la canalización de imágenes se ejecute correctamente, la AMI personalizada aparecerá como imagen de salida y estará lista para su uso. Para obtener más información, consulte Crear una canalización de imágenes mediante el asistente de la consola de Image Builder de EC2.

Uso del componente de compilación administrado por Amazon

Si no es viable utilizar una AMI de Windows optimizada para Amazon EKS como base, puede utilizar en su lugar el componente de compilación administrado por Amazon. Esta opción puede estar retrasada con respecto a las versiones de Kubernetes compatibles más recientes.

1. Inicie una receta nueva de Image Builder.

a. Abra la consola de EC2 Image Builder enhttps://console.aws.amazon.com/imagebuilder.

b. En el panel de navegación izquierdo, elija Recetas de imágenes.

c. Seleccione Crear receta de imagen.

2. En la sección Detalles de la receta, ingrese un nombre y una versión.

3. Determine qué opción utilizará para crear su AMI personalizada en la sección Imagen base:

- Seleccione imágenes administradas: elija Windows para su sistema operativo (SO) de imágenes. A continuación, elija una de las siguientes opciones para Origen de la imagen.
  - Inicio rápido (administrado por Amazon): en el menú desplegable Nombre de la imagen, seleccione una versión de Windows Server compatible con Amazon EKS. Para obtener más información, consulte the section called “Windows”.
  - Imágenes de mi propiedad: en Nombre de la imagen, seleccione el ARN de su propia imagen con su propia licencia. La imagen que proporcione no puede tener ya instalados los componentes de Amazon EKS.
- Escribir el ID personalizado de la AMI: en el ID de la AMI, ingrese el ID de su AMI con su propia licencia. La imagen que proporcione no puede tener ya instalados los componentes de Amazon EKS.

4. En la sección Componentes de compilación: Windows, haga lo siguiente:
   a. En la lista desplegable situada a la derecha del cuadro de búsqueda Buscar componentes por nombre, seleccione Administrado por Amazon.
   b. En el cuadro de búsqueda Buscar componentes por nombre, ingrese `eks`.
   c. Seleccione el resultado de búsqueda **`eks-optimized-ami-windows`**, aunque el resultado devuelto puede no ser la versión que desea.
   d. En el cuadro de búsqueda Buscar componentes por nombre, ingrese `update-windows`.
   e. Seleccione la casilla de verificación del resultado de búsqueda update-windows. Este componente incluye los parches de Windows más recientes para el sistema operativo.
5. En la sección Componentes seleccionados, haga lo siguiente:
   a. Elija Opciones de control de versiones para **`eks-optimized-ami-windows`** .
   b. Elija Especificar la versión del componente.
   c. En el campo Versión del componente, ingrese *`version.x`* y reemplace *`version`* por una versión de Kubernetes compatible. Al ingresar una *`x`* para una parte del número de versión, se indica que se debe utilizar la versión más reciente del componente, que también se alinea con la parte de la versión que defina explícitamente. Preste atención a la salida de la consola, ya que le indicará si la versión que desea está disponible como componente administrado. Tenga en cuenta que es posible que las versiones de Kubernetes más recientes no estén disponibles para el componente de compilación. Para obtener más información sobre versiones disponibles, consulte the section called “Recuperación de información sobre las versiones de los componentes de `eks-optimized-ami-windows`”.

6. Complete las entradas restantes de la receta de imágenes con las configuraciones necesarias. Para obtener más información, consulte Crear una nueva versión de una receta de imagen (consola) en la Guía del usuario de Image Builder.
7. Elija Crear receta.
8. Utilice la nueva receta de imagen en una canalización de imágenes nueva o existente. Una vez que la canalización de imágenes se ejecute correctamente, la AMI personalizada aparecerá como imagen de salida y estará lista para su uso. Para obtener más información, consulte Crear una canalización de imágenes mediante el asistente de la consola de Image Builder de EC2.

Recuperación de información sobre las versiones de los componentes de **`eks-optimized-ami-windows`**

Puede recuperar información específica sobre lo que se instala con cada componente. Por ejemplo, puede comprobar qué versión de `kubelet` está instalada. Los componentes pasan por pruebas funcionales en las versiones de sistemas operativos Windows compatibles con Amazon EKS. Para obtener más información, consulte the section called “Calendario de versiones”. Es posible que otras versiones de sistemas operativos Windows que se indican como compatibles o que han llegado al fin del soporte técnico no sean compatibles con el componente.

1. Abra la consola de EC2 Image Builder enhttps://console.aws.amazon.com/imagebuilder.
2. En el panel de navegación izquierdo, elija Componentes.
3. En la lista desplegable situada a la derecha del cuadro de búsqueda Buscar componentes por nombre, cambie De mi propiedad a Inicio rápido (administrado por Amazon).
4. En el recuadro Find components by name (Buscar componentes por nombre), ingrese `eks`.
5. (Opcional) Si utiliza una versión reciente, seleccione dos veces la columna Versión para ordenarla en orden descendente.
6. Elija el enlace de **`eks-optimized-ami-windows`** con la versión que desee.

La descripción de la página resultante muestra la información específica.

Cómo habilitar la reparación automática de los nodos e investigar los problemas de estado de los nodos

El estado del nodo hace referencia a su funcionamiento operativo y a su capacidad para ejecutar cargas de trabajo de manera eficiente. Un nodo en buen estado mantiene la conectividad esperada,

cuenta con recursos suficientes y puede ejecutar los pods correctamente sin interrupciones. Para obtener información sobre cómo obtener detalles sobre los nodos, consulte the section called “Cómo ver el estado de los nodos” y the section called “Cómo obtener los registros de nodos”.

Para ayudar a mantener nodos en buen estado, Amazon EKS ofrece el agente de supervisión de nodos y la reparación automática de nodos.

> **Important**
>
> El agente de supervisión de nodos y la reparación automática de nodos solo están disponibles en Linux. Estas características no están disponibles en Windows.

Agente de supervisión de nodos

El agente de supervisión de nodos lee automáticamente los registros de los nodos para detectar ciertos problemas de estado. Analiza los registros de los nodos para detectar fallas y muestra diversa información de estado sobre los nodos de trabajo. Se aplica una `NodeCondition` específica a los nodos de trabajo para cada categoría de problemas detectados, como los relacionados con almacenamiento o redes. Las descripciones de los problemas de estado detectados están disponibles en el panel de observabilidad. Para obtener más información, consulte the section called “Problemas de estado de los nodos”.

El agente de supervisión de nodos se incluye como una capacidad para todos los clústeres del modo automático de Amazon EKS. Para otros tipos de clústeres, puede agregar el agente de supervisión como complemento de Amazon EKS. Para obtener más información, consulte the section called “Cómo crear un complemento”.

Reparación automática de nodos

La reparación automática de nodos es una característica adicional que supervisa continuamente el estado de los nodos, reacciona automáticamente a los problemas detectados y reemplaza los nodos cuando es posible. Esto contribuye a mantener la disponibilidad general del clúster con una intervención manual mínima. Si se produce un error en una comprobación de estado, el nodo se acordona automáticamente para que no se programen nuevos pods en este.

Por sí sola, la reparación automática de nodos puede reaccionar ante la condición `Ready` del `kubelet` y cualquier objeto de nodo que sea eliminado manualmente. Cuando se combina con el

agente de supervisión de nodos, la reparación automática de nodos puede reaccionar ante más condiciones que de otro modo no se detectarían. Algunas de estas condiciones adicionales son `KernelReady`, `NetworkingReady` y `StorageReady`.

Esta recuperación automatizada de nodos aborda automáticamente problemas intermitentes de los nodos, como fallos al unirse al clúster, kubelets no receptivos y un aumento en los errores de aceleradores (dispositivos). La confiabilidad mejorada ayuda a reducir el tiempo de inactividad de las aplicaciones y a mejorar las operaciones del clúster. La reparación automática de nodos no puede solucionar ciertos problemas que se reportan, como `DiskPressure`, `MemoryPressure` y `PIDPressure`. Amazon EKS espera 10 minutos antes de actuar sobre las `NodeConditions` `AcceleratedHardwareReady` y 30 minutos para todas las demás condiciones.

Los grupos de nodos administrados también desactivarán automáticamente las reparaciones de nodos por motivos de seguridad en dos circunstancias. Cualquier operación de reparación que ya esté en curso continuará en ambas circunstancias.

- Si el controlador de recuperación de aplicaciones (ARC) ha provocado un cambio de zona en el clúster, se detendrán todas las operaciones de reparación posteriores.
- Si el grupo de nodos tiene más de cinco nodos y más del 20 % de los nodos del grupo de nodos están en mal estado, se detendrán las operaciones de reparación.

Puede habilitar la reparación automática de nodos al crear o editar un grupo de nodos administrado.

- Cuando utilice la consola Amazon EKS, active la casilla de verificación Habilitar la reparación automática de nodos para el grupo de nodos administrado. Para obtener más información, consulte the section called “Creación”.
- Al utilizar la AWS CLI, agregue `--node-repair-config enabled=true` al comando `eks create nodegroup` o `eks update-nodegroup-config`.
- Para una `ClusterConfig` de `eksctl` que usa un grupo de nodos administrado con reparación automática de nodos, consulte 44-node-repair.yaml en GitHub.

Amazon EKS proporciona un control más detallado sobre el comportamiento de reparación automática de nodos mediante lo siguiente:

- `maxUnhealthyNodeThresholdCount` y `maxUnhealthyNodeThresholdPercentage`
  - Estos campos le permiten especificar un umbral de recuento o porcentaje de nodos en mal estado, por encima del cual se detendrán las acciones de reparación automática de nodos.

De este modo, se proporciona un mayor control sobre el “radio de acción” de las reparaciones automáticas de nodos.

  - Puede establecer el porcentaje o el recuento absoluto, pero no ambos
- `maxParallelNodesRepairedCount` y `maxParallelNodesRepairedPercentage`
  - Estos campos le permiten especificar el número máximo de nodos que se pueden reparar simultáneamente o en paralelo, expresado como un recuento o un porcentaje de todos los nodos en mal estado. Esto le permite controlar con mayor precisión el ritmo de las sustituciones de nodos.
  - Al igual que con el umbral de nodos en mal estado, puede establecer el porcentaje o el recuento absoluto, pero no ambos.
- `nodeRepairConfigOverrides`
  - Se trata de una estructura compleja que permite establecer anulaciones detalladas para acciones de reparación específicas. Estas anulaciones controlan la acción de reparación y el tiempo de demora de la reparación antes de que un nodo se considere elegible para la reparación.
  - Los campos específicos de esta estructura son:
    - `nodeMonitoringCondition`: la condición en mal estado notificada por el agente de supervisión de nodos.
    - `nodeUnhealthyReason`: el motivo por el que el agente de supervisión de nodos identificó el nodo como en mal estado.
    - `minRepairWaitTimeMins`: el tiempo mínimo (en minutos) que deben persistir la condición de reparación y el motivo de mal estado antes de que el nodo sea elegible para la reparación.
    - `repairAction`: la acción que debe llevar a cabo el sistema de reparación cuando se cumplen las condiciones anteriores.
  - Si utiliza este campo, debe especificar todos los campos de la estructura. También puede proporcionar una lista de estas anulaciones.
  - `nodeMonitoringCondition` y `nodeUnhealthyReason` son entradas de texto manuales que se configuran para indicar que se quiere desviar del comportamiento predeterminado del sistema.
  - Los campos `minRepairWaitTimeMins` y `repairAction` permiten especificar las desviaciones del comportamiento predeterminado del sistema.

Problemas de estado de los nodos

En las siguientes tablas se describen los problemas de estado de los nodos que el agente de supervisión de nodos puede detectar. Existen dos tipos de problemas:

- Condición: un problema terminal que requiere una acción correctiva, como la sustitución o el reinicio de la instancia. Cuando la reparación automática está habilitada, Amazon EKS realizará una acción de reparación, ya sea la sustitución o el reinicio del nodo. Para obtener más información, consulte the section called “Condiciones de nodos”.
- Evento: un problema temporal o una configuración de nodo subóptima. No se realizará ninguna acción de reparación automática. Para obtener más información, consulte the section called “Eventos de nodos”.

Problemas de estado de los nodos AcceleratedHardware

La condición de supervisión es `AcceleratedHardwareReady` para los problemas de la tabla siguiente que tengan una gravedad de “Estado”.

Si la reparación automática está habilitada, las acciones de reparación que aparecen en la lista comenzarán 10 minutos tras la detección del problema. Para obtener más información sobre los errores de XID, consulte Errores de Xid en la documentación sobre la implementación y administración de las GPU de NVIDIA. Para obtener más información sobre los mensajes XID individuales, consulte Comprensión de los mensajes Xid en la documentación sobre implementación y administración de GPU de NVIDIA.

Nombre	Gravedad	Descripción
DCGMDiagnosticFailure	Condición	Se produjo un error en un caso de prueba del conjunto de pruebas de diagnóstico activo de DCGM.
DCGMError	Condición	Se perdió la conexión con el proceso host del DCGM o no fue posible establecerla.

Nombre	Gravedad	Descripción
DCGMFieldError[Código]	Evento	El DCGM detectó la degradación de la GPU a través de un identificador de campo.
DCGMHealthCode[Código]	Evento	Una comprobación de estado del DCGM falló de manera no fatal.
DCGMHealthCode[Código]	Condición	Una comprobación de estado del DCGM falló de manera fatal.
NeuronDMAError	Condición	Un motor de DMA encontró un error no recuperable.
NeuronHBMUncorrectableError	Condición	Un HBM encontró un error no corregible y generó resultados incorrectos.
NeuronNCUncorrectableError	Condición	Se detectó un error de memoria incorregible en Neuron Core.
NeuronSRAMUncorrectableError	Condición	Una SRAM integrada en el chip encontró un error de paridad y generó resultados incorrectos.
NvidiaDeviceCountMismatch	Evento	La cantidad de GPU visibles a través de NVML no coincide con el número de dispositivos de NVIDIA del sistema de archivos.
NvidiaDoubleBitError	Condición	El controlador de la GPU produjo un error de doble bit.

Nombre	Gravedad	Descripción
NvidiaNCCLError	Evento	Se ha producido un error segfault en NVIDIA Collective Communications Library (`libnccl`).
NvidiaNVLinkError	Condición	El controlador de la GPU notificó errores de NVLink.
NvidiaPCIeError	Evento	Las repeticiones de PCIe se activaron para recuperarse de errores de transmisión.
NvidiaPageRetirement	Evento	El controlador de la GPU ha marcado una página de memoria para su retirada. Esto puede ocurrir si hay un único error de doble bit o si se encuentran dos errores de bit único en la misma dirección.
NvidiaPowerError	Evento	El uso de energía de las GPU superó los umbrales permitidos.
NvidiaThermalError	Evento	El estado térmico de las GPU superó los umbrales permitidos.
NvidiaXID[Código]Error	Condición	Se ha producido un error crítico de la GPU.
NvidiaXID[Code]Warning	Evento	Se ha producido un error no crítico de la GPU.

Problemas de estado de los nodos ContainerRuntime

La condición de supervisión es `ContainerRuntimeReady` para los problemas de la tabla siguiente que tengan una gravedad de “Estado”.

Nombre	Gravedad	Descripción
ContainerRuntimeFailed	Evento	El tiempo de ejecución del contenedor no pudo crear un contenedor, lo que probablem ente está relacionado con cualquier problema reportado si ocurre de manera repetida.
DeprecatedContainerdConfigu ration	Evento	Recientemente, se incorporó al nodo a través de `containerd` una imagen de contenedor con un manifiesto de imagen obsoleto (versión 2, esquema 1).
KubeletFailed	Evento	El kubelet entró en un estado de fallo.
LivenessProbeFailures	Evento	Se detectó una falla en la sonda de actividad, lo que podría indicar problemas en el código de la aplicación o valores de tiempo de espera insuficientes si ocurre de manera repetida.
PodStuckTerminating	Condición	Un pod está o estuvo atascado al intentar terminar durante un tiempo excesivo, lo que puede ser causado por errores en CRI que impiden la progresión del estado del pod.

Nombre	Gravedad	Descripción
ReadinessProbeFailures	Evento	Se detectó una falla en la sonda de preparación, lo que podría indicar problemas en el código de la aplicación o valores de tiempo de espera insuficientes si ocurre de manera repetida.
[Nombre]RepeatedRestart	Evento	Una unidad systemd se reinicia con frecuencia.
ServiceFailedToStart	Evento	No se pudo iniciar una unidad de systemd.

Problemas de estado de los nodos del núcleo

La condición de supervisión es `KernelReady` para los problemas de la tabla siguiente que tengan una gravedad de “Estado”.

Nombre	Gravedad	Descripción
AppBlocked	Evento	La tarea ha estado bloqueada durante un largo período para su programación, generalme nte debido a estar bloqueada en la entrada o salida.
AppCrash	Evento	Se colapsó una aplicación del nodo.
ApproachingKernelPidMax	Evento	La cantidad de procesos está próxima a alcanzar la cantidad máxima de PID disponibles según la configuración de `kernel.pid_max` actual.

Nombre	Gravedad	Descripción
		Una vez alcanzado este límite, no se podrán inicializar más procesos.
ApproachingMaxOpenFiles	Evento	La cantidad de archivos abiertos está próxima a la cantidad máxima de archivos abiertos posibles dada la configuración actual del núcleo. Una vez alcanzado este límite, no se podrán abrir nuevos archivos.
ConntrackExceededKernel	Evento	El seguimiento de conexione s excedió el límite máximo del núcleo, lo que impidió el establecimiento de nuevas conexiones y podría ocasionar la pérdida de paquetes.
ExcessiveZombieProcesses	Evento	Los procesos que no pueden ser completamente recuperad os se acumulan en grandes cantidades, lo que indica problemas en la aplicació n y podría llevar a alcanzar los límites de procesos del sistema.

Nombre	Gravedad	Descripción
ForkFailedOutOfPIDs	Condición	Una llamada a fork o exec ha fallado debido a que el sistema se ha quedado sin identificadores de proceso o memoria, lo cual podría ser causado por procesos zombis o agotamiento de la memoria física.
KernelBug	Evento	Se detectó un error en el núcleo y fue reportado por el propio núcleo de Linux, aunque esto a veces puede ser causado por nodos con un alto uso de CPU o memoria que provocan retrasos en el procesamiento de eventos.
LargeEnvironment	Evento	La cantidad de variables de entorno de este proceso es mayor de lo esperado, lo que se podría deber a la existenci a de muchos servicios con `enableServiceLinks` configurado en verdadero , lo que podría provocar problemas de rendimiento.
RapidCron	Evento	Un trabajo cron se ejecuta con una frecuencia inferior a cinco minutos en este nodo, lo que podría afectar el rendimiento si el trabajo consume recursos significativos.

Nombre	Gravedad	Descripción
SoftLockup	Evento	La CPU se detuvo durante un periodo determinado.

Problemas de estado de los nodos de red

La condición de supervisión es `NetworkingReady` para los problemas de la tabla siguiente que tengan una gravedad de “Estado”.

Nombre	Gravedad	Descripción
BandwidthInExceeded	Evento	Los paquetes se han puesto en cola o se han descartad o porque el ancho de banda agregado de entrada ha superado el máximo para la instancia.
BandwidthOutExceeded	Evento	Los paquetes se han puesto en cola o se han descartad o porque el ancho de banda agregado de salida ha superado el máximo para la instancia.
ConntrackExceeded	Evento	El seguimiento de conexione s excedió el límite máximo de la instancia, lo que impidió el establecimiento de nuevas conexiones, lo que podría ocasionar la pérdida de paquetes.
IPAMDInconsistentState	Evento	El estado del punto de control de IPAMD en el disco no

Nombre	Gravedad	Descripción
		refleja las IP en el tiempo de ejecución del contenedor.
IPAMDNoIPs	Evento	IPAMD se quedó sin direccion es IP.
IPAMDNotReady	Condición	IPAMD no se puede conectar al servidor de la API.
IPAMDNotRunning	Condición	El proceso de CNI de Amazon VPC no se encontró en ejecución.
IPAMDRepeatedlyRestart	Evento	Se han producido múltiples reinicios en el servicio IPAMD.
InterfaceNotRunning	Condición	Esta interfaz parece no estar en ejecución o existen problemas de red.
InterfaceNotUp	Condición	Parece que esta interfaz no está activa o que existen problemas de red.
KubeProxyNotReady	Evento	Kube-proxy no pudo ver ni enumerar los recursos.
LinkLocalExceeded	Evento	Se descartaron paquetes porque los paquetes por segundo (PPS) del tráfico hacia los servicios proxy locales excedieron el máximo de la interfaz de red.

Nombre	Gravedad	Descripción
MACAddressPolicyMisconfigured	Evento	La configuración del enlace systemd-networkd tiene un valor `MACAddressPolicy` incorrecto.
MissingDefaultRoutes	Evento	Faltan reglas de ruta predeterminadas.
MissingIPRoutes	Evento	Faltan rutas para las IP de los pods.
MissingIPRules	Evento	Faltan reglas para las IP de los pods.
MissingLoopbackInterface	Condición	La interfaz de bucle de retorno falta en esta instancia, lo que provoca fallos en los servicios que dependen de la conectividad local.
NetworkSysctl	Evento	La configuración de `sysctl` de red de este nodo es potencialmente incorrecta.
PPSExceeded	Evento	Los paquetes han sido puestos en cola o descartados porque los paquetes por segundo (PPS) bidireccionales excedieron el máximo permitido para la instancia.

Nombre	Gravedad	Descripción
PortConflict	Evento	Si un pod utiliza hostPort, puede escribir reglas de `iptables` que sobrescriban los puertos ya asignados del host, lo que podría impedir el acceso del servidor de la API al `kubelet`.
UnexpectedRejectRule	Evento	Se encontró una regla inesperada de `REJECT` o `DROP` en las `iptables`, lo que podría bloquear el tráfico esperado.

Problemas de estado en el nodo de almacenamiento

La condición de supervisión es `StorageReady` para los problemas de la tabla siguiente que tengan una gravedad de “Estado”.

Nombre	Gravedad	Descripción
EBSInstanceIOPSExceeded	Evento	Se ha superado el máximo de IOPS de la instancia.
EBSInstanceThroughputExceeded	Evento	Se ha superado el rendimiento máximo de la instancia.
EBSVolumeIOPSExceeded	Evento	Se ha superado el máximo de IOPS para un volumen de EBS concreto.
EBSVolumeThroughputExceeded	Evento	Se ha superado el rendimiento máximo de un volumen de Amazon EBS concreto.

Nombre	Gravedad	Descripción
EtcHostsMountFailed	Evento	El montaje del `/etc/hosts` generado por el kubelet falló debido al remonte de `/var/lib/kubelet/pods` por parte de userdata durante la operación del `kubelet-container` .
IODelays	Evento	Se detectó un retraso en la entrada o salida de un proceso, lo que podría indicar un aprovisionamiento insuficiente de recursos de entrada y salida si es excesivo.
KubeletDiskUsageSlow	Evento	El `kubelet` informa de un uso lento del disco al intentar acceder al sistema de archivos. Esto podría indicar que no hay suficientes entradas y salidas en el disco o problemas con el sistema de archivos.
XFSSmallAverageClusterSize	Evento	El tamaño medio del clúster de XFS es pequeño, lo que indica una fragmentación excesiva del espacio libre. De este modo, se puede impedir la creación de archivos a pesar de que haya nodos de indexación o espacio libre disponibles.

Cómo ver el estado de los nodos

En este tema se explican las herramientas y métodos que existen para supervisar el estado de los nodos en los clústeres de Amazon EKS. La información abarca condiciones, eventos y casos de detección de nodos útiles a la hora de identificar y diagnosticar problemas a nivel de nodo. Utilice los comandos y patrones descritos aquí para inspeccionar los recursos de estado de los nodos, interpretar las condiciones de estado y analizar los eventos de los nodos para la resolución de problemas operativos.

Puede obtener información sobre el estado de los nodos con comandos de Kubernetes para todos los nodos. Además, si utiliza el agente de supervisión de nodos a través del modo automático de Amazon EKS o el complemento administrado de Amazon EKS, obtendrá una mayor variedad de señales de nodos útiles para la resolución de problemas. Las descripciones de los problemas de estado detectados por el agente de supervisión de nodos también se encuentran disponibles en el panel de observabilidad. Para obtener más información, consulte the section called “Estado de los nodos”.

Condiciones de nodos

Las condiciones del nodo representan problemas en el terminal que requieren acciones correctoras, como la sustitución de la instancia o el reinicio.

Para obtener las condiciones de todos los nodos:

```
kubectl get nodes -o 'custom-
columns=NAME:.metadata.name,CONDITIONS:.status.conditions[*].type,STATUS:.status.conditions[*].
```

Para obtener las condiciones detalladas de un nodo específico

```
kubectl describe node node-name
```

Ejemplo del resultado de condición de un nodo en buen estado:

```
  - 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
```

Ejemplo de condición de un nodo en mal estado con un problema de red:

```
  - 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
```

Eventos de nodos

Los eventos de nodos indican problemas temporales o configuraciones que no son óptimas.

Para obtener todos los eventos notificados por el agente de supervisión de nodos

Si el agente de supervisión de nodos está disponible, puede ejecutar el siguiente comando.

```
kubectl get events --field-selector=reportingComponent=eks-node-monitoring-agent
```

Resultado de ejemplo:

```
LAST SEEN   TYPE      REASON       OBJECT
 MESSAGE
4s          Warning   SoftLockup   node/ip-192-168-71-251.us-west-2.compute.internal
 CPU stuck for 23s
```

Para obtener eventos de todos los nodos

```
kubectl get events --field-selector involvedObject.kind=Node
```

Para obtener eventos de un nodo específico

```
kubectl get events --field-selector involvedObject.kind=Node,involvedObject.name=node-
name
```

Para ver eventos en tiempo real

```
kubectl get events -w --field-selector involvedObject.kind=Node
```

Ejemplo de resultado de un evento:

```
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
```

Comandos de resolución de problemas habituales

```
# 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
```

Recuperación de los registros de nodos de un nodo administrado mediante kubectl y S3

Aprenda a recuperar los registros de nodos de un nodo administrado por Amazon EKS que tenga el agente de supervisión de nodos.

Requisitos previos

Asegúrese de contar con lo siguiente:

- Un clúster de Amazon EKS con el agente de supervisión de nodos existente. Para obtener más información, consulte the section called “Estado de los nodos”.
- La herramienta de línea de comandos `kubectl` instalada y configurada para comunicarse con el clúster.
- AWS CLI instalada y sesión iniciada con los permisos suficientes para crear buckets y objetos de S3.
- Una versión reciente de Python 3 instalada

- El SDK de AWS para Python 3 y Boto 3 instalado.

Paso 1: Creación de un destino de bucket de S3 (opcional)

Si aún no tiene un bucket de S3 para almacenar los registros, cree uno. Utilice el siguiente comando de AWS CLI. El bucket utiliza de manera predeterminada la lista de control de acceso `private`. Sustituya *`bucket-name`* por el nombre único que haya elegido.

```
aws s3api create-bucket --bucket <bucket-name>
```

Paso 2: Creación de una URL de S3 previamente firmada para HTTP Put

Amazon EKS devuelve los registros de nodo mediante una operación HTTP PUT a una URL especificada. En este tutorial, generaremos una URL de HTTP PUT de S3 previamente firmada.

Los registros se devolverán como un gzip tarball, con la extensión `.tar.gz`.

> **Note**
>
> Debe usar la API de AWS o un SDK para crear la URL de carga previamente firmada de S3 para que EKS cargue el archivo de registro. No puede crear una URL de carga previamente firmada de S3 mediante AWS CLI.

1. Determine en qué parte del bucket desea almacenar los registros. Por ejemplo, puede utilizar *`2024-11-12/logs1.tar.gz`* como clave.
2. Copie el siguiente código de Python en el archivo *`presign-upload.py`*. Sustituya *`<bucket-name>`* y *`<key>`*. La clave debe terminar con `.tar.gz`.

   ```
   import boto3; print(boto3.client('s3').generate_presigned_url(
      ClientMethod='put_object',
      Params={'Bucket': '[.replaceable]`<bucket-name>`', 'Key':
    '[.replaceable]`<key>`'},
      ExpiresIn=[.replaceable]`1000`
   ))
   ```
3. Ejecute el script con

```
python presign-upload.py
```

4. Anote la URL de resultado. Utilice este valor en el siguiente paso como el `http-put-destination`.

Para obtener más información, consulte Generate a presigned URL to upload a file en la documentación del AWS SDK para Python Boto3.

Paso 3: Creación del recurso de NodeDiagnostic

Identifique el nombre del nodo del que desea recopilar los registros.

Cree un manifiesto `NodeDiagnostic` que utilice el nombre del nodo como nombre del recurso y que proporcione un destino de URL de HTTP PUT.

```
apiVersion: eks.amazonaws.com/v1alpha1
kind: NodeDiagnostic
metadata:
    name: <node-name>
spec:
    logCapture:
        destination: http-put-destination
```

Aplique el manifiesto al clúster.

```
kubectl apply -f nodediagnostic.yaml
```

Para comprobar el estado de la recopilación, puede describir el recurso `NodeDiagnostic`:

- Un estado de `Success` o `SuccessWithErrors` indica que la tarea se completó y los registros se cargaron en el destino indicado (`SuccessWithErrors` indica que es posible que falten algunos registros)
- Si el estado es Error, confirme que la URL de carga esté formada correctamente y no haya caducado.

```
kubectl describe nodediagnostics.eks.amazonaws.com/<node-name>
```

Paso 4: Descarga de los registros de S3

Espere aproximadamente un minuto antes de intentar descargar los registros. A continuación, use la CLI de S3 para descargar los registros.

```
# Once NodeDiagnostic shows Success status, download the logs
aws s3 cp s3://<bucket-name>/key ./<path-to-node-logs>.tar.gz
```

Paso 5: Cómo limpiar el recurso de NodeDiagnostic

- Los recursos de `NodeDiagnostic` no se eliminan automáticamente. Deberá limpiarlos por cuenta propia después de haber obtenido los artefactos de registro

```
# Delete the NodeDiagnostic resource
kubectl delete nodediagnostics.eks.amazonaws.com/<node-name>
```

Información general sobre los Nodos híbridos de Amazon EKS

Con los Nodos híbridos de Amazon EKS, puede utilizar la infraestructura en las instalaciones y periférica como nodos en los clústeres de Amazon EKS. AWS administra el plano de control de Kubernetes alojado en AWS del clúster de Amazon EKS y usted administra los nodos híbridos que se ejecutan en los entornos en las instalaciones o periféricos. Así se unifica la administración de Kubernetes en los entornos y se delega la administración del plano de control de Kubernetes en AWS para las aplicaciones en las instalaciones y periféricas.

Los Nodos híbridos de Amazon EKS funcionan con cualquier hardware en las instalaciones o máquinas virtuales, lo que aporta la eficacia, escalabilidad y disponibilidad propias de Amazon EKS a dondequiera que las aplicaciones se ejecuten. Puede utilizar una amplia gama de características de Amazon EKS con los Nodos híbridos de Amazon EKS, incluidos complementos de Amazon EKS, Pod Identity de Amazon EKS, entradas de acceso a clústeres, información sobre clústeres y soporte ampliado para versiones de Kubernetes. Los Nodos híbridos de Amazon EKS se integran de forma nativa con los servicios de AWS, incluidos AWS Systems Manager, AWS IAM Roles Anywhere, Amazon Managed Service para Prometheus y Amazon CloudWatch para la supervisión centralizada, el registro y la administración de identidades.

Con los Nodos híbridos de Amazon EKS, no existen compromisos iniciales ni cuotas mínimas, y se cobra por hora por los recursos de vCPU de los nodos híbridos cuando están asociados a los

clústeres de Amazon EKS. Para obtener más información sobre los precios, consulte Precios de Amazon EKS.

Características

Los Nodos híbridos de EKS ofrecen las siguientes características de alto nivel:

- Plano de control de Kubernetes administrado: AWS administra el plano de control de Kubernetes alojado en AWS del clúster de EKS, mientras que su responsabildiad es administrar los nodos híbridos que se ejecutan en los entornos en las instalaciones o periféricos. Así se unifica la administración de Kubernetes en los entornos y se delega la administración del plano de control de Kubernetes en AWS para las aplicaciones en las instalaciones y periféricas. Si trasladaa el plano de control de Kubernetes a AWS, podrá conservar la capacidad existente en las instalaciones para las aplicaciones y confiar en que el plano de control de Kubernetes escalará junto con las cargas de trabajo.
- Experiencia de EKS coherente: la mayoría de las características de EKS son compatibles con los Nodos híbridos de EKS, con lo que es posible disfrutar de una experiencia coherente de EKS tanto en entornos en las instalaciones como en la nube, incluidos los complementos de EKS, Pod Identity de EKS, las entradas de acceso al clúster, la información del clúster, el soporte ampliado de versiones de Kubernetes y más. Consulte the section called “Configuración de complementos” para obtener más información sobre los complementos de EKS compatibles con los Nodos híbridos de EKS.
- Observabilidad y administración de identidades centralizadas: los Nodos híbridos de EKS se integran de forma nativa con los servicios de AWS, incluidos AWS Systems Manager, AWS IAM Roles Anywhere, Amazon Managed Service para Prometheus y Amazon CloudWatch, para ofrecer supervisión, registro y administración de identidades centralizados.
- Ampliación a la nube o incremento de capacidad en las instalaciones: un único clúster de EKS puede ejecutar nodos híbridos junto con nodos en las regiones de AWS, las Zonas locales de AWS oAWS Outposts, lo que permite una ampliación dinámica hacia la nube o el incremento de capacidad en las instalaciones en los clústeres de EKS. Consulte Consideraciones para clústeres en modo mixto para obtener más información.
- Infraestructura flexible: los Nodos híbridos de EKS permiten utilizar infraestructura propia, por lo que funcionan de manera independiente del entorno de infraestructura que se utilice para los nodos híbridos. Puede ejecutar nodos híbridos en máquinas físicas o virtuales, y en arquitecturas x86 y ARM, lo que permite migrar cargas de trabajo en las instalaciones que se ejecutan en nodos híbridos entre distintos tipos de infraestructura.

- Redes flexibles: al utilizar los nodos híbridos de EKS, la comunicación entre el plano de control de EKS y los nodos híbridos se enruta a través de la VPC y las subredes que se transmiten durante la creación del clúster, que se basa en el mecanismo existente en EKS para la conexión en red del plano de control a los nodos. Esto es flexible con respecto al método que prefiera para conectar las redes en las instalaciones a una VPC en AWS. Existen varias opciones documentadas, como AWS Site-to-Site VPN y AWS Direct Connect, o una solución de VPN propia. Además, puede elegir el método que mejor se adapte al caso de uso.

Límites

- Hasta 15 CIDR para redes de nodos remotos y 15 CIDR para redes de pods remotos por clúster.

Consideraciones

- Los Nodos híbridos de EKS se pueden utilizar con clústeres de EKS nuevos o existentes.
- Los Nodos híbridos de EKS se encuentran disponibles en todas las regiones de AWS, excepto en las regiones de AWS GovCloud (EE. UU.) y AWS China.
- Los Nodos híbridos de EKS deben disponer de una conexión fiable entre el entorno en las instalaciones y AWS. Los Nodos híbridos de EKS no son adecuados para entornos desconectados, interrumpidos, intermitentes o limitados (DDIL). Si ejecuta en un entorno DDIL, considere la posibilidad de utilizar Amazon EKS Anywhere. Consulte las prácticas recomendadas para los nodos híbridos de EKS para obtener información sobre cómo se comportan los nodos híbridos durante los escenarios de desconexión de la red.
- No se admite la ejecución de Nodos híbridos de EKS en infraestructura en la nube, incluidas las regiones de AWS, las Zonas locales de AWS, AWS Outposts o en otras nubes. Se aplicará la tarifa de nodos híbridos si ejecuta nodos híbridos en instancias de Amazon EC2.
- El cobro correspondiente a los nodos híbridos comenzará en el momento en que los nodos se unan al clúster de EKS y finalizará cuando los nodos se eliminen del clúster. Asegúrese de eliminar los nodos híbridos del clúster de EKS si no los utiliza.

Recursos adicionales

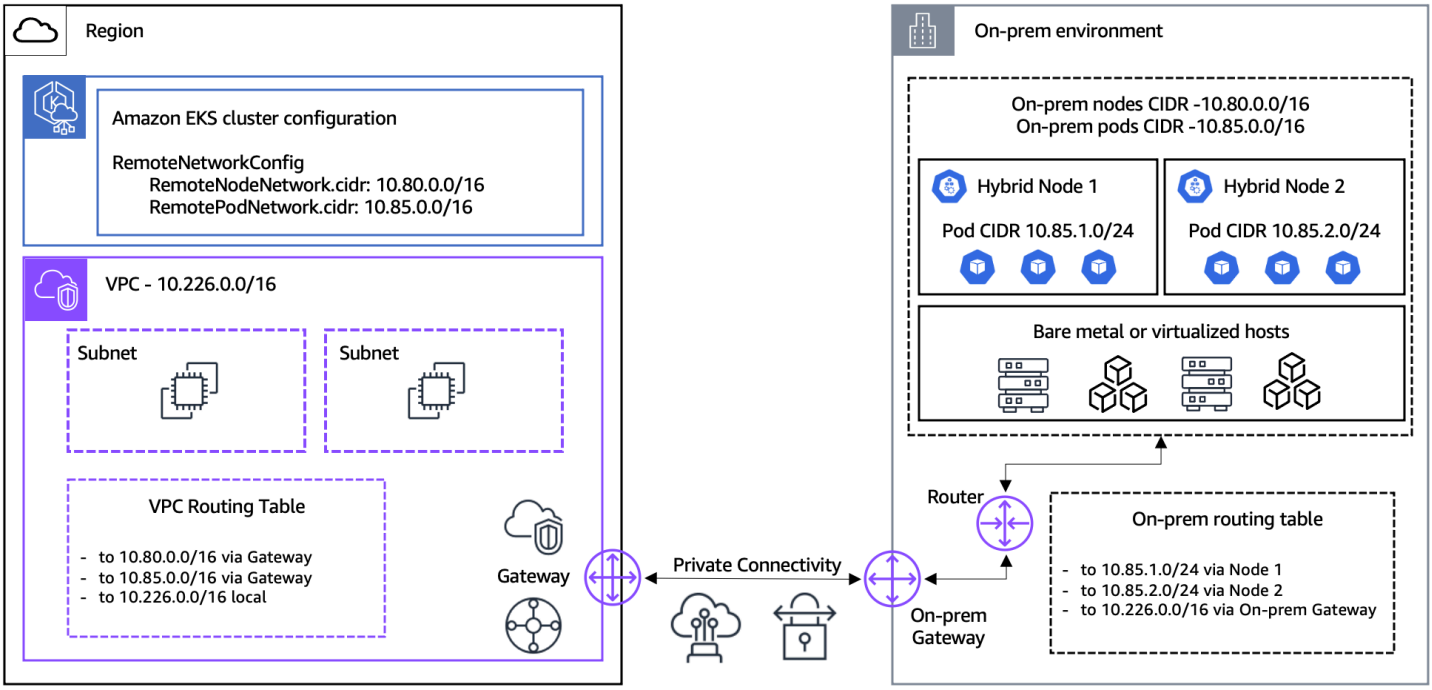
- Taller sobre nodos híbridos de EKS: instrucciones paso a paso para implementar nodos híbridos de EKS en un entorno de demostración.

- AWS re:Invent: Nodos híbridos de EKS: sesión de AWS re:Invent para presentar el lanzamiento de los nodos híbridos de EKS, en la que un cliente mostró cómo utiliza los nodos híbridos de EKS en su entorno.
- AWS re:Post: redes de clústeres para nodos híbridos de EKS: artículo en el que se explican varios métodos para configurar las redes de los nodos híbridos de EKS.
- Blog de AWS: Ejecutar la inferencia de GenAI en todos los entornos con nodos híbridos de EKS: entrada de blog que muestra cómo ejecutar la inferencia de GenAI en todos los entornos con nodos híbridos de EKS.

Configuración previa requerida para los nodos híbridos

Para usar los Nodos híbridos de Amazon EKS, debe tener conectividad privada desde el entorno en las instalaciones hacia AWS, servidores bare metal o máquinas virtuales y viceversa con un sistema operativo compatible. Además, debe configurar AWS IAM Roles Anywhere o las activaciones híbridas de AWS Systems Manager (SSM). Usted será el responsable de administrar estos requisitos previos durante todo el ciclo de vida de los nodos híbridos.

- Conectividad de red híbrida desde el entorno en las instalaciones hacia AWS y viceversa.
- Infraestructura en forma de máquinas físicas o virtuales
- Sistema operativo compatible con nodos híbridos
- Proveedor de credenciales de IAM en las instalaciones configurado



Conectividad de red híbrida

La comunicación entre el plano de control de Amazon EKS y los nodos híbridos se enruta a través de la VPC y las subredes que se transmiten durante la creación del clúster, que se basa en el mecanismo existente en Amazon EKS para la conexión en red del plano de control a los nodos. Existen varias opciones documentadas disponibles para conectar el entorno en las instalaciones con la VPC, como AWS Site-to-Site VPN, AWS Direct Connect o una conexión de VPN propia. Consulte las guías del usuario de AWS Site-to-Site VPN y AWS Direct Connect para obtener más información sobre cómo usar esas soluciones para la conexión de red híbrida.

Para disfrutar de una experiencia óptima, recomendamos una conectividad de red fiable de al menos 100 Mbps y un máximo de 200 ms de latencia de ida y vuelta para la conexión de los nodos híbridos a la región de AWS. Esta es una guía general que se adapta a la mayoría de los casos de uso, pero no es un requisito estricto. Los requisitos de ancho de banda y latencia pueden variar en función de la cantidad de nodos híbridos y de las características de la carga de trabajo, como el tamaño de la imagen de la aplicación, la elasticidad de la aplicación, las configuraciones de supervisión y registro, y las dependencias de la aplicación para acceder a datos almacenados en otros servicios de AWS. Recomendamos que realice pruebas con aplicaciones y entornos propios antes de la implementación en la fase de producción con el fin de comprobar que la configuración de red cumple los requisitos de las cargas de trabajo.

Configuración de red en las instalaciones

Debe habilitar el acceso de red entrante desde el plano de control de Amazon EKS al entorno en las instalaciones para permitir que el plano de control de Amazon EKS se comunique con el `kubelet` que se ejecuta en los nodos híbridos y, opcionalmente, con los webhooks que se ejecutan en los nodos híbridos. Además, debe habilitar el acceso de red saliente para que los nodos híbridos y los componentes que se ejecutan en estos se puedan comunicar con el plano de control de Amazon EKS. Puede configurar esta comunicación para que permanezca completamente privada a AWS Direct Connect, AWS Site-to-Site VPN o a la conexión de VPN propia.

Los rangos de enrutamiento entre dominios sin clases (CIDR) que se utilizan para las redes de nodos y pods en las instalaciones deben emplear rangos de direcciones IPv4 RFC-1918. El enrutador en las instalaciones debe estar configurado con rutas a los nodos y, opcionalmente, a los pods en las instalaciones. Consulte the section called “Configuración de redes en las instalaciones” para obtener más información sobre los requisitos de red en las instalaciones, incluida la lista completa de puertos y protocolos necesarios que deben habilitarse en el firewall y en el entorno local.

Configuración del clúster de EKS

Para minimizar la latencia, le recomendamos crear el clúster de Amazon EKS en la región de AWS más cercana al entorno en las instalaciones o periférico. Los CIDR de nodos y pods en las instalaciones, durante la creación del clúster de Amazon EKS, se transfieren a través de dos campos de la API: `RemoteNodeNetwork` y `RemotePodNetwork`. Es posible que necesite hablar con el equipo de redes en las instalaciones para identificar los CIDR de nodos y pods en las instalaciones. El CIDR de nodos se asigna desde la red en las instalaciones y el CIDR de pods se asigna desde la interfaz de red del contenedor (CNI) si se utiliza una red superpuesta para la CNI. Cilium y Calico utilizan redes superpuestas de forma predeterminada.

Los CIDR de nodos y pods en las instalaciones que configura mediante los campos `RemoteNodeNetwork` y `RemotePodNetwork` se utilizan para configurar el plano de control de Amazon EKS a fin de dirigir el tráfico a través de la VPC al `kubelet` y los pods que se ejecutan en los nodos híbridos. Los CIDR de nodos y pods en las instalaciones no se pueden superponer entre sí, con el CIDR de VPC que se transmite durante la creación del clúster ni con la configuración de IPv4 de servicio correspondiente al clúster de Amazon EKS. Además, los CIDR de los pods deben ser exclusivos de cada clúster de EKS para que el enrutador en las instalaciones pueda enrutar el tráfico.

Recomendamos utilizar acceso mediante un punto de conexión público o privado para el punto de conexión del servidor de la API de Kubernetes en Amazon EKS. Si elige “Público y privado”, el

punto de conexión del servidor de la API de Kubernetes de Amazon EKS siempre se resolverá en las IP públicas de los nodos híbridos que se ejecutan fuera de la VPC, lo que puede impedir que los nodos híbridos se unan al clúster. Cuando utiliza el acceso de punto de conexión público, el punto de conexión del servidor de la API de Kubernetes se resuelve en direcciones IP públicas y la comunicación desde los nodos híbridos al plano de control de Amazon EKS se enrutará a través de Internet. Cuando elige el acceso de punto de conexión privado, el punto de conexión del servidor de la API de Kubernetes se resuelve en direcciones IP privadas y la comunicación desde los nodos híbridos hacia el plano de control de Amazon EKS se enruta a través del enlace de conectividad privada, que en la mayoría de los casos es AWS Direct Connect o AWS Site-to-Site VPN.

Configuración de la VPC

Debe configurar la VPC que se transmite durante la creación del clúster de Amazon EKS con rutas en su tabla de enrutamiento para las redes de los nodos y, opcionalmente, de los pods en las instalaciones, mediante la puerta de enlace privada virtual (VGW) o puerta de enlace de tránsito (TGW) como destino. A continuación se muestra un ejemplo. Sustituya `REMOTE_NODE_CIDR` y `REMOTE_POD_CIDR` por los valores de la red en las instalaciones.

Destino	Objetivo	Descripción
10.226.0.0/16	local	Tráfico local a las rutas de la VPC dentro de la VPC
REMOTE_NODE_CIDR	tgw-abcdef123456	El CIDR del nodo en las instalaciones dirige el tráfico a la puerta de enlace de tránsito
REMODE_POD_CIDR	tgw-abcdef123456	El CIDR del pod en las instalaciones dirige el tráfico a la puerta de enlace de tránsito

Configuración del grupo de seguridad

Al crear un clúster, Amazon EKS crea un grupo de seguridad denominado `eks-cluster-sg-<cluster-name>-<uniqueID>`. No puede modificar las reglas de entrada de este grupo de seguridad de clúster, pero puede restringir las reglas de salida. Debe agregar un grupo de seguridad adicional al clúster para permitir que el kubelet y, opcionalmente, los webhooks que se ejecutan en

los nodos híbridos contacten al plano de control de Amazon EKS. Las reglas de entrada requeridas para este grupo de seguridad adicional se muestran a continuación. Sustituya `REMOTE_NODE_CIDR` y `REMOTE_POD_CIDR` por los valores de la red en las instalaciones.

Nombre	ID de regla del grupo de seguridad	Versión de IP	Tipo	Protocolo	Intervalo de puertos	Origen
Nodo en las instalaciones entrante	sgr-abcdef123456	IPv4	HTTPS	TCP	443	REMOTE_NODE_CIDR
Pod en las instalaciones entrante	sgr-abcdef654321	IPv4	HTTPS	TCP	443	REMOTE_POD_CIDR

Infraestructura

Debe tener servidores bare metal o máquinas virtuales disponibles para su uso como nodos híbridos. Los nodos híbridos son independientes de la infraestructura subyacente y admiten arquitecturas x86 y ARM. El servicio de Nodos Híbridos de Amazon EKS adopta un enfoque de “use su propia infraestructura”, donde usted tiene la responsabilidad de aprovisionar y administrar los servidores bare metal o las máquinas virtuales que emplea para los nodos híbridos. Si bien no existe un requisito de recursos mínimos estricto, le recomendamos utilizar hosts con al menos 1 CPU virtual y 1 GiB de RAM para los nodos híbridos.

Sistema operativo

Bottlerocket, Amazon Linux 2023 (AL2023), Ubuntu y RHEL se validan de forma continua para su uso como sistema operativo de nodo para los nodos híbridos. Bottlerocket es compatible únicamente con AWS en los entornos VMware vSphere. Los planes de AWS Support no cubren AL2023 cuando se ejecuta fuera de Amazon EC2. AL2023 solo se puede utilizar en entornos virtualizados en las instalaciones. Consulte la Guía del usuario de Amazon Linux 2023 para obtener más información.

AWS admite la integración de nodos híbridos con los sistemas operativos Ubuntu y RHEL, pero no proporciona soporte para los sistemas operativos en sí.

Usted es responsable del aprovisionamiento y la administración del sistema operativo. Al probar nodos híbridos por primera vez, lo más sencillo es ejecutar la CLI de Nodos híbridos de Amazon EKS (`nodeadm`) en un host ya aprovisionado. En el caso de implementaciones en producción, se recomienda incluir `nodeadm` en las imágenes base del sistema operativo, configurado para ejecutarse como un servicio de systemd, de modo que los hosts se unan automáticamente a los clústeres de Amazon EKS al iniciar.

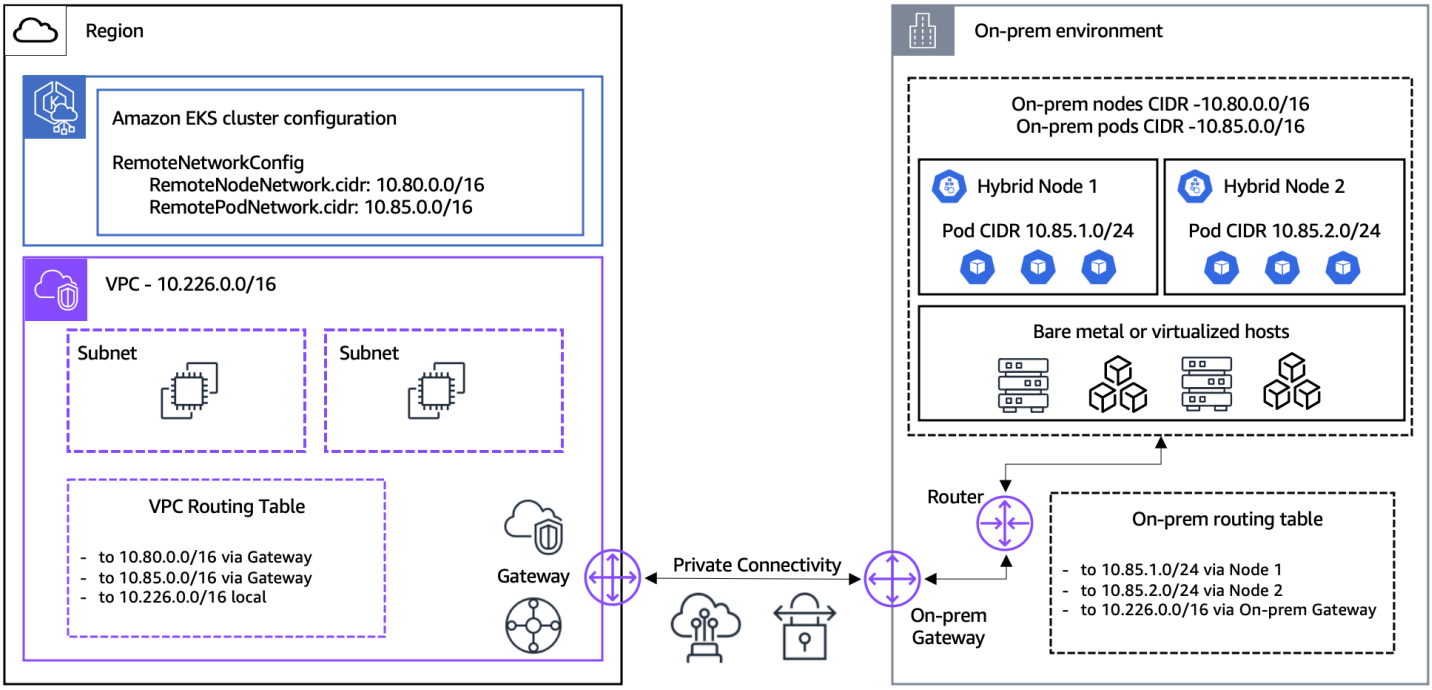
Proveedor de credenciales de IAM en las instalaciones

Los Nodos híbridos de Amazon EKS utilizan credenciales de IAM temporales aprovisionadas por activaciones híbridas de AWS SSM o AWS IAM Roles Anywhere para autenticarse con el clúster de Amazon EKS. Debe usar activaciones híbridas de AWS SSM o AWS IAM Roles Anywhere con la CLI (`nodeadm`) de los Nodos híbridos de Amazon EKS. Le recomendamos utilizar las activaciones híbridas de AWS SSM si no dispone de una infraestructura de clave pública (PKI) existente con una entidad de certificación (CA) y certificados para los entornos en las instalaciones. Si ya dispone de PKI y certificados en las instalaciones, utilice AWS IAM Roles Anywhere.

De manera similar a the section called “Rol de IAM de nodo” para los nodos que se ejecutan en Amazon EC2, deberá crear un rol de IAM para los nodos híbridos con los permisos necesarios para unir nodos híbridos a los clústeres de Amazon EKS. Si utiliza AWS IAM Roles Anywhere, configure una política de confianza que permita a AWS IAM Roles Anywhere asumir el rol de IAM de nodos híbridos y configurar el perfil de AWS IAM Roles Anywhere con el rol de IAM de nodos híbridos como rol asumible. Si utiliza AWS SSM, configure una política de confianza que permita a AWS SSM asumir el rol de IAM de nodos híbridos y crear la activación híbrida con el rol de IAM de nodos híbridos. Consulte the section called “Cómo preparar las credenciales” para averiguar cómo crear el rol de IAM de nodos híbridos con los permisos necesarios.

Cómo preparar las redes para los nodos híbridos

En este tema se ofrece información general sobre la configuración de red que se debe establecer antes de crear el clúster de Amazon EKS y conectar los nodos híbridos. En esta guía se presupone que ha cumplido los requisitos previos para la conectividad de red híbrida mediante AWS Site-to-Site VPN, AWS Direct Connect o una solución de VPN propia.



Configuración de redes en las instalaciones

Requisitos mínimos de red

Para disfrutar de una experiencia óptima, recomendamos una conectividad de red fiable de al menos 100 Mbps y un máximo de 200 ms de latencia de ida y vuelta para la conexión de los nodos híbridos a la región de AWS. Esta es una guía general que se adapta a la mayoría de los casos de uso, pero no es un requisito estricto. Los requisitos de ancho de banda y latencia pueden variar en función de la cantidad de nodos híbridos y de las características de la carga de trabajo, como el tamaño de la imagen de la aplicación, la elasticidad de la aplicación, las configuraciones de supervisión y registro, y las dependencias de la aplicación para acceder a datos almacenados en otros servicios de AWS. Recomendamos que realice pruebas con aplicaciones y entornos propios antes de la implementación en la fase de producción con el fin de comprobar que la configuración de red cumple los requisitos de las cargas de trabajo.

CIDR de nodos y pods en las instalaciones

Identifique los CIDR de nodos y pods que utilizará para los nodos híbridos y las cargas de trabajo que se ejecutan en estos. El CIDR de nodos se asigna desde la red en las instalaciones y el CIDR de pods se asigna desde la interfaz de red del contenedor (CNI) si se utiliza una red superpuesta para la CNI. Transmite los CIDR de los nodos en las instalaciones y los CIDR de los pods como entradas al crear su clúster de EKS con los campos `RemoteNodeNetwork` y `RemotePodNetwork`. Los CIDR

de los nodos en las instalaciones deben ser enrutables en su red local. Consulte la siguiente sección para obtener información sobre la enrutabilidad del CIDR del pod en las instalaciones.

Los bloques de CIDR del pod y del nodo en las instalaciones deben cumplir los siguientes requisitos:

1. Estar dentro de uno de los siguientes rangos `IPv4` RFC-1918: `10.0.0.0/8`, `172.16.0.0/12` o `192.168.0.0/16`.
2. Que no se solapen entre sí, el CIDR de la VPC del clúster de EKS o el CIDR `IPv4` del servicio de Kubernetes.

Enrutamiento de red del pod en las instalaciones

Si utiliza los nodos híbridos de EKS, generalmente recomendamos que configure los CIDR del pod en las instalaciones de manera tal que se puedan enrutar en su red local para permitir la comunicación y la funcionalidad completas del clúster entre los entornos de la nube y los que están en las instalaciones.

Redes de pods enrutables

Si puede configurar la red de los pods para que se pueda enrutar en su red en las instalaciones, siga las instrucciones que se indican a continuación.

1. Configure el campo `RemotePodNetwork` para su clúster de EKS con el CIDR del pod en las instalaciones, sus tablas de enrutamiento de VPC con el CIDR del pod en las instalaciones y el grupo de seguridad de su clúster de EKS con el CIDR del pod en las instalaciones.
2. Existen varias técnicas que puede utilizar para hacer que el CIDR de los pods en las instalaciones sea enrutable en la red en las instalaciones, como el protocolo de puerta de enlace fronteriza (BGP), rutas estáticas u otras soluciones de enrutamiento personalizadas. BGP es la solución recomendada, ya que es más escalable y fácil de administrar que las soluciones alternativas que requieren configuración manual o personalizada de rutas. AWS admite las capacidades de BGP de Cilium y Calico para anunciar los CIDR de los pods. Consulte the section called “Cómo configurar una CNI” y the section called “CIDR de pod remoto enrutable” para obtener más información.
3. Los webhooks se pueden ejecutar en nodos híbridos, ya que el plano de control de EKS puede comunicarse con las direcciones IP del pod asignadas a los webhooks.
4. Las cargas de trabajo que se ejecutan en los nodos de la nube pueden comunicarse directamente con las cargas de trabajo que se ejecutan en los nodos híbridos del mismo clúster de EKS.

5. Otros servicios de AWS, como los equilibradores de carga de aplicación de AWS y Amazon Managed Service para Prometheus, pueden comunicarse con las cargas de trabajo que se ejecutan en nodos híbridos para equilibrar el tráfico de red y analizar las métricas de los pods.

Redes de pods no enrutables

Si no puede hacer que las redes de los pods se enruten en la red en las instalaciones, siga las instrucciones que se indican a continuación.

1. Los webhooks no se pueden ejecutar en nodos híbridos, ya que requieren conectividad desde el plano de control de EKS a las direcciones IP de los pods asignadas a los webhooks. En este caso, le recomendamos que ejecute los webhooks en los nodos de la nube del mismo clúster de EKS que los nodos híbridos; consulte the section called “Configuración de webhooks” para obtener más información.
2. Las cargas de trabajo que se ejecutan en los nodos de la nube no pueden comunicarse directamente con las cargas de trabajo que se ejecutan en los nodos híbridos cuando se utiliza la CNI de la VPC para los nodos de la nube y Cilium o Calico para los nodos híbridos.
3. Utilice la distribución de tráfico del servicio para mantener el tráfico local en la zona de origen. Para obtener más información sobre la distribución de tráfico del servicio, consulte the section called “Configurar la distribución de tráfico del servicio”.
4. Configure su CNI para que utilice la máscara de salida o la traducción de direcciones de red (NAT) para el tráfico de los pods cuando salga de los hosts en las instalaciones. Esta opción está habilitada de forma predeterminada en Cilium. Calico requiere que `natOutgoing` se establezca en `true`.
5. Otros servicios de AWS, como los equilibradores de carga de aplicación de AWS y Amazon Managed Service para Prometheus, no pueden comunicarse con las cargas de trabajo que se ejecutan en nodos híbridos.

Se requiere acceso durante la instalación y actualización de nodos híbridos

Debe tener acceso a los siguientes dominios durante el proceso de instalación en el que instala las dependencias de los nodos híbridos en los hosts. Este proceso se puede realizar una vez al crear las imágenes del sistema operativo o en cada host en tiempo de ejecución. Esto incluye la instalación inicial y cuando se actualiza la versión de Kubernetes de los nodos híbridos.

Algunos paquetes se instalan mediante el administrador de paquetes predeterminado del sistema operativo. En AL2023 y RHEL, el comando `yum` se usa para instalar `containerd`,

`ca-certificates`, `iptables` y `amazon-ssm-agent`. En Ubuntu, `apt` se usa para instalar `containerd`, `ca-certificates` y `iptables`, mientras que `snap` se usa para instalar `amazon-ssm-agent`.

Componente	URL	Protocolo	Puerto
Artefactos de nodos de EKS (S3)	https://hybrid-assets.eks.amazonaws.com	HTTPS	443
Puntos de conexión de servicio de EKS	https://eks.*region*.amazonaws.com	HTTPS	443
Puntos de conexión de servicio de ECR	https://api.ecr.*region*.amazonaws.com	HTTPS	443
Puntos de conexión de ECR de EKS	Consulte the section called “Visualización de los registros de imágenes de Amazon” para conocer los puntos de conexión regionales.	HTTPS	443
Punto de conexión binario SSM [1]	https://amazon-ssm-*region*.s3.*region*.amazonaws.com	HTTPS	443
Punto de conexión de servicio de SSM [1]	https://ssm.*region*.amazonaws.com	HTTPS	443
Punto de conexión binario de IAM Anywhere [2]	https://rolesanywhere.amazonaws.com	HTTPS	443

Componente	URL	Protocolo	Puerto
Punto de conexión de servicio de IAM Anywhere [2]	https://rolesanywhere.*region*.amazonaws.com	HTTPS	443
Puntos de conexión del administrador de paquetes del sistema operativo	Los puntos de conexión del repositorio de paquetes son específicos del sistema operativo y pueden variar según la región geográfica.	HTTPS	443

Note

[1] El acceso a los puntos de conexión de AWS SSM solo es necesario si utiliza activaciones híbridas de AWS SSM para el proveedor de credenciales de IAM en las instalaciones.
[2] El acceso a los puntos de conexión de AWS IAM solo es necesario si utiliza AWS IAM Roles Anywhere como proveedor de credenciales de IAM en las instalaciones.

Acceso necesario para las operaciones de clúster en curso

El siguiente acceso a la red para el firewall en las instalaciones es necesario para las operaciones de clúster en curso.

Important

Según el CNI que elija, deberá configurar reglas de acceso a la red adicionales para los puertos del CNI. Consulte la documentación de Cilium y la documentación de Calico para obtener más información.

Tipo	Protocolo	Dirección	Puerto	Origen	Destino	Uso
HTTPS	TCP	Salida	443	CIDR de nodos remotos	IP de clúster de EKS [1]	Servidor de API de kubelet a Kubernetes
HTTPS	TCP	Salida	443	CIDR de pods remotos	IP de clúster de EKS [1]	Pod a servidor de API de Kubernetes
HTTPS	TCP	Salida	443	CIDR de nodos remotos	Punto de conexión de servicio de SSM	Actualización de credenciales de activaciones híbridas de SSM y señales de SSM cada 5 minutos
HTTPS	TCP	Salida	443	CIDR de nodos remotos	Punto de conexión del servicio IAM Anywhere	Actualización de credenciales de IAM Roles Anywhere
HTTPS	TCP	Salida	443	CIDR de pods remotos	Punto de conexión regional de STS	Pod a punto de conexión de STS. Obligatorio

Tipo	Protocolo	Dirección	Puerto	Origen	Destino	Uso
						solo para IRSA
HTTPS	TCP	Salida	443	CIDR de nodos remotos	Punto de conexión del servicio Auth de Amazon EKS	Nodo a punto de conexión de autenticación de Amazon EKS. Obligatorio solo para Pod Identity de Amazon EKS
HTTPS	TCP	Entrada	10250	IP de clúster de EKS [1]	CIDR de nodos remotos	Servidor de la API de Kubernetes a kubelet
HTTPS	TCP	Entrada	Puertos de webhook	IP de clúster de EKS [1]	CIDR de pods remotos	Servidor de la API de Kubernetes a webhooks

Tipo	Protocolo	Dirección	Puerto	Origen	Destino	Uso
HTTPS	TCP,UDP	Entrada,Salida	53	CIDR de pods remotos	CIDR de pods remotos	Pod a CoreDNS. Si ejecuta al menos una réplica de CoreDNS en la nube, debe permitir el tráfico DNS a la VPC donde se ejecuta CoreDNS.
Definido por el usuario	Definido por el usuario	Entrada,Salida	Puertos de la aplicación	CIDR de pods remotos	CIDR de pods remotos	Pod a pod

> **Note**
>
> [1] Las direcciones IP del clúster de EKS. Consulte la siguiente sección sobre las interfaces de red elásticas de Amazon EKS.

Interfaces de red de Amazon EKS

Amazon EKS asocia interfaces de red a las subredes de la VPC que se transmiten durante la creación del clúster para permitir la comunicación entre el plano de control de EKS y la VPC. Las interfaces de red que Amazon EKS crea se pueden encontrar tras la creación del clúster en la consola de Amazon EC2 o con la AWS CLI. Las interfaces de red originales se eliminan y se crean nuevas interfaces de red cuando se aplican cambios en el clúster de EKS, como actualizaciones de la versión de Kubernetes. Para restringir el rango de IP de las interfaces de red de Amazon EKS, es posible utilizar tamaños de subred limitados para las subredes que se transmiten durante la creación

del clúster, lo que facilita la configuración del firewall en las instalaciones para permitir la conectividad entrante/saliente a este conjunto de IP conocidas y limitadas. Para controlar en qué subredes se crean las interfaces de red, puede limitar la cantidad de subredes que especifica al crear un clúster o puede actualizar las subredes después de crear el clúster.

Las interfaces de red aprovisionadas por Amazon EKS tienen una descripción del formato `Amazon EKS` *`your-cluster-name`* . Consulte el ejemplo que aparece a continuación para ver un comando de la AWS CLI que se puede utilizar para buscar las direcciones IP de las interfaces de red que Amazon EKS aprovisiona. Sustituya `VPC_ID` por el ID de la VPC que se transmite durante la creación del clúster.

```
aws ec2 describe-network-interfaces \
--query 'NetworkInterfaces[?(VpcId == VPC_ID && contains(Description,Amazon
 EKS))].PrivateIpAddress'
```

AWS VPC y configuración de subredes

Los requisitos de VPC y subredes existentes para Amazon EKS se aplican a los clústeres con nodos híbridos. Además, el CIDR de la VPC no se puede solapar con los CIDR de los nodos y pods en las instalaciones. Debe configurar rutas en la tabla de enrutamiento de la VPC para el CIDR de nodos en las instalaciones y, opcionalmente, el CIDR de pods. Estas rutas se deben configurar para dirigir el tráfico a la puerta de enlace que se utiliza para la conectividad de la red híbrida, que comúnmente es una puerta de enlace privada virtual (VGW) o una puerta de enlace de tránsito (TGW). Si está utiliza TGW o VGW para conectar la VPC con el entorno en las instalaciones, debe crear una conexión TGW o VGW para la VPC. La VPC debe tener un nombre de host DNS y admitir la resolución DNS.

Para los siguientes pasos, se utiliza la AWS CLI. También puede crear estos recursos en la Consola de administración de AWS o con otras interfaces, como AWS CloudFormation, AWS CDK o Terraform.

Paso 1: Creación de la VPC

1. Ejecute el siguiente comando para crear una VPC. Reemplace `IPv4` con un rango CIDR `10.0.0.0/16` RFC-1918 (privado) o no RFC-1918 (público) (por ejemplo, `VPC_CIDR`). Nota: La resolución de DNS, que es un requisito de EKS, está habilitada para la VPC de forma predeterminada.

   ```
   aws ec2 create-vpc --cidr-block VPC_CIDR
   ```

2. Habilite los nombres de host de DNS para la VPC. Nota: La resolución de DNS está habilitada para la VPC de forma predeterminada. Sustituya `VPC_ID` por el ID de la VPC que creó en el paso anterior.

```
aws ec2 modify-vpc-attribute --vpc-id VPC_ID --enable-dns-hostnames
```

Paso 2: Creación de subredes

Cree al menos 2 subredes. Amazon EKS usa estas subredes para las interfaces de red del clúster. Para obtener más información, consulte Requisitos y consideraciones de las subredes.

1. Puede buscar las zonas de disponibilidad de una región de AWS con el siguiente comando. Reemplace `us-west-2` por la región.

```
aws ec2 describe-availability-zones \
     --query 'AvailabilityZones[?(RegionName == us-west-2)].ZoneName'
```

2. Cree una subred. Sustituya `VPC_ID` por el ID de la VPC. Sustituya `SUBNET_CIDR` por el bloque de CIDR de la subred (por ejemplo, 10.0.1.0/24). Sustituya `AZ` por la zona de disponibilidad en la que se creará la subred (por ejemplo, us-west-2a). Las subredes que cree deben estar en al menos dos zonas de disponibilidad diferentes.

```
aws ec2 create-subnet \
    --vpc-id VPC_ID \
    --cidr-block SUBNET_CIDR \
    --availability-zone AZ
```

(Opcional) Paso 3: adjunción de una VPC con la puerta de enlace de tránsito (TGW) de Amazon VPC o una puerta de enlace privada virtual (VGW) de AWS Direct Connect

Si utiliza una TGW o VGW, conecte la VPC a la TGW o VGW. Para obtener más información, consulte Amazon VPC attachments in Amazon VPC Transit Gateways o AWS Direct Connect virtual private gateway associations.

Transit Gateway

Ejecute el siguiente comando para conectar una puerta de enlace de tránsito. Sustituya `VPC_ID` por el ID de la VPC. Sustituya `SUBNET_ID1` y `SUBNET_ID2` por el ID de las subredes que creó en el paso anterior. Sustituya `TGW_ID` por el ID de la TGW.

```
aws ec2 create-transit-gateway-vpc-attachment \
    --vpc-id VPC_ID \
    --subnet-ids SUBNET_ID1 SUBNET_ID2 \
    --transit-gateway-id TGW_ID
```

Puerta de enlace privada virtual

Ejecute el siguiente comando para conectar una puerta de enlace de tránsito. Sustituya `VPN_ID` por el ID de la VGW. Sustituya `VPC_ID` por el ID de la VPC.

```
aws ec2 attach-vpn-gateway \
    --vpn-gateway-id VPN_ID \
    --vpc-id VPC_ID
```

(Opcional) Paso 4: Creación de una tabla de enrutamiento

Puede modificar la tabla de enrutamiento principal de la VPC o puede crear una tabla de enrutamiento personalizada. En los siguientes pasos, se crea una tabla de enrutamiento personalizada con las rutas a los CIDR de nodos y pods en las instalaciones. Para obtener más información, consulte Tablas de enrutamiento de subredes. Sustituya `VPC_ID` por el ID de la VPC.

```
aws ec2 create-route-table --vpc-id VPC_ID
```

Paso 5: Creación de rutas para nodos y pods en las instalaciones

Cree rutas en la tabla de enrutamiento para cada uno de los nodos remotos en las instalaciones. Puede modificar la tabla de enrutamiento principal de la VPC o utilizar la tabla de enrutamiento personalizada que creó en el paso anterior.

En los ejemplos que aparecen a continuación se muestra cómo crear rutas para los CIDR de nodos y pods en las instalaciones. En los ejemplos, se utiliza una puerta de enlace de tránsito (TGW) para conectar la VPC con el entorno en las instalaciones. Si tiene múltiples CIDR de nodos y pods en las instalaciones, repita los pasos para cada CIDR.

- Si utiliza una puerta de enlace de Internet o una puerta de enlace privada virtual (VGW), sustituya `--transit-gateway-id` por `--gateway-id`.
- Sustituya `RT_ID` por el ID de la tabla de enrutamiento que creó en el paso anterior.
- Sustituya `REMOTE_NODE_CIDR` por el rango de CIDR que utilizará para los nodos híbridos.
- Sustituya `REMOTE_POD_CIDR` por el rango de CIDR que utilizará para los pods que se ejecutan en nodos híbridos. El rango de CIDR del pod corresponde a la configuración de la interfaz de red de contenedores (CNI), que habitualmente utiliza una red superpuesta en las instalaciones. Para obtener más información, consulte the section called “Cómo configurar una CNI”.
- Sustituya `TGW_ID` por el ID de la TGW.

Red de nodos remotos

```
aws ec2 create-route \
    --route-table-id RT_ID \
    --destination-cidr-block REMOTE_NODE_CIDR \
    --transit-gateway-id TGW_ID
```

Red de pods remotos

```
aws ec2 create-route \
    --route-table-id RT_ID \
    --destination-cidr-block REMOTE_POD_CIDR \
    --transit-gateway-id TGW_ID
```

(Opcional) Paso 6: Asociación de las subredes a la tabla de enrutamiento

Si creó una tabla de enrutamiento personalizada en el paso anterior, asocie cada una de las subredes que creó en el paso anterior a la tabla de enrutamiento personalizada. Si va a modificar la tabla de enrutamiento principal de la VPC, las subredes se asociarán automáticamente a la tabla de enrutamiento principal de la VPC y podrá omitir este paso.

Ejecute el siguiente comando para cada una de las subredes que creó en los pasos anteriores. Sustituya `RT_ID` por la tabla de enrutamiento que creó en el paso anterior. Sustituya `SUBNET_ID` por el ID de una subred.

```
aws ec2 associate-route-table --route-table-id RT_ID --subnet-id SUBNET_ID
```

Configuración del grupo de seguridad del clúster

El siguiente acceso para el grupo de seguridad del clúster de EKS se necesita para las operaciones de clúster en curso. Amazon EKS crea automáticamente las reglas de grupo de seguridad de entrada necesarias para los nodos híbridos al crear o actualizar el clúster con las redes de nodos y pods remotos configuradas. Como los grupos de seguridad permiten todo el tráfico saliente de forma predeterminada, Amazon EKS no modifica automáticamente las reglas de salida del grupo de seguridad del clúster para los nodos híbridos. Si desea personalizar el grupo de seguridad del clúster, puede limitar el tráfico a las reglas de la siguiente tabla.

Tipo	Protocolo	Dirección	Puerto	Origen	Destino	Uso
HTTPS	TCP	Entrada	443	CIDR de nodos remotos	N/A	Servidor de la API de Kubelet a Kubernetes
HTTPS	TCP	Entrada	443	CIDR de pods remotos	N/A	Pods que requieren acceso al servidor de la API de K8s cuando el CNI no utiliza NAT para el tráfico de pods.
HTTPS	TCP	Salida	10250	N/A	CIDR de nodos remotos	Servidor de la API de Kubernetes a Kubelet

Tipo	Protocolo	Dirección	Puerto	Origen	Destino	Uso
HTTPS	TCP	Salida	Puertos de webhook	N/A	CIDR de pods remotos	Servidor de la API de Kubernetes a webhook (si se ejecutan webhooks en nodos híbridos)

> **Important**
>
> Límites de reglas de grupos de seguridad: los grupos de seguridad de Amazon EC2 tienen un máximo de 60 reglas de entrada de forma predeterminada. Es posible que las reglas de entrada de los grupos de seguridad no se apliquen si el grupo de seguridad del clúster se acerca a este límite. En este caso, puede que sea necesario agregar manualmente las reglas de entrada que faltan.
> Responsabilidad de limpiar el CIDR: si se eliminan las redes de nodos o pods remotos de los clústeres de EKS, EKS no elimina automáticamente las reglas de los grupos de seguridad correspondientes. Usted tiene la responsabilidad de eliminar manualmente las redes de nodos o pods remotos que no se utilicen de las reglas de su grupo de seguridad.

Para obtener más información acerca del grupo de seguridad de clúster que crea Amazon EKS, consulte the section called “Requisitos del grupo de seguridad”.

(Opcional) Configuración manual del grupo de seguridad

Si necesita crear grupos de seguridad adicionales o modificar las reglas que se crean automáticamente, puede utilizar los siguientes comandos como referencia. De forma predeterminada, el siguiente comando crea un grupo de seguridad que permite todos los accesos salientes. Puede restringir el acceso saliente para incluir únicamente las reglas indicadas anteriormente. Si considera limitar las reglas de salida, recomendamos que pruebe exhaustivamente la conectividad de todas las aplicaciones y pods antes de aplicar las reglas modificadas a un clúster en fase de producción.

- En el primer comando, sustituya `SG_NAME` por el nombre del grupo de seguridad
- En el primer comando, sustituya `VPC_ID` por el ID de la VPC que creó en el paso anterior
- En el segundo comando, sustituya `SG_ID` por el ID del grupo de seguridad que creó en el primer comando
- En el segundo comando, sustituya `REMOTE_NODE_CIDR` y `REMOTE_POD_CIDR` por los valores de los nodos híbridos y la red en las instalaciones.

```
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"}]}]'
```

Cómo preparar el sistema operativo para los nodos híbridos

Bottlerocket, Amazon Linux 2023 (AL2023), Ubuntu y RHEL se validan de forma continua para su uso como sistema operativo de nodo para los nodos híbridos. Bottlerocket es compatible únicamente con AWS en los entornos VMware vSphere. Los planes de AWS Support no cubren AL2023 cuando se ejecuta fuera de Amazon EC2. AL2023 solo se puede utilizar en entornos virtualizados en las instalaciones. Consulte la Guía del usuario de Amazon Linux 2023 para obtener más información. AWS admite la integración de nodos híbridos con los sistemas operativos Ubuntu y RHEL, pero no proporciona soporte para los sistemas operativos en sí.

Usted es responsable del aprovisionamiento y la administración del sistema operativo. Al probar nodos híbridos por primera vez, lo más sencillo es ejecutar la CLI de Nodos híbridos de Amazon EKS (`nodeadm`) en un host ya aprovisionado. En el caso de implementaciones en producción, se recomienda incluir `nodeadm` en las imágenes del sistema operativo, configurado para ejecutarse como un servicio de systemd, de modo que los hosts se unan automáticamente a los clústeres de Amazon EKS al iniciar. Si usa Bottlerocket como sistema operativo de nodo en vSphere, no es necesario utilizar `nodeadm`, ya que Bottlerocket incluye las dependencias necesarias para nodos híbridos y se conectará automáticamente al clúster configurado al iniciar el host.

Compatibilidad de versiones

La tabla que aparece a continuación representa las versiones del sistema operativo compatibles y validadas para su uso como sistema operativo de nodo para nodos híbridos. Si utiliza otras variantes o versiones del sistema operativo que no aparecen en esta tabla, la compatibilidad de los nodos híbridos con la variante o versión del sistema operativo no está cubierta por AWS Support. Los nodos híbridos son independientes de la infraestructura subyacente y admiten arquitecturas x86 y ARM.

Sistema operativo	Versiones
Amazon Linux	Amazon Linux 2023 (AL2023)
Bottlerocket	Variantes de VMware desde la v1.37.0 en adelante que ejecutan Kubernetes v1.28 o superior
Ubuntu	Ubuntu 20.04, Ubuntu 22.04, Ubuntu 24.04
Red Hat Enterprise Linux	RHEL 8, RHEL 9

Consideraciones de los sistemas operativos

General

- La CLI (`nodeadm`) de los Nodos híbridos de Amazon EKS se puede utilizar para simplificar la instalación y configuración de los componentes y dependencias de los nodos híbridos. Puede ejecutar el proceso `nodeadm install` durante las canalizaciones de creación de imágenes del sistema operativo o en tiempo de ejecución en cada host en las instalaciones. Para más información sobre los componentes que `nodeadm` instala, consulte el the section called “Nodeadm de nodos híbridos”.
- Si utiliza un proxy en el entorno en las instalaciones para acceder a Internet, se requiere una configuración adicional del sistema operativo para que los procesos de instalación y actualización configuren el administrador de paquetes para utilizar el proxy. Para obtener instrucciones, consulte the section called “Configuración del proxy”.

Bottlerocket

- Los pasos y herramientas para conectar un nodo de Bottlerocket son diferentes de los requeridos para otros sistemas operativos y se abordan por separado en the section called “Conexión de nodos híbridos con Bottlerocket”, en lugar de los pasos indicados en the section called “Cómo conectar nodos híbridos”.
- Los pasos para Bottlerocket no requieren el uso de la herramienta de interfaz de la línea de comandos (CLI) para nodos híbridos, `nodeadm`.
- Solo se admiten las variantes de VMware de Bottlerocket a partir de la versión v1.37.0 con los Nodos híbridos de EKS. Las variantes de VMware de Bottlerocket están disponibles para las versiones v1.28 y posteriores de Kubernetes. Otras variantes de Bottlerocket no son compatibles como sistema operativo de nodos híbridos. NOTA: Las variantes de VMware de Bottlerocket solo se encuentran disponibles para la arquitectura x86_64.

Containerd

- Containerd es el tiempo de ejecución de contenedores estándar de Kubernetes y es una dependencia para los nodos híbridos, así como para todos los tipos de computación de nodos de Amazon EKS. La CLI (`nodeadm`) de los Nodos Híbridos de Amazon EKS intenta instalar containerd durante el proceso `nodeadm install`. Puede configurar la instalación de containerd en tiempo de ejecución de `nodeadm install` con la opción de línea de comandos `--containerd-source`. Las opciones válidas son `none`, `distro` y `docker`. Si utiliza RHEL, `distro` no es una opción válida y puede, o bien configurar `nodeadm` para instalar la compilación containerd desde los repositorios de Docker, o instalar containerd manualmente. Cuando se usa AL2023 o Ubuntu, `nodeadm` instala containerd desde la distribución del sistema operativo de forma predeterminada. Si no quiere que nodeadm instale containerd, use la opción `--containerd-source none`.

Ubuntu

- Si usa Ubuntu 24.04, es posible que tenga que actualizar la versión de containerd o cambiar la configuración de AppArmor para adoptar una corrección que permita que los pods se terminen correctamente. Consulte Ubuntu #2065423. Se requiere un reinicio para aplicar los cambios al perfil de AppArmor. La versión más reciente de Ubuntu 24.04 tiene una versión actualizada de containerd en su administrador de paquetes con la corrección (versión 1.7.19+ de containerd).

ARM

- Si utiliza hardware de ARM, se requiere un procesador compatible con ARMv8.2 con la extensión de criptografía (ARMv8.2+crypto) para ejecutar la versión 1.31 o posterior del complemento kube-proxy de EKS. Todos los sistemas Raspberry Pi anteriores a Raspberry Pi 5, así como los procesadores basados en Cortex-A72, no cumplen este requisito. Como solución alternativa, puede seguir utilizando la versión 1.30 del complemento kube-proxy de EKS hasta que finalice el soporte extendido en julio de 2026 (consulte el calendario de lanzamiento de Kubernetes o utilice una imagen de kube-proxy personalizada ascendente).
- El siguiente mensaje de error del registro de kube-proxy indica esta incompatibilidad:

```
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.
```

Creación de imágenes del sistema operativo

Amazon EKS proporciona plantillas Packer de ejemplo que se pueden utilizar para crear imágenes del sistema operativo que incluyan `nodeadm` y lo configuren de modo que se ejecute al iniciar el host. Este proceso se recomienda para no tener que extraer las dependencias de los nodos híbridos individualmente en cada host y para automatizar el proceso de arranque de los nodos híbridos. Puede utilizar las plantillas Packer de ejemplo con una imagen ISO de Ubuntu 22.04, Ubuntu 24.04, RHEL 8 o RHEL 9 y puede obtener imágenes con estos formatos: OVA, Qcow2 o sin procesar.

Requisitos previos

Antes de utilizar las plantillas Packer de ejemplo, debe tener instalado lo siguiente en la máquina desde la que ejecuta Packer.

- Versión 1.11.0 o superior de Packer. Para obtener instrucciones sobre cómo instalar Packer, consulte Instalación de Packer en la documentación de Packer.
- Si se crean OVA, la versión 1.4.0 o superior del complemento VMware vSphere.
- Si crea imágenes `Qcow2` o sin procesar, utilice la versión 1.x del complemento QEMU

Configuración de las variables de entorno

Antes de ejecutar la compilación de Packer, establezca las siguientes variables de entorno en la máquina desde la que ejecuta Packer.

General

Se deben establecer las siguientes variables de entorno para crear imágenes con todos los sistemas operativos y formatos de salida.

Variable de entorno	Tipo	Descripción
PKR_SSH_PASSWORD	Cadena	Durante el aprovisionamiento, Packer utiliza las variables `ssh_username` y `ssh_password` para acceder mediante SSH a la máquina creada. Esto debe coincidir con las contraseñas utilizadas para crear el usuario inicial dentro de los archivos kickstart o user-data del SO respectivo. El valor predeterminado se establece como “builder” o “ubuntu” según el sistema operativo. Al configurar la contraseña, asegúrese de cambiarla dentro del archivo `ks.cfg` o `user-data` correspondiente de modo que coincida.
ISO_URL	Cadena	URL de la ISO que se va a utilizar. Puede ser un enlace web para descargar de un servidor, o una ruta absoluta a un archivo local
ISO_CHECKSUM	Cadena	Suma de comprobación asociada para la ISO suministrada.

Variable de entorno	Tipo	Descripción
CREDENTIAL_PROVIDER	Cadena	Proveedor de credenciales para nodos híbridos. Los valores válidos son `ssm` (predeterminados) para las activaciones híbridas de SSM y `iam` para IAM Roles Anywhere
K8S_VERSION	Cadena	Versión de Kubernetes para nodos híbridos (por ejemplo, `1.31`). Para ver las versiones de Kubernetes compatibl es, consulte las versiones compatibles de Amazon EKS.
NODEADM_ARCH	Cadena	Arquitecturas para `nodeadm install`. Seleccione `amd` o `arm`.

RHEL

Si utiliza RHEL, se deben configurar las siguientes variables de entorno.

Variable de entorno	Tipo	Descripción
RH_USERNAME	Cadena	Nombre de usuario del administrador de suscripci ones de RHEL
RH_PASSWORD	Cadena	Contraseña del administrador de suscripciones de RHEL
RHEL_VERSION	Cadena	La versión iso de Rhel que se utiliza. Los valores válidos son 8 o 9.

Ubuntu

No se requieren variables de entorno específicas de Ubuntu.

vSphere

Si va a crear un OVA de VMware vSphere, deberá configurar las siguientes variables de entorno.

Variable de entorno	Tipo	Descripción
VSPHERE_SERVER	Cadena	Dirección del servidor vSphere
VSPHERE_USER	Cadena	Nombre de usuario de vSphere
VSPHERE_PASSWORD	Cadena	Contraseña de vSphere
VSPHERE_DATACENTER	Cadena	Nombre del centro de datos de vSphere
VSPHERE_CLUSTER	Cadena	Nombre del clúster de vSphere
VSPHERE_DATASTORE	Cadena	Nombre del almacén de datos de vSphere
VSPHERE_NETWORK	Cadena	Nombre de la red de vSphere
VSPHERE_OUTPUT_FOLDER	Cadena	Carpeta de salida de vSphere para las plantillas

QEMU

Variable de entorno	Tipo	Descripción
PACKER_OUTPUT_FORMAT	Cadena	Formato de salida para el generador QEMU. Los valores válidos son `qcow2` y `raw`.

Validación de la plantilla

Antes de ejecutar la compilación, valide la plantilla con el siguiente comando después de configurar las variables de entorno. Sustituya `template.pkr.hcl` si utiliza un nombre diferente para la plantilla.

```
packer validate template.pkr.hcl
```

Generación de imágenes

Genere las imágenes con los siguientes comandos y utilice la marca `-only` para especificar el destino y el sistema operativo de las imágenes. Sustituya `template.pkr.hcl` si utiliza un nombre diferente para la plantilla.

OVA de vSphere

> **Note**
>
> Si utiliza RHEL con vSphere necesita convertir los archivos kickstart a una imagen OEMDRV y transmitirla como una ISO desde la que arrancar. Para obtener más información, consulte Readme de Packer en el repositorio de GitHub de los Nodos híbridos de EKS.

OVA de Ubuntu

```
packer build -only=general-build.vsphere-iso.ubuntu22 template.pkr.hcl
```

OVA de Ubuntu

```
packer build -only=general-build.vsphere-iso.ubuntu24 template.pkr.hcl
```

OVA de RHEL

```
packer build -only=general-build.vsphere-iso.rhel8 template.pkr.hcl
```

OVA de RHEL

```
packer build -only=general-build.vsphere-iso.rhel9 template.pkr.hcl
```

QEMU

> **Note**
>
> Si va a generar una imagen para una CPU de host específica que no coincide con el host del generador, consulte la documentación de QEMU para obtener el nombre que coincida con la CPU de host y utilice la marca `-cpu` con el nombre de la CPU de host cuando ejecute los siguientes comandos.

Ubuntu 22.04 Qcow2/sin procesar

```
packer build -only=general-build.qemu.ubuntu22 template.pkr.hcl
```

Ubuntu 24.04 Qcow2/sin procesar

```
packer build -only=general-build.qemu.ubuntu24 template.pkr.hcl
```

RHEL 8 Qcow2/sin procesar

```
packer build -only=general-build.qemu.rhel8 template.pkr.hcl
```

RHEL 9 Qcow2/sin procesar

```
packer build -only=general-build.qemu.rhel9 template.pkr.hcl
```

Cómo transmitir la configuración de nodeadm a través de user-data

Puede transmitir la configuración de `nodeadm` en los user-data a través de cloud-init para configurar y conectar automáticamente los nodos híbridos al clúster de EKS al iniciar el host. A continuación, encontrará un ejemplo de cómo lograrlo al utilizar VMware vSphere como infraestructura para los nodos híbridos.

1. Instale la CLI de `govc` según las instrucciones del archivo govc readme en GitHub.
2. Después de ejecutar la compilación de Packer en la sección anterior y aprovisionar la plantilla, podrá clonar la plantilla para crear varios nodos diferentes de la siguiente manera. Debe clonar la plantilla para cada nueva máquina virtual que cree y que vaya a utilizar para nodos híbridos. Sustituya las variables del comando que aparece a continuación por los valores correspondientes al entorno. El `VM_NAME` en el comando que aparece a continuación se utiliza

como NODE_NAME cuando se introducen los nombres de las máquinas virtuales a través del archivo `metadata.yaml`.

```
govc vm.clone -vm "/PATH/TO/TEMPLATE" -ds="YOUR_DATASTORE" \
    -on=false -template=false -folder=/FOLDER/TO/SAVE/VM "VM_NAME"
```

3. Después de clonar la plantilla para cada una de sus nuevas máquinas virtuales, cree un `userdata.yaml` y `metadata.yaml` para las máquinas virtuales. Las máquinas virtuales pueden compartir los mismos `userdata.yaml` y `metadata.yaml`, que deberá completar según cada máquina virtual y a través de los pasos que se indican a continuación. La configuración `nodeadm` se crea y define en la sección `write_files` del `userdata.yaml`. En el siguiente ejemplo, se utilizan las activaciones híbridas de AWS SSM como proveedor de credenciales en las instalaciones para los nodos híbridos. Para obtener más información sobre la configuración de `nodeadm`, consulte the section called “Nodeadm de nodos híbridos”.

   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:

Cree un `metadata.yaml` para el entorno. Mantenga el formato de la variable `"$NODE_NAME"` en el archivo, ya que este se completará con valores en un paso posterior.

```
instance-id: "$NODE_NAME"
local-hostname: "$NODE_NAME"
network:
  version: 2
  ethernets:
    nics:
      match:
        name: ens*
      dhcp4: yes
```

4. Agregue los archivos `userdata.yaml` y `metadata.yaml` como cadenas `gzip+base64` con los siguientes comandos. Se deben ejecutar los siguientes comandos para cada una de las máquinas virtuales que se van a crear. Sustituya `VM_NAME` por el nombre de la máquina virtual que va a actualizar.

```
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. Encienda las nuevas máquinas virtuales, que se conectarán automáticamente al clúster de EKS que configuró.

```
govc vm.power -on "${NODE_NAME}"
```

Cómo preparar las credenciales para los nodos híbridos

Los Nodos híbridos de Amazon EKS utilizan credenciales de IAM temporales aprovisionadas por activaciones híbridas de AWS SSM o AWS IAM Roles Anywhere para autenticarse con el clúster de Amazon EKS. Debe usar activaciones híbridas de AWS SSM o AWS IAM Roles Anywhere con la CLI (`nodeadm`) de los Nodos híbridos de Amazon EKS. No debe utilizar ambas opciones, únicamente activaciones híbridas de AWS o AWS IAM Roles Anywhere. Le recomendamos utilizar las activaciones híbridas de AWS SSM si no dispone de una infraestructura de clave pública (PKI) existente con una entidad de certificación (CA) y certificados para los entornos en las instalaciones. Si ya dispone de PKI y certificados en las instalaciones, utilice AWS IAM Roles Anywhere.

Rol de IAM de nodos híbridos

Antes de poder conectar nodos híbridos al clúster de Amazon EKS, deberá crear un rol de IAM que se utilizará con las activaciones híbridas de AWS SSM o AWS IAM Roles Anywhere para las credenciales de los nodos híbridos. Tras la creación del clúster, utilizará este rol con una entrada de acceso de Amazon EKS o una entrada de `aws-auth` ConfigMap para asignar el rol de IAM al control de acceso basado en roles (RBAC) de Kubernetes. Para obtener más información sobre cómo asociar el de IAM de Nodos híbridos al RBAC de Kubernetes, consulte the section called “Cómo preparar el acceso al clúster”.

El rol de IAM de nodos híbridos debe tener los siguientes permisos:

- Permisos para que `nodeadm` utilice la acción `eks:DescribeCluster` a fin de recopilar información sobre el clúster al que quiere conectar los nodos híbridos. Si no activa la acción `eks:DescribeCluster`, deberá transmitir el punto de conexión de la API de Kubernetes, el paquete de CA del clúster y el CIDR IPv4 del servicio a la configuración de nodos que transmite al comando `nodeadm init`.
- Permisos para que `nodeadm` utilice la acción `eks:ListAccessEntries` a fin de enumerar las entradas de acceso en el clúster al que quiere conectar los nodos híbridos. Si no activa la acción `eks:ListAccessEntries`, deberá transmitir el indicador `--skip cluster-access-validation` al ejecutar el comando `nodeadm init`.

- Permisos para que el kubelet pueda utilizar imágenes de contenedores de Amazon Elastic Container Registry (Amazon ECR), según lo dispuesto en la política AmazonEC2ContainerRegistryPullOnly.
- Si usa AWS SSM, permisos para que `nodeadm init` utilice las activaciones híbridas de AWS SSM, tal y como se define en la política aws-managed-policy/latest/reference/AmazonSSMManagedInstanceCore.html.
- Si usa AWS SSM, permisos para usar la acción `ssm:DeregisterManagedInstance` y la acción `ssm:DescribeInstanceInformation` para que `nodeadm uninstall` anule el registro de instancias.
- (Opcional) Permisos para que el agente de Pod Identity de Amazon EKS utilice la acción `eks-auth:AssumeRoleForPodIdentity` para recuperar las credenciales de los pods.

Configuración de activaciones híbridas de AWS SSM

Antes de configurar las activaciones híbridas de AWS SSM, deberá crear y configurar un rol de IAM de nodos híbridos. Para obtener más información, consulte the section called “Creación del rol de IAM de nodos híbridos”. Siga las instrucciones que aparecen en Cómo crear una activación híbrida para registrar nodos en Systems Manager en la Guía del usuario de AWS Systems Manager para crear una activación híbrida de AWS SSM para los nodos híbridos. El código de activación y el ID que recibe se utilizan con `nodeadm` al registrar los hosts como nodos híbridos en el clúster de Amazon EKS. Podrá volver a este paso más adelante, una vez que haya creado y preparado los clústeres de Amazon EKS para nodos híbridos.

> **Important**
>
> Systems Manager regresa inmediatamente el código e ID de activación a la consola o la ventana de comandos, en función de cómo haya creado la activación. Copie esta información y guárdela en un lugar seguro. Si sale de la consola o cierra la ventana de comandos, podría perder esta información. Si la pierde, debe crear una nueva activación.

De forma predeterminada, las activaciones híbridas de AWS SSM se mantienen activas durante 24 horas. También puede especificar una `--expiration-date` al crear la activación híbrida en formato de marca de tiempo, por ejemplo `2024-08-01T00:00:00`. Cuando utiliza AWS SSM como proveedor de credenciales, el nombre de nodo de los nodos híbridos no se puede configurar, sino que AWS SSM lo genera automáticamente. Puede ver y administrar las instancias administradas

por AWS SSM en la consola de AWS Systems Manager, en Administrador de flotas. Puede registrar hasta 1000 nodos activados de manera híbrida estándar por cada cuenta por región de AWS sin costo adicional. Sin embargo, para registrar más de 1000 nodos híbridos es necesario activar el nivel de instancias avanzadas. El uso del nivel de instancias avanzadas conlleva un cargo que no está incluido en los precios de los Nodos híbridos de Amazon EKS. Para obtener más información, consulte Precios de AWS Systems Manager.

Consulte el siguiente ejemplo para saber cómo crear una activación híbrida de AWS SSM con el rol de IAM de nodos híbridos. Si utiliza activaciones híbridas de AWS SSM para las credenciales de los nodos híbridos, los nombres de los nodos híbridos tendrán el mismo formato `mi-012345678abcdefgh` y las credenciales temporales aprovisionadas por AWS SSM serán válidas durante 1 hora. No puede modificar el nombre del nodo ni la duración de las credenciales si utiliza AWS SSM como proveedor de credenciales. AWS SSM rota automáticamente las credenciales temporales y la rotación no afecta al estado de los nodos o las aplicaciones.

Le recomendamos utilizar una activación híbrida de AWS SSM por clúster de EKS para limitar el permiso `ssm:DeregisterManagedInstance` de AWS SSM del rol de IAM de nodos híbridos de modo que únicamente pueda anular el registro de las instancias asociadas a la activación híbrida de AWS SSM. En el ejemplo que aparece en esta página, se utiliza una etiqueta con el ARN del clúster de EKS, que se puede utilizar para asignar la activación híbrida de AWS SSM al clúster de EKS. También puede utilizar la etiqueta y el método que prefiera para determinar el alcance de los permisos de AWS SSM en función de los límites de permisos y requisitos. La opción `REGISTRATION_LIMIT` del siguiente comando es un número entero que se utiliza para limitar la cantidad de máquinas que pueden utilizar la activación híbrida de AWS SSM (por ejemplo, `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
```

Consulte las instrucciones de Creación de una activación híbrida para registrar nodos en Systems Manager para obtener más información sobre los ajustes de configuración disponibles para las activaciones híbridas de AWS SSM.

Configuración de AWS IAM Roles Anywhere

Siga las instrucciones que aparecen en Introducción a IAM Roles Anywhere en la Guía del usuario de IAM Roles Anywhere para configurar el anclaje de veracidad y el perfil que utilizará como credenciales de IAM temporales para el rol de IAM de nodos híbridos. Puede crear el perfil sin agregar ningún rol. Puede crear este perfil, volver a estos pasos para crear el rol de IAM de nodos híbridos y, a continuación, agregar el rol al perfil una vez creado. También puede utilizar los pasos de AWS CloudFormation que aparecen más adelante en esta página para completar la configuración de IAM Roles Anywhere para los nodos híbridos.

Al agregar el rol de IAM de nodos híbridos al perfil, seleccione Aceptar el nombre de sesión de rol personalizado en el panel de nombre de sesión de Rol personalizado, situado en la parte inferior de la página Editar perfil de la consola de AWS IAM Roles Anywhere. Corresponde al campo acceptRoleSessionName de la API `CreateProfile`. Esto permite proporcionar un nombre de nodo personalizado para los nodos híbridos en la configuración que se transmite a `nodeadm` durante el proceso de arranque. Es necesario transmitir un nombre de nodo personalizado durante el proceso de `nodeadm init`. Puede actualizar el perfil para aceptar un nombre de sesión de rol personalizado después de crear el perfil.

Puede configurar la duración de la validez de las credenciales con AWS IAM Roles Anywhere mediante el campo durationSeconds del perfil de AWS IAM Roles Anywhere. La duración predeterminada es de una hora con un máximo de 12 horas. La configuración `MaxSessionDuration` del rol de IAM de nodos híbridos debe ser mayor que la configuración `durationSeconds` del perfil de AWS IAM Roles Anywhere. Para obtener más información sobre `MaxSessionDuration`, consulte la documentación sobre la API UpdateRole.

Los certificados y claves por máquina que genere a partir de la entidad de certificación (CA) se deben colocar en el directorio `/etc/iam/pki` de cada nodo híbrido con los nombres de archivo `server.pem` para el certificado y `server.key` para la clave.

Creación del rol de IAM de nodos híbridos

Para ejecutar los pasos de esta sección, la entidad principal de IAM que utilice la consola de AWS o AWS CLI debe tener los siguientes permisos.

- `iam:CreatePolicy`
- `iam:CreateRole`
- `iam:AttachRolePolicy`
- Si utiliza AWS IAM Roles Anywhere

- `rolesanywhere:CreateTrustAnchor`
- `rolesanywhere:CreateProfile`
- `iam:PassRole`

AWS CloudFormation

Si aún no lo ha hecho, instale y configure AWS CLI. Consulte Instalación o actualización de la versión más reciente de AWS CLI.

Pasos para las activaciones híbridas de AWS SSM

La pila de CloudFormation crea el rol de IAM de nodos híbridos con los permisos descritos anteriormente. La plantilla de CloudFormation no crea la activación híbrida de AWS SSM.

1. Descargue la plantilla de AWS SSM CloudFormation para nodos híbridos:

```
curl -OL 'https://raw.githubusercontent.com/aws/eks-hybrid/refs/heads/main/example/
hybrid-ssm-cfn.yaml'
```

2. Cree un `cfn-ssm-parameters.json` con las opciones siguientes:
   a. Sustituya `ROLE_NAME` por el nombre del rol de IAM de nodos híbridos. De forma predeterminada, la plantilla de CloudFormation utiliza `AmazonEKSHybridNodesRole` como el nombre del rol que crea si no se especifica ningún nombre.
   b. Sustituya `TAG_KEY` por la clave de etiqueta del recurso de AWS SSM que utilizó al crear la activación híbrida de AWS SSM. La combinación de la clave de etiqueta y el valor de la etiqueta se utiliza en la condición de `ssm:DeregisterManagedInstance` para permitir que el rol de IAM de nodos híbridos anule el registro de las instancias administradas por AWS SSM asociadas a la activación híbrida de AWS SSM. En la plantilla de CloudFormation, el valor predeterminado de `TAG_KEY` es `EKSClusterARN`.
   c. Sustituya `TAG_VALUE` por el valor de etiqueta del recurso de AWS SSM que utilizó al crear la activación híbrida de AWS SSM. La combinación de la clave de etiqueta y el valor de la etiqueta se utiliza en la condición de `ssm:DeregisterManagedInstance` para permitir que el rol de IAM de nodos híbridos anule el registro de las instancias administradas por AWS SSM asociadas a la activación híbrida de AWS SSM. Si utiliza la `TAG_KEY` predeterminada de `EKSClusterARN`, transmita el ARN del clúster de EKS como el `TAG_VALUE`. Los ARN del clúster de EKS tienen el formato `arn:aws:eks:AWS_REGION:AWS_ACCOUNT_ID:cluster/CLUSTER_NAME`.

```
{
  "Parameters": {
    "RoleName": "ROLE_NAME",
    "SSMDeregisterConditionTagKey": "TAG_KEY",
    "SSMDeregisterConditionTagValue": "TAG_VALUE"
  }
}
```

3. Implementación de la pila de CloudFormation. Sustituya `STACK_NAME` por el nombre de la pila de CloudFormation.

```
aws cloudformation deploy \
    --stack-name STACK_NAME \
    --template-file hybrid-ssm-cfn.yaml \
    --parameter-overrides file://cfn-ssm-parameters.json \
    --capabilities CAPABILITY_NAMED_IAM
```

Pasos correspondientes a AWS IAM Roles Anywhere

La pila de CloudFormation crea el anclaje de veracidad de AWS IAM Roles Anywhere con la entidad de certificación (CA) que configure, crea el perfil de AWS IAM Roles Anywhere y crea el rol de IAM de nodos híbridos con los permisos descritos anteriormente.

1. Para configurar una entidad de certificación
   a. Para utilizar un recurso de AWS Private CA, abra la consola de AWS Private Certificate Authority. Siga las instrucciones que aparecen en la Guía del usuario de AWS Private CA.
   b. Para usar una entidad de certificación externa, siga las instrucciones proporcionadas por esta. El contenido del certificado se proporcionará más adelante.
   c. Los certificados emitidos por entidades de certificación públicas no se pueden utilizar como anclajes de veracidad.
2. Descargue la plantilla de CloudFormation de AWS IAM Roles Anywhere para nodos híbridos

```
curl -OL 'https://raw.githubusercontent.com/aws/eks-hybrid/refs/heads/main/example/
hybrid-ira-cfn.yaml'
```

3. Cree un `cfn-iamra-parameters.json` con las opciones siguientes:

a. Sustituya `ROLE_NAME` por el nombre del rol de IAM de nodos híbridos. De forma predeterminada, la plantilla de CloudFormation utiliza `AmazonEKSHybridNodesRole` como el nombre del rol que crea si no se especifica ningún nombre.

b. Sustituya `CERT_ATTRIBUTE` por el atributo de certificado por máquina que identifique de forma exclusiva al host. El atributo de certificado que utilice debe coincidir con el nodeName que utilice para la configuración de `nodeadm` al conectar nodos híbridos al clúster. Para obtener más información, consulte la the section called “Nodeadm de nodos híbridos”. De forma predeterminada, la plantilla de CloudFormation utiliza `${aws:PrincipalTag/x509Subject/CN}` como `CERT_ATTRIBUTE`, que corresponde al campo CN de los certificados por máquina. También puede transmitir `$(aws:PrincipalTag/x509SAN/Name/CN}` como `CERT_ATTRIBUTE`.

c. Sustituya `CA_CERT_BODY` por el contenido del certificado de la entidad de certificación sin saltos de línea. El `CA_CERT_BODY` debe estar en formato Privacy Enhanced Mail (PEM). Si tiene un certificado de entidad de certificación en formato PEM, elimine los saltos de línea y las líneas BEGIN CERTIFICATE y END CERTIFICATE antes de incluir el contenido del certificado de la entidad de certificación en el archivo `cfn-iamra-parameters.json`.

```
{
  "Parameters": {
    "RoleName": "ROLE_NAME",
    "CertAttributeTrustPolicy": "CERT_ATTRIBUTE",
    "CABundleCert": "CA_CERT_BODY"
  }
}
```

4. Implemente la plantilla de CloudFormation. Sustituya `STACK_NAME` por el nombre de la pila de CloudFormation.

```
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

Si aún no lo ha hecho, instale y configure AWS CLI. Consulte Instalación o actualización de la versión más reciente de AWS CLI.

Cómo crear la política de descripción de clústeres de EKS

1. Cree un archivo denominado `eks-describe-cluster-policy.json` con el siguiente contenido:

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks:DescribeCluster"
            ],
            "Resource": "*"
        }
    ]
}
```

2. Cree la política con el siguiente comando:

```
aws iam create-policy \
    --policy-name EKSDescribeClusterPolicy \
    --policy-document file://eks-describe-cluster-policy.json
```

Pasos para las activaciones híbridas de AWS SSM

1. Cree un archivo denominado `eks-hybrid-ssm-policy.json` con el siguiente contenido. La política concede permisos para dos acciones: `ssm:DescribeInstanceInformation` y `ssm:DeregisterManagedInstance`. La política restringe el permiso `ssm:DeregisterManagedInstance` a las instancias administradas por AWS SSM asociadas a la activación híbrida de AWS SSM en función de la etiqueta de recurso que especifique en la política de confianza.
   a. Sustituya `AWS_REGION` por la región AWS para la activación híbrida de AWS SSM.
   b. Reemplace `AWS_ACCOUNT_ID` por su ID de cuenta de AWS.
   c. Sustituya `TAG_KEY` por la clave de etiqueta del recurso de AWS SSM que utilizó al crear la activación híbrida de AWS SSM. La combinación de la clave de etiqueta y el valor de la etiqueta se utiliza en la condición de `ssm:DeregisterManagedInstance` para permitir que el rol de IAM de nodos híbridos anule el registro de las instancias administradas por AWS SSM

asociadas a la activación híbrida de AWS SSM. En la plantilla de CloudFormation, el valor predeterminado de `TAG_KEY` es `EKSClusterARN`.

d. Sustituya `TAG_VALUE` por el valor de etiqueta del recurso de AWS SSM que utilizó al crear la activación híbrida de AWS SSM. La combinación de la clave de etiqueta y el valor de la etiqueta se utiliza en la condición de `ssm:DeregisterManagedInstance` para permitir que el rol de IAM de nodos híbridos anule el registro de las instancias administradas por AWS SSM asociadas a la activación híbrida de AWS SSM. Si utiliza la `TAG_KEY` predeterminada de `EKSClusterARN`, transmita el ARN del clúster de EKS como el `TAG_VALUE`. Los ARN del clúster de EKS tienen el formato `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. Cree la política mediante el siguiente comando

```
aws iam create-policy \
    --policy-name EKSHybridSSMPolicy \
    --policy-document file://eks-hybrid-ssm-policy.json
```

3. Cree un archivo denominado `eks-hybrid-ssm-trust.json`. Sustituya `AWS_REGION` por la región de AWS de la activación híbrida de AWS SSM y `AWS_ACCOUNT_ID` por el ID de la cuenta de AWS.

```
{
   "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. Cree el rol con el siguiente comando.

```
aws iam create-role \
    --role-name AmazonEKSHybridNodesRole \
    --assume-role-policy-document file://eks-hybrid-ssm-trust.json
```

5. Asocie la `EKSDescribeClusterPolicy` y la `EKSHybridSSMPolicy` que creó en los pasos anteriores. Reemplace `AWS_ACCOUNT_ID` por su ID de cuenta de AWS.

```
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. Asocie las políticas `AmazonEC2ContainerRegistryPullOnly` y `AmazonSSMManagedInstanceCore` administradas por AWS.

```
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
```

Pasos correspondientes a AWS IAM Roles Anywhere

Para utilizar AWS IAM Roles Anywhere, debe configurar el anclaje de veracidad de AWS IAM Roles Anywhere antes de crear el rol de IAM de nodos híbridos. Para obtener instrucciones, consulte the section called “Configuración de AWS IAM Roles Anywhere”.

1. Cree un archivo denominado `eks-hybrid-iamra-trust.json`. Sustituya `TRUST_ANCHOR` ARN por el ARN del anclaje de veracidad que creó en los pasos de the section called “Configuración de AWS IAM Roles Anywhere”. La condición en esta política de confianza limita la capacidad de AWS IAM Roles Anywhere para asumir el rol de IAM de nodos híbridos. Esto permite el intercambio de credenciales temporales de IAM únicamente cuando el nombre de la sesión del rol coincide con el CN especificado en el certificado x509 instalado en los nodos híbridos. Alternativamente, puede utilizar otros atributos del certificado para identificar de manera única el nodo. El atributo de certificado que utilice en la política de confianza debe corresponder al `nodeName` que haya establecido en la configuración de `nodeadm`. Para obtener más información, consulte la the section called “Nodeadm de nodos híbridos”.

```
{
    "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. Cree el rol con el siguiente comando.

```
aws iam create-role \
    --role-name AmazonEKSHybridNodesRole \
    --assume-role-policy-document file://eks-hybrid-iamra-trust.json
```

3. Asocie la `EKSDescribeClusterPolicy` que creó en los pasos anteriores. Reemplace `AWS_ACCOUNT_ID` por su ID de cuenta de AWS.

```
aws iam attach-role-policy \
    --role-name AmazonEKSHybridNodesRole \
    --policy-arn arn:aws:iam::AWS_ACCOUNT_ID:policy/EKSDescribeClusterPolicy
```

4. Asocie la política `AmazonEC2ContainerRegistryPullOnly` administrada por AWS.

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

Consola de administración de AWS

Cómo crear la política de descripción de clústeres de EKS

1. Abra la consola de Amazon IAM.
2. En el panel de navegación izquierdo, elija Políticas.
3. En la página Políticas, seleccione Crear una política.
4. En la página Especificar permisos, en Seleccionar un panel de servicio, elija EKS.
   a. Filtre las acciones de DescribeCluster y seleccione la acción de lectura DescribeCluster.
   b. Elija Siguiente.
5. En la página Revisar y crear
   a. Ingresa un Nombre de política para la política, como `EKSDescribeClusterPolicy`.
   b. Seleccione Crear política.

Pasos para las activaciones híbridas de AWS SSM

1. Abra la consola de Amazon IAM.
2. En el panel de navegación izquierdo, elija Políticas.
3. En la página Políticas, seleccione Crear una política.
4. En la página Especificar permisos, en el Editor de políticas en la esquina superior derecha, elija JSON. Pegue el siguiente fragmento. Sustituya `AWS_REGION` por la región de AWS de la activación híbrida de AWS SSM y `AWS_ACCOUNT_ID` por el ID de la cuenta de AWS. Sustituya `TAG_KEY` y `TAG_VALUE` por la clave de etiqueta del recurso de AWS SSM que utilizó al crear la activación híbrida de AWS SSM.

```
{
    "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. Elija Siguiente.

5. En la página Revisar y crear.
   a. Ingrese un nombre de Política para la política, como `EKSHybridSSMPolicy`
   b. Elija Crear política.
6. En el panel de navegación izquierdo, elija Roles.
7. En la página Roles, elija Crear rol.
8. En la página Seleccionar entidad de confianza, haga lo siguiente:
   a. En la sección de tipo de Entidad de confianza, elija Política de confianza personalizada. Pegue lo siguiente en el editor de políticas de confianza personalizadas. Sustituya `AWS_REGION` por la región de AWS de la activación híbrida de AWS SSM y `AWS_ACCOUNT_ID` por el ID de la cuenta de AWS.

```
{
   "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. Seleccione Siguiente.

9. En la página Agregar permisos, asocie una política personalizada o haga lo siguiente:

   a. En el cuadro Filtrar políticas, ingrese `EKSDescribeClusterPolicy` o el nombre de la política que creó anteriormente. Marque la casilla situada a la izquierda del nombre de la política en los resultados de la búsqueda.

   b. En el cuadro Filtrar políticas, ingrese `EKSHybridSSMPolicy` o el nombre de la política que creó anteriormente. Marque la casilla situada a la izquierda del nombre de la política en los resultados de la búsqueda.

   c. En el cuadro Filtrar políticas, escriba `AmazonEC2ContainerRegistryPullOnly`. Marque la casilla situada a la izquierda de `AmazonEC2ContainerRegistryPullOnly` en los resultados de la búsqueda.

   d. En el cuadro Filtrar políticas, escriba `AmazonSSMManagedInstanceCore`. Marque la casilla situada a la izquierda de `AmazonSSMManagedInstanceCore` en los resultados de la búsqueda.

   e. Elija Siguiente.

10. En la página Nombrar, revisar y crear, haga lo siguiente:

   a. En Nombre del rol, ingrese un nombre único para su rol, por ejemplo, `AmazonEKSHybridNodesRole`.

   b. En Description (Descripción), sustituya el texto actual por un texto descriptivo, como `Amazon EKS - Hybrid Nodes role`.

   c. Seleccione Crear rol.

Pasos correspondientes a AWS IAM Roles Anywhere

Para utilizar AWS IAM Roles Anywhere, debe configurar el anclaje de veracidad de AWS IAM Roles Anywhere antes de crear el rol de IAM de nodos híbridos. Para obtener instrucciones, consulte the section called “Configuración de AWS IAM Roles Anywhere”.

1. Abra la consola de Amazon IAM.
2. En el panel de navegación izquierdo, elija Roles.
3. En la página Roles, elija Crear rol.
4. En la página Seleccionar entidad de confianza, haga lo siguiente:
   a. En la sección de tipo de Entidad de confianza, elija Política de confianza personalizada. Pegue lo siguiente en el editor de políticas de confianza personalizadas. Sustituya `TRUST_ANCHOR ARN` por el ARN del anclaje de veracidad que creó en los pasos de the section called “Configuración de AWS IAM Roles Anywhere”. La condición en esta política de confianza limita la capacidad de AWS IAM Roles Anywhere para asumir el rol de IAM de nodos híbridos. Esto permite el intercambio de credenciales temporales de IAM únicamente cuando el nombre de la sesión del rol coincide con el CN especificado en el certificado x509 instalado en los nodos híbridos. Alternativamente, puede utilizar otros atributos del certificado para identificar de manera única el nodo. El atributo de certificado que utilice en la política de confianza debe corresponder al nodeName que haya establecido en la configuración de nodeadm. Para obtener más información, consulte la the section called “Nodeadm de nodos híbridos”.

```
{
    "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. Seleccione Siguiente.

5. En la página Agregar permisos, asocie una política personalizada o haga lo siguiente:
   a. En el cuadro Filtrar políticas, ingrese `EKSDescribeClusterPolicy` o el nombre de la política que creó anteriormente. Marque la casilla situada a la izquierda del nombre de la política en los resultados de la búsqueda.
   b. En el cuadro Filtrar políticas, escriba `AmazonEC2ContainerRegistryPullOnly`. Marque la casilla situada a la izquierda de `AmazonEC2ContainerRegistryPullOnly` en los resultados de la búsqueda.
   c. Elija Siguiente.
6. En la página Nombrar, revisar y crear, haga lo siguiente:
   a. En Nombre del rol, ingrese un nombre único para su rol, por ejemplo, `AmazonEKSHybridNodesRole`.
   b. En Description (Descripción), sustituya el texto actual por un texto descriptivo, como `Amazon EKS - Hybrid Nodes role`.

c. Seleccione Crear rol.

Cómo crear un clúster de Amazon EKS con nodos híbridos

En este tema se ofrece información general sobre las opciones disponibles y se describen los aspectos que se deben tener en cuenta al crear un clúster de Amazon EKS habilitado para nodos híbridos. Los Nodos híbridos de EKS cuentan con el mismo soporte de versiones de Kubernetes que los clústeres de Amazon EKS con nodos en la nube, incluidos los soportes estándar y extendido.

Si no tiene previsto usar los Nodos híbridos de EKS, consulte la documentación principal para crear clústeres de Amazon EKS en the section called “Creación de un clúster ”.

Requisitos previos

- the section called “Requisitos previos” cumplidos. Antes de crear el clúster habilitado para los nodos híbridos, debe tener identificados los CIDR del nodo en las instalaciones y, opcionalmente, los de los pods. También necesita haber creado la VPC y las subredes según los requisitos de EKS y de los nodos híbridos, así como configurar un grupo de seguridad con reglas de entrada para los CIDR en las instalaciones y, opcionalmente, los de los pods. Para obtener más información sobre estos requisitos previos, consulte the section called “Preparación de las redes”.
- La versión más reciente de la Interfaz de Línea de Comandos de AWS (AWS CLI) instalada y configurada en el dispositivo. Para comprobar su versión actual, utilice `aws --version`. Los administradores de paquetes, tales como yum, apt-get o Homebrew para macOS suelen estar atrasados varias versiones respecto de la versión de AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación o actualización de la versión más reciente de AWS CLI y Configuración de los ajustes de AWS CLI en la Guía del usuario de la Interfaz de la línea de comandos de AWS.
- Una entidad principal de IAM con permisos para crear roles de IAM y asociar políticas, así como para crear y describir clústeres de EKS

Consideraciones

- El clúster debe utilizar el modo `API` o `API_AND_CONFIG_MAP` de autenticación de clústeres.
- El clúster debe utilizar la familia de direcciones IPv4.
- El clúster debe utilizar conectividad de punto de conexión de clúster pública o privada. El clúster no puede utilizar la conectividad de punto de conexión de clúster “pública y privada”, ya que el punto

de conexión del servidor de la API de Kubernetes de Amazon EKS se resolverá en las IP públicas de los nodos híbridos que se ejecutan fuera de la VPC.

- Se admite la autenticación OIDC para clústeres de EKS con nodos híbridos.
- Puede añadir, cambiar o eliminar la configuración de nodos híbridos de un clúster existente. Para obtener más información, consulte the section called “Clúster existente”.

Paso 1: creación de un rol de IAM del clúster

Si ya cuenta con un rol de IAM del clúster o va a crear el clúster con `eksctl` o AWS CloudFormation, puede omitir este paso. De forma predeterminada, `eksctl` y la plantilla de AWS CloudFormation crean el rol de IAM del clúster en su nombre.

1. Ejecute el siguiente comando para crear un archivo de política de confianza JSON de IAM.

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

2. Creación del rol de IAM del clúster de Amazon EKS. Si es necesario, introduzca eks-cluster-role-trust-policy.json con la ruta del equipo en la que escribió el archivo en el paso anterior. El comando asocia la política de confianza creada en el paso anterior al rol. Para crear un rol de IAM, a la entidad principal de IAM que está creando el rol se le debe asignar la acción `iam:CreateRole` (permiso).

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

3. Puede asignar la política administrada de Amazon EKS o bien crear su propia política personalizada. Para conocer los permisos mínimos que debe utilizar en su política personalizada, consulte the section called “Rol de IAM de nodo”. Adjunte la política administrada de IAM por

Amazon EKS denominada `AmazonEKSClusterPolicy` al rol. Para adjuntar una política de IAM a una entidad principal de IAM, se debe asignar una de las siguientes acciones de IAM (permisos) a la entidad principal que adjunta la política: `iam:AttachUserPolicy` o `iam:AttachRolePolicy`.

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

Paso 2: Creación del clúster habilitado para nodos híbridos

Puede crear un clúster mediante:

- eksctl
- AWS CloudFormation
- AWS CLI
- Consola de administración de AWS

Cómo crear el clúster habilitado para nodos híbridos: eksctl

Debe instalar la versión más reciente de la herramienta de línea de comandos de `eksctl`. Para instalar o actualizar `eksctl`, consulte la sección de Instalación en la documentación de `eksctl`.

1. Cree `cluster-config.yaml` para definir un clúster IPv4 de Amazon EKS habilitado para nodos híbridos. Sustituya los siguientes valores en el `cluster-config.yaml`. Para obtener una lista completa de los ajustes, consulte la documentación de eksctl.
   a. Reemplace `CLUSTER_NAME` por el nombre del clúster. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster.
   b. Reemplace `AWS_REGION` por la región de AWS en la que desea crear el clúster.
   c. Reemplace `K8S_VERSION` por cualquier versión compatible con Amazon EKS.
   d. Sustituya `CREDS_PROVIDER` por `ssm` o `ira` según el proveedor de credenciales que configuró en los pasos correspondientes a the section called “Cómo preparar las credenciales”.
   e. Sustituya `CA_BUNDLE_CERT` si el proveedor de credenciales está configurado en `ira`, que utiliza AWS IAM Roles Anywhere como proveedor de credenciales. El CA_BUNDLE_CERT

es el contenido del certificado de la entidad de certificación (CA) y depende de la entidad de certificación que elija. El certificado debe estar en formato Privacy Enhanced Mail (PEM).

f. Sustituya `GATEWAY_ID` por el ID de la puerta de enlace privada virtual o de la puerta de enlace de tránsito que se va a asociar a la VPC.

g. Sustituya `REMOTE_NODE_CIDRS` por el CIDR del nodo en las instalaciones para los nodos híbridos.

h. Sustituya `REMOTE_POD_CIDRS` por el CIDR del pod en las instalaciones para las cargas de trabajo que se ejecutan en nodos híbridos. De otro modo, elimine la línea de la configuración si no va a ejecutar webhooks en nodos híbridos. Debe configurar los `REMOTE_POD_CIDRS` si la CNI no utiliza la traducción de direcciones de red (NAT) ni la ocultación de las direcciones IP de los pods cuando el tráfico del pod sale de los hosts en las instalaciones. Debe configurar `REMOTE_POD_CIDRS` si ejecuta webhooks en nodos híbridos. Consulte the section called “Configuración de webhooks” para obtener más información.

i. Los bloques de CIDR del pod y del nodo en las instalaciones deben cumplir los siguientes requisitos:

   i. Estar dentro de uno de los siguientes rangos RFC-1918 de IPv4: `10.0.0.0/8`, `172.16.0.0/12` o `192.168.0.0/16`.

   ii. Que no se solapen entre sí, el `VPC CIDR` del clúster o el CIDR de IPv4 del servicio de Kubernetes.

```
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. Use el siguiente comando:

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

   El aprovisionamiento de clústeres tarda varios minutos. Mientras se crea el clúster, aparecen varias líneas de salida. La última línea de salida es similar a la siguiente línea de ejemplo.

```
[#]  EKS cluster "CLUSTER_NAME" in "REGION" region is ready
```

3. Continúe con the section called “Paso 3: actualización de kubeconfig”.

Cómo crear un clúster con nodos híbridos habilitados: AWS CloudFormation

La pila de CloudFormation crea el rol de IAM del clúster de EKS y un clúster de EKS con la `RemoteNodeNetwork` y la `RemotePodNetwork` que especifique. Modifique la plantilla de CloudFormation si necesita personalizar configuraciones para el clúster de EKS que no están expuestas en la plantilla de CloudFormation.

1. Descargar la plantilla de CloudFormation.

```
curl -OL 'https://raw.githubusercontent.com/aws/eks-hybrid/refs/heads/main/example/
hybrid-eks-cfn.yaml'
```

2. Cree un `cfn-eks-parameters.json` y especifique la configuración de cada valor.
   a. `CLUSTER_NAME`: el nombre del clúster de EKS que se va a crear
   b. `CLUSTER_ROLE_NAME`: el nombre del rol de IAM del clúster de EKS que se va a crear. El valor predeterminado de la plantilla es “EKSClusterRole”.
   c. `SUBNET1_ID`: el ID de la primera subred que creó en los pasos para cumplir los requisitos previos
   d. `SUBNET2_ID`: el ID de la segunda subred que creó en los pasos para cumplir los requisitos previos
   e. `SG_ID`: el ID del grupo de seguridad que creó en los pasos para cumplir los requisitos previos
   f. `REMOTE_NODE_CIDRS`: el CIDR del nodo en las instalaciones para los nodos híbridos
   g. `REMOTE_POD_CIDRS`: el CIDR del pod en las instalaciones para cargas de trabajo que se ejecutan en nodos híbridos. Debe configurar los `REMOTE_POD_CIDRS` si la CNI no utiliza la traducción de direcciones de red (NAT) ni la ocultación de las direcciones IP de los pods cuando el tráfico del pod sale de los hosts en las instalaciones. Debe configurar

REMOTE_POD_CIDRS si ejecuta webhooks en nodos híbridos. Consulte the section called "Configuración de webhooks" para obtener más información.

h. Los bloques de CIDR del pod y del nodo en las instalaciones deben cumplir los siguientes requisitos:

   i. Estar dentro de uno de los siguientes rangos RFC-1918 de IPv4: 10.0.0.0/8, 172.16.0.0/12 o 192.168.0.0/16.

   ii. Que no se solapen entre sí, el VPC CIDR del clúster o el CIDR de IPv4 del servicio de Kubernetes.

i. CLUSTER_AUTH: el modo de autenticación del clúster. Los valores válidos son API y API_AND_CONFIG_MAP. El valor predeterminado que aparece en la plantilla es API_AND_CONFIG_MAP.

j. CLUSTER_ENDPOINT: la conectividad de punto de conexión del clúster. Los valores válidos son "Pública" o "Privada". El valor predeterminado que aparece en la plantilla es Privada, lo que significa que solo se podrá establecer conexión con el punto de conexión de la API de Kubernetes desde dentro de la VPC.

k. K8S_VERSION: la versión de Kubernetes que se va a utilizar para el clúster. Consulte Versiones compatibles de Amazon EKS.

```
{
  "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. Implementación de la pila de CloudFormation. Sustituya STACK_NAME por el nombre de la pila de CloudFormation y AWS_REGION por la región de AWS en la que se creará el clúster.

```
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
```

El aprovisionamiento de clústeres tarda varios minutos. Puede comprobar el estado de la pila con el siguiente comando. Sustituya `STACK_NAME` por el nombre de la pila de CloudFormation y `AWS_REGION` por la región de AWS en la que se creará el clúster.

```
aws cloudformation describe-stacks \
    --stack-name STACK_NAME \
    --region AWS_REGION \
    --query 'Stacks[].StackStatus'
```

4. Continúe con the section called “Paso 3: actualización de kubeconfig”.

Cómo crear el clúster habilitado para nodos híbridos: AWS CLI

1. Ejecute el siguiente comando para crear un clúster de EKS habilitado para nodos híbridos. Antes de ejecutar el comando, sustituya los siguientes valores por la configuración deseada. Para obtener una lista completa de los ajustes, consulte la documentación de the section called “Creación de un clúster ”.
   a. `CLUSTER_NAME`: el nombre del clúster de EKS que se va a crear
   b. `AWS_REGION`: región de AWS en la que se creará el clúster.
   c. `K8S_VERSION`: la versión de Kubernetes que se va a utilizar para el clúster. Consulte Versiones compatibles de Amazon EKS.
   d. `ROLE_ARN`: el rol de clúster de Amazon EKS que configuró para el clúster. Consulte Rol de IAM del clúster de Amazon EKS para obtener más información.
   e. `SUBNET1_ID`: el ID de la primera subred que creó en los pasos para cumplir los requisitos previos
   f. `SUBNET2_ID`: el ID de la segunda subred que creó en los pasos para cumplir los requisitos previos
   g. `SG_ID`: el ID del grupo de seguridad que creó en los pasos para cumplir los requisitos previos
   h. Puede utilizar `API` y `API_AND_CONFIG_MAP` para el modo de autenticación de acceso al clúster. En el siguiente comando, el modo de autenticación de acceso al clúster está establecido en `API_AND_CONFIG_MAP`.

i. Puede utilizar los parámetros `endpointPublicAccess` y `endpointPrivateAccess` para habilitar o desactivar el acceso público y privado al punto de conexión del servidor de la API de Kubernetes del clúster. En el siguiente comando, `endpointPublicAccess` se establece en falso y `endpointPrivateAccess` se establece en verdadero.

j. `REMOTE_NODE_CIDRS`: el CIDR del nodo en las instalaciones para los nodos híbridos.

k. `REMOTE_POD_CIDRS`: (opcional) el CIDR del pod en las instalaciones para cargas de trabajo que se ejecutan en nodos híbridos.

l. Los bloques de CIDR del pod y del nodo en las instalaciones deben cumplir los siguientes requisitos:

   i. Estar dentro de uno de los siguientes rangos RFC-1918 de IPv4: `10.0.0.0/8`, `172.16.0.0/12` o `192.168.0.0/16`.

   ii. Que no se solapen entre sí, el `VPC CIDR` del clúster de Amazon EKS o el CIDR de IPv4 del servicio de Kubernetes.

```
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. La provisión del clúster puede tardar varios minutos. Puede consultar el estado del clúster con el siguiente comando. Sustituya `CLUSTER_NAME` por el nombre del clúster que va a crear y `AWS_REGION` por la región de AWS en la que se va a crear el clúster. No continúe con el siguiente paso hasta que la salida devuelta sea `ACTIVE`.

```
aws eks describe-cluster \
    --name CLUSTER_NAME \
    --region AWS_REGION \
    --query "cluster.status"
```

3. Continúe con the section called “Paso 3: actualización de kubeconfig”.

Creación del clúster habilitado para nodos híbridos: Consola de administración de AWS

1. Abra la consola de Amazon EKS en Consola de Amazon EKS.
2. Elija Agregar clúster y, a continuación, elija Crear.
3. En la página Configurar clúster rellene los siguientes campos:
   a. Nombre: un nombre único para el clúster. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones bajos. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster.
   b. Rol de IAM del clúster: elija el rol de IAM del clúster de Amazon EKS que creó para permitir que el plano de control de Kubernetes administre los recursos de AWS en su nombre.
   c. Versión de Kubernetes: la versión de Kubernetes que debe utilizarse para el clúster. Le recomendamos que seleccione la versión más reciente, a menos que necesite una versión anterior.
   d. Política de actualización: elija entre extendida o estándar.
      i. Extendida: esta opción es compatible con la versión de Kubernetes durante 26 meses a partir de la fecha de lanzamiento. El periodo de soporte extendido tiene un costo adicional por hora que comenzará una vez finalizado el periodo de soporte estándar. Una vez finalizado el soporte extendido, el clúster se actualizará automáticamente a la siguiente versión.
      ii. Estándar: esta opción es compatible con la versión de Kubernetes durante 14 meses a partir de la fecha de lanzamiento. No supone ningún costo adicional. Una vez finalizado el soporte estándar, el clúster se actualizará automáticamente a la siguiente versión.
   e. Acceso al clúster: elija permitir o denegar el acceso del administrador al clúster y seleccione un modo de autenticación. Los siguientes modos de autenticación son compatibles con los clústeres habilitados para nodos híbridos.
      i. API de EKS: el clúster obtendrá las entidades principales de IAM autenticadas únicamente de las API de entrada de acceso de EKS.
      ii. API de EKS y ConfigMap: el clúster obtendrá las entidades principales de IAM autenticadas tanto de las API de entrada de acceso de EKS como de `aws-auth` ConfigMap.
   f. Cifrado de secretos: (opcional) elija habilitar el cifrado de secretos de los secretos de Kubernetes con una clave de KMS. También puede habilitarlo después de crear el clúster. Antes de habilitar esta capacidad, asegúrese de estar familiarizado con la información de the section called “Habilitación del cifrado de secretos”.

g. Cambio de zona de ARC: si esta opción está activada, EKS registrará el clúster en el cambio de zona de ARC de modo que pueda recurrir al cambio de zona para desplazar el tráfico fuera de aplicaciones de una zona de disponibilidad.

h. Etiquetas: (opcional) agregue las etiquetas a su clúster. Para obtener más información, consulte the section called “Etiquetado de recursos”.

i. Cuando haya terminado con esta página, seleccione Siguiente.

4. En la página Especificar red seleccione valores para los siguientes campos:

a. VPC: elija una VPC existente que cumpla con the section called “Requisitos de VPC y subred” y los requisitos de los Nodos híbridos de Amazon EKS. Antes de elegir una VPC, recomendamos que esté familiarizado con todos los requisitos y consideraciones que se indican en Consulte los requisitos de red de Amazon EKS para VPC, subredes y nodos híbridos. No puede cambiar la VPC que desea utilizar después de crear un clúster. Si no hay ninguna VPC en la lista, debe crear una primero. Para obtener más información, consulte the section called “Creación de una VPC” y Requisitos de red para los Nodos híbridos de Amazon EKS.

b. Subredes: de forma predeterminada, se preseleccionan todas las subredes disponibles de la VPC especificada en el campo anterior. Debe seleccionar al menos dos.

c. Grupos de seguridad: (opcional) especifique uno o varios grupos de seguridad que desea que Amazon EKS asocie a las interfaces de red que crea. Al menos uno de los grupos de seguridad que especifique debe tener reglas de entrada para los CIDR del nodo en las instalaciones y, opcionalmente, los de los pods. Consulte Requisitos de red de los Nodos híbridos de Amazon EKS para obtener más información. Independientemente de que elija grupos de seguridad o no, Amazon EKS crea un grupo de seguridad que permite la comunicación entre el clúster y la VPC. Amazon EKS asocia este grupo de seguridad, y el que elija, a las interfaces de red que crea. Para obtener más información acerca del grupo de seguridad de clústeres que Amazon EKS crea, consulte the section called “Requisitos del grupo de seguridad”. Puede modificar las reglas del grupo de seguridad de clústeres que Amazon EKS crea.

d. Elija la familia de direcciones IP del clúster: debe elegir IPv4 para los clústeres habilitados para nodos híbridos.

e. (Opcional) Elija Configuración del rango de direcciones IP del servicio de Kubernetes y especifique un Rango de servicio IPv4.

f. Elija Configuración de redes remotas para habilitar nodos híbridos y especifique los CIDR del nodo en las instalaciones y de los pods para los nodos híbridos.

g. Debe configurar el CIDR del pod remoto si la CNI no utiliza la traducción de direcciones de red (NAT) ni la ocultación de las direcciones IP de los pods cuando el tráfico del pod sale de los

hosts en las instalaciones. Debe configurar el CIDR de pod remoto si va a ejecutar webhooks en nodos híbridos.

h. Los bloques de CIDR del pod y del nodo en las instalaciones deben cumplir los siguientes requisitos:

   i. Estar dentro de uno de los siguientes rangos RFC-1918 de IPv4: `10.0.0.0/8`, `172.16.0.0/12` o `192.168.0.0/16`.

   ii. Que no se solapen entre sí, el `VPC CIDR` del clúster o el CIDR de IPv4 del servicio de Kubernetes.

i. Para el Acceso al punto de conexión del clúster, seleccione una opción. Una vez que se crea el clúster, puede cambiar esta opción. En el caso de los clústeres habilitados para nodos híbridos, debe elegir entre Pública o Privada. Antes de seleccionar una opción no predeterminada, asegúrese de familiarizarse con las opciones y sus implicaciones. Para obtener más información, consulte the section called “Acceso al punto de conexión del clúster”.

j. Cuando haya terminado con esta página, seleccione Siguiente.

5. (Opcional) En la página Configuración de la observabilidad, seleccione qué opciones de métricas y registro de planos de control desea activar. De forma predeterminada, cada tipo de registro está desactivado.

   a. Para obtener más información sobre la opción de las métricas de Prometheus, consulte the section called “Métricas de Prometheus”.

   b. Para obtener más información sobre las opciones de registro de control de EKS, consulte the section called “Registros del plano de control”.

   c. Cuando haya terminado con esta página, seleccione Siguiente.

6. En la página Seleccionar complementos, elija los complementos que desea agregar al clúster.

   a. Puede elegir tantos complementos de Amazon EKS y complementos de Marketplace de AWS como necesite. Los complementos de Amazon EKS que no son compatibles con los nodos híbridos están marcados con el mensaje “No es compatible con los nodos híbridos” y cuentan con una regla de antiafinidad que evita que se ejecuten en nodos híbridos. Consulte Configuración de complementos para nodos híbridos para obtener más información. Si los complementos de Marketplace de AWS que quiere instalar no aparecen en la lista, puede buscar los complementos de Marketplace de AWS disponibles al introducir texto en el cuadro de búsqueda. También puede buscar por categoría, proveedor o modelo de precios y, a continuación, elegir los complementos en los resultados de la búsqueda.

b. Algunos complementos, como CoreDNS y kube-proxy, se instalan de forma predeterminada. Si desactiva alguno de los complementos predeterminados, esto puede afectar a su capacidad para ejecutar aplicaciones de Kubernetes.

c. Cuando haya terminado con esta página, seleccione `Next`.

7. En la página Configurar las opciones de complementos seleccionados, seleccione la versión que desee instalar.

   a. Siempre puede actualizar a una versión posterior después de crear el clúster. Puede actualizar la configuración de cada complemento después de crear el clúster. Para obtener más información acerca de la configuración de complementos, consulte the section called “Cómo actualizar un complemento”. Para ver las versiones de complementos compatibles con los nodos híbridos, consulte the section called “Configuración de complementos”.

   b. Cuando haya terminado con esta página, seleccione Siguiente.

8. En la página Revisar y crear, revise la información que introdujo o seleccionó en las páginas anteriores. Si necesita realizar cambios, seleccione Edit (Editar). Cuando esté satisfecho, seleccione Crear. El campo Estado muestra CREANDO mientras se aprovisiona el clúster. El aprovisionamiento de clústeres tarda varios minutos.

9. Continúe con the section called “Paso 3: actualización de kubeconfig”.

Paso 3: actualización de kubeconfig

Si ha creado el clúster mediante `eksctl`, puede omitir este paso. Esto se debe a que `eksctl` ya ha completado este paso. Habilite `kubectl` para comunicarse con el clúster al agregar contexto nuevo al archivo de configuración `kubectl`. Para obtener más información acerca de cómo crear y actualizar el archivo, consulte the section called “Acceso al clúster con kubectl”.

```
aws eks update-kubeconfig --name CLUSTER_NAME --region AWS_REGION
```

Un ejemplo de salida sería el siguiente.

```
Added new context arn:aws:eks:AWS_REGION:111122223333:cluster/CLUSTER_NAME to /home/
username/.kube/config
```

Confirme la comunicación con el clúster ejecutando el siguiente comando.

```
kubectl get svc
```

Un ejemplo de salida sería el siguiente.

```
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   28h
```

Paso 4: configuración del clúster

Para el siguiente paso, consulte the section called “Cómo preparar el acceso al clúster” para habilitar el acceso para que los nodos híbridos se unan al clúster.

Habilitación de nodos híbridos en un clúster de Amazon EKS existente o modificación de la configuración

Este tema ofrece una descripción general de las opciones disponibles y explica qué aspectos debe tener en cuenta al agregar, modificar o eliminar la configuración de nodos híbridos de un clúster de Amazon EKS.

Para permitir que un clúster de Amazon EKS utilice nodos híbridos, añada los rangos CIDR de direcciones IP del nodo en las instalaciones y, opcionalmente, de la red de pods en la configuración `RemoteNetworkConfig`. EKS usa esta lista de CIDR para habilitar la conectividad entre el clúster y las redes en las instalaciones. Para obtener una lista completa de las opciones a la hora de actualizar la configuración del clúster, consulte UpdateClusterConfig en la referencia de la API de Amazon EKS.

Puede realizar cualquiera de las siguientes acciones en la configuración de red de los nodos híbridos de EKS en un clúster:

- Agregar una configuración de red remota para habilitar los nodos híbridos de EKS en un clúster existente.
- Agregar, cambiar o eliminar las redes de nodos remotos o las redes de pods remotos de un clúster existente.
- Eliminar todos los rangos CIDR de redes de nodos remotos para deshabilitar los nodos híbridos de EKS en un clúster existente.

Requisitos previos

- Antes de habilitar su clúster de Amazon EKS para nodos híbridos, asegúrese de que su entorno cumpla los requisitos descritos en the section called “Requisitos previos” y detallados en the

section called “Preparación de las redes”, the section called “Cómo preparar el sistema operativo” ythe section called “Cómo preparar las credenciales”.

- El clúster debe utilizar la familia de direcciones IPv4.
- El clúster debe utilizar el modo `API` o `API_AND_CONFIG_MAP` de autenticación de clústeres. El proceso para modificar el modo de autenticación del clúster se describe en the section called “Modo de autenticación”.
- Recomendamos utilizar acceso mediante un punto de conexión público o privado para el punto de conexión del servidor de la API de Kubernetes en Amazon EKS, pero no ambos. Si elige “Público y privado”, el punto de conexión del servidor de la API de Kubernetes de Amazon EKS siempre se resolverá en las IP públicas de los nodos híbridos que se ejecutan fuera de la VPC, lo que puede impedir que los nodos híbridos se unan al clúster. El proceso para modificar el acceso de red al clúster se describe en the section called “Acceso al punto de conexión del clúster”.
- La versión más reciente de la Interfaz de Línea de Comandos de AWS (AWS CLI) instalada y configurada en el dispositivo. Para comprobar su versión actual, utilice `aws --version`. Los administradores de paquetes, tales como yum, apt-get o Homebrew para macOS suelen estar atrasados varias versiones respecto de la versión de AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación o actualización a la versión más reciente de AWS CLI y Configuración de los ajustes de AWS CLI en la Guía del usuario de la interfaz de la línea de comandos de AWS.
- Una entidad principal de IAM con permiso para llamar a UpdateClusterConfig en su clúster de Amazon EKS.
- Actualice los complementos a versiones compatibles con los nodos híbridos. Para ver las versiones de complementos compatibles con los nodos híbridos, consulte the section called “Configuración de complementos”.
- Si está ejecutando complementos que no son compatibles con los nodos híbridos, asegúrese de que el complemento DaemonSet o Deployment tenga la siguiente regla de afinidad para evitar la implementación en nodos híbridos. Añada la siguiente regla de afinidad si aún no está presente.

```
affinity:
  nodeAffinity:
    requiredDuringSchedulingIgnoredDuringExecution:
      nodeSelectorTerms:
      - matchExpressions:
        - key: eks.amazonaws.com/compute-type
          operator: NotIn
          values:
```

```
            - hybrid
```

Consideraciones

El objeto `remoteNetworkConfig` JSON tiene el siguiente comportamiento durante una actualización:

- Cualquier parte existente de la configuración que no especifique permanece sin cambios. Si no especifica ninguna de `remoteNodeNetworks` o `remotePodNetworks`, esa parte seguirá siendo la misma.
- Si modificará las listas `remoteNodeNetworks` o `remotePodNetworks` de CIDR, debe especificar la lista completa de CIDR que desee incluir en la configuración final. Al especificar un cambio en la lista `remoteNodeNetworks` o `remotePodNetworks` de CIDR, EKS reemplaza la lista original durante la actualización.
- Los bloques de CIDR del pod y del nodo en las instalaciones deben cumplir los siguientes requisitos:
  1. Debe estar comprendido en uno de los intervalos RFC-1918 de IPv4: 10.0.0.0/8, 172.16.0.0/12 o 192.168.0.0/16.
  2. Que no se solapen entre sí, todos los CIDR de la VPC del clúster de Amazon EKS o el CIDR de IPv4 del servicio de Kubernetes.

Habilitación de nodos híbridos en un clúster existente

Puede habilitar los nodos híbridos de EKS en un clúster existente con lo siguiente:

- AWS CloudFormation
- CLI de AWS
- Consola de administración de AWS

Habilitación de los nodos híbridos de EKS en un clúster existente: AWS CloudFormation

1. Para habilitar los nodos híbridos de EKS en su clúster, añada `RemoteNodeNetwork` y (opcional) `RemotePodNetwork` a la plantilla de CloudFormation y actualice la pila. Tenga en cuenta que `RemoteNodeNetwork` es una lista con un máximo de un elemento `Cidrs`, y `Cidrs` es una lista de varios rangos de CIDR de IP.

```
RemoteNetworkConfig:
  RemoteNodeNetworks:
    - Cidrs: [RemoteNodeCIDR]
  RemotePodNetworks:
    - Cidrs: [RemotePodCIDR]
```

2. Siga en the section called “Cómo preparar el acceso al clúster”.

Habilitación de los nodos híbridos de EKS en un clúster existente: AWS CLI

1. Ejecute el siguiente comando para habilitar `RemoteNetworkConfig` para nodos híbridos de EKS para su clúster de EKS. Antes de ejecutar el comando, sustituya los siguientes valores por la configuración deseada. Para obtener una lista completa de los ajustes, consulte UpdateClusterConfig en la referencia de la API de Amazon EKS.
   a. `CLUSTER_NAME`: nombre del clúster de EKS que se va a actualizar.
   b. `AWS_REGION`: región de AWS en la que se ejecuta el clúster de EKS.
   c. `REMOTE_NODE_CIDRS`: el CIDR del nodo en las instalaciones para los nodos híbridos.
   d. `REMOTE_POD_CIDRS`: (opcional) el CIDR del pod en las instalaciones para cargas de trabajo que se ejecutan en nodos híbridos.

```
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. La actualización del clúster puede tardar varios minutos. Puede consultar el estado del clúster con el siguiente comando. Sustituya `CLUSTER_NAME` por el nombre del clúster que va a modificar y `AWS_REGION` por la región de AWS en la que se va a ejecutar el clúster. No continúe con el siguiente paso hasta que la salida devuelta sea `ACTIVE`.

```
aws eks describe-cluster \
    --name CLUSTER_NAME \
    --region AWS_REGION \
    --query "cluster.status"
```

3. Siga en the section called “Cómo preparar el acceso al clúster”.

Habilitación de los nodos híbridos de EKS en un clúster existente: Consola de administración de AWS

1. Abra la consola de Amazon EKS en Consola de Amazon EKS.
2. Elija el nombre del clúster para mostrar la información del clúster.
3. En la pestaña Redes, elija Administrar.
4. En el menú desplegable, elija Redes remotas.
5. Elija Configuración de redes remotas para habilitar nodos híbridos y especifique los CIDR del nodo en las instalaciones y de los pods para los nodos híbridos.
6. Elija Save changes (Guardar cambios) para terminar. Espere a que el estado del clúster vuelva a ser Activo.
7. Siga en the section called “Cómo preparar el acceso al clúster”.

Actualización de la configuración de los nodos híbridos en un clúster existente

Puede modificar `remoteNetworkConfig` en un clúster híbrido existente mediante cualquiera de las siguientes opciones:

- AWS CloudFormation
- CLI de AWS
- Consola de administración de AWS

Actualización de la configuración en un clúster existente: AWS CloudFormation

1. Actualice la plantilla de CloudFormation con los nuevos valores de CIDR de red.

```
RemoteNetworkConfig:
  RemoteNodeNetworks:
    - Cidrs: [NEW_REMOTE_NODE_CIDRS]
  RemotePodNetworks:
    - Cidrs: [NEW_REMOTE_POD_CIDRS]
```

> **Note**
>
> Al actualizar las listas CIDR de `RemoteNodeNetworks` o `RemotePodNetworks`, incluya todos los CIDR (nuevos y existentes). EKS reemplaza la lista completa durante las

actualizaciones. Si se omiten estos campos de la solicitud de actualización, se conservan las configuraciones existentes.

2. Actualice la pila de CloudFormation con la plantilla modificada y espere a que se complete la actualización de la pila.

Actualización de la configuración híbrida en un clúster existente: AWS CLI

1. Para modificar los CIDR de la red remota, ejecute el siguiente comando. Sustituya los valores por los ajustes que desee:

```
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**
>
> Al actualizar las listas CIDR de `remoteNodeNetworks` o `remotePodNetworks`, incluya todos los CIDR (nuevos y existentes). EKS reemplaza la lista completa durante las actualizaciones. Si se omiten estos campos de la solicitud de actualización, se conservan las configuraciones existentes.

2. Espere a que el estado del clúster vuelva a ser ACTIVO antes de proceder.

Actualización de la configuración híbrida en un clúster existente: Consola de administración de AWS

1. Abra la consola de Amazon EKS en Consola de Amazon EKS.
2. Elija el nombre del clúster para mostrar la información del clúster.
3. En la pestaña Redes, elija Administrar.
4. En el menú desplegable, elija Redes remotas.
5. Actualice los CIDR en `Remote node networks` y `Remote pod networks - Optional` según sea necesario.
6. Seleccione Guardar cambios y espere a que el estado del clúster vuelva a ser Activo.

Inhabilitación de los nodos híbridos en un clúster existente

Puede inhabilitar los nodos híbridos de EKS en un clúster existente con lo siguiente:

- AWS CloudFormation
- CLI de AWS
- Consola de administración de AWS

Inhabilitación de los nodos híbridos de EKS en un clúster existente: AWS CloudFormation

1. Para inhabilitar los nodos híbridos de EKS en su clúster, configure `RemoteNodeNetworks` y `RemotePodNetworks` para vaciar las matrices en su plantilla de CloudFormation y actualice la pila.

```
RemoteNetworkConfig:
  RemoteNodeNetworks: []
  RemotePodNetworks: []
```

Inhabilitación de los nodos híbridos de EKS en un clúster existente: AWS CLI

1. Ejecute el siguiente comando para eliminar `RemoteNetworkConfig` de su clúster de EKS. Antes de ejecutar el comando, sustituya los siguientes valores por la configuración deseada. Para obtener una lista completa de los ajustes, consulte UpdateClusterConfig en la referencia de la API de Amazon EKS.
   a. `CLUSTER_NAME`: nombre del clúster de EKS que se va a actualizar.
   b. `AWS_REGION`: región de AWS en la que se ejecuta el clúster de EKS.

```
aws eks update-cluster-config \
    --name CLUSTER_NAME \
    --region AWS_REGION \
    --remote-network-config '{"remoteNodeNetworks":[],"remotePodNetworks":[]}'
```

2. La actualización del clúster puede tardar varios minutos. Puede consultar el estado del clúster con el siguiente comando. Sustituya `CLUSTER_NAME` por el nombre del clúster que va a modificar y `AWS_REGION` por la región de AWS en la que se va a ejecutar el clúster. No continúe con el siguiente paso hasta que la salida devuelta sea `ACTIVE`.

```
aws eks describe-cluster \
```

```
    --name CLUSTER_NAME \
    --region AWS_REGION \
    --query "cluster.status"
```

Inhabilitación de los nodos híbridos de EKS en un clúster existente: Consola de administración de AWS

1. Abra la consola de Amazon EKS en Consola de Amazon EKS.
2. Elija el nombre del clúster para mostrar la información del clúster.
3. En la pestaña Redes, elija Administrar.
4. En el menú desplegable, elija Redes remotas.
5. Seleccione Configurar redes remotas para habilitar los nodos híbridos y elimine todos los CIDR que aparecen en `Remote node networks` y `Remote pod networks - Optional`.
6. Elija Save changes (Guardar cambios) para terminar. Espere a que el estado del clúster vuelva a ser Activo.

Cómo preparar el acceso al clúster para los nodos híbridos

Antes de conectar los nodos híbridos al clúster de Amazon EKS, debe habilitar el rol de IAM de nodos híbridos con permisos de Kubernetes para unirse al clúster. Consulte the section called "Cómo preparar las credenciales" para obtener información sobre cómo crear el rol de IAM de nodos híbridos. Amazon EKS admite dos formas de asociar entidades principales de IAM al control de acceso basado en roles (RBAC) de Kubernetes: las entradas de acceso de Amazon EKS y `aws-auth` ConfigMap. Para obtener más información sobre la administración del acceso a Amazon EKS, consulte the section called "Acceso a la API de Kubernetes".

Utilice los siguientes procedimientos para asociar el rol de IAM de nodos híbridos a los permisos de Kubernetes. Para utilizar las entradas de acceso de Amazon EKS, el clúster se debe haber creado con los modos de autenticación `API` o `API_AND_CONFIG_MAP`. Para utilizar `aws-auth` ConfigMap, el clúster se debe haber creado con el modo de autenticación `API_AND_CONFIG_MAP`. El modo de autenticación solo con `CONFIG_MAP` no es compatible con los clústeres de Amazon EKS habilitados para nodos híbridos.

Uso de entradas de acceso de Amazon EKS para el rol de IAM de nodos híbridos

Existe un tipo de entrada de acceso de Amazon EKS para nodos híbridos denominado HYBRID_LINUX que se puede utilizar con un rol de IAM. Con este tipo de entrada de acceso, el

nombre de usuario se establece automáticamente en system:node:{{SessionName}}. Para obtener más información sobre cómo crear entradas de acceso, consulte the section called “Creación de entradas de acceso”.

AWS CLI

1. Debe tener la versión más reciente de AWS CLI instalada y configurada en el dispositivo. Para comprobar su versión actual, utilice `aws --version`. Los administradores de paquetes, tales como yum, apt-get o Homebrew para macOS suelen estar atrasados varias versiones respecto de la versión de AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y configuración rápida con aws configure en la Guía del usuario de la interfaz de línea de comandos de AWS.
2. Cree la entrada de acceso con el siguiente comando. Sustituya CLUSTER_NAME por el nombre del clúster y HYBRID_NODES_ROLE_ARN por el ARN del rol que creó en los pasos para the section called “Cómo preparar las credenciales”.

```
aws eks create-access-entry --cluster-name CLUSTER_NAME \
    --principal-arn HYBRID_NODES_ROLE_ARN \
    --type HYBRID_LINUX
```

Consola de administración de AWS

1. Abra la consola de Amazon EKS en Consola de Amazon EKS.
2. Elija el nombre del clúster habilitado para nodos híbridos.
3. Elija la pestaña Acceso.
4. Elija Crear entrada de acceso.
5. En entidad principal de IAM, seleccione el rol de IAM de nodos híbridos que creó en los pasos correspondientes a the section called “Cómo preparar las credenciales”.
6. En Tipo, seleccione Hybrid Linux.
7. (Opcional) En Etiquetas, asigne etiquetas a la entrada de acceso. Por ejemplo, para facilitar la búsqueda de todos los recursos con la misma etiqueta.
8. Elija Omitir para revisar y crear. No puede agregar políticas a la entrada de acceso de Hybrid Linux ni cambiar su alcance de acceso.

9. Revise la configuración de su entrada de acceso. Si algo parece incorrecto, seleccione Anterior para volver a repasar los pasos y corregir el error. Si la configuración es correcta, seleccione Crear.

Uso de aws-auth ConfigMap para el rol de IAM de nodos híbridos

En los siguientes pasos, creará o actualizará `aws-auth` ConfigMap con el ARN del rol de IAM de nodos híbridos que creó en los pasos correspondientes a the section called “Cómo preparar las credenciales”.

1. Compruebe si cuenta con `aws-auth` ConfigMap existente para el clúster. Tenga en cuenta que si utiliza un archivo `kubeconfig` específico, debe utilizar la marca `--kubeconfig`.

```
kubectl describe configmap -n kube-system aws-auth
```

2. Si aparece un `aws-auth` ConfigMap, actualícelo según sea necesario.
   a. Abra el ConfigMap para fines de edición.

```
kubectl edit -n kube-system configmap/aws-auth
```

   b. Añada una nueva entrada de `mapRoles` según sea necesario. Sustituya `HYBRID_NODES_ROLE_ARN` por el ARN del rol de IAM de nodos híbridos. Tenga en cuenta que `{{SessionName}}` es el formato de plantilla correcto para guardar en el ConfigMap. No lo sustituya por otros valores.

```
data:
  mapRoles: |
  - groups:
    - system:bootstrappers
    - system:nodes
    rolearn: HYBRID_NODES_ROLE_ARN
    username: system:node:{{SessionName}}
```

   c. Guarde el archivo y salga del editor de texto.
3. Si no existe un `aws-auth` ConfigMap para el clúster, créelo con el siguiente comando. Sustituya `HYBRID_NODES_ROLE_ARN` por el ARN del rol de IAM de nodos híbridos. Tenga en cuenta que `{{SessionName}}` es el formato de plantilla correcto para guardar en el ConfigMap. No lo sustituya por otros valores.

```
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
```

Ejecución de cargas de trabajo en las instalaciones en nodos híbridos

En un clúster de EKS con nodos híbridos habilitados, es posible ejecutar aplicaciones en las instalaciones y periféricas en una infraestructura propia con los mismos clústeres, características y herramientas de Amazon EKS que se utilizan en la nube de AWS.

En las siguientes secciones se ofrecen instrucciones paso a paso sobre cómo se utilizan los nodos híbridos.

Temas

- Cómo conectar nodos híbridos
- Conexión de nodos híbridos con Bottlerocket
- Actualización de nodos híbridos para el clúster
- Aplicación de revisiones de actualizaciones de seguridad para nodos híbridos
- Cómo eliminar nodos híbridos

Cómo conectar nodos híbridos

> **Note**
>
> Los siguientes pasos se aplican a nodos híbridos que ejecutan sistemas operativos compatibles, con excepción de Bottlerocket. Para conocer los pasos para conectar un nodo

híbrido que ejecute Bottlerocket, consulte the section called “Conexión de nodos híbridos con Bottlerocket”.

En este tema se explica cómo conectar nodos híbridos a un clúster de Amazon EKS. Una vez que los nodos híbridos se unan al clúster, aparecerán con el estado No listos en la consola de Amazon EKS y en las herramientas compatibles con Kubernetes, como kubectl. Tras completar los pasos que se indican en esta página, proceda a the section called “Cómo configurar una CNI” para preparar los nodos híbridos para ejecutar aplicaciones.

Requisitos previos

Antes de conectar los nodos híbridos al clúster de Amazon EKS, compruebe que se cumplen los requisitos previos indicados.

- Dispone de conectividad de red entre el entorno en las instalaciones con la región de AWS que aloja el clúster de Amazon EKS. Para obtener más información, consulte the section called “Preparación de las redes”.
- Dispone de un sistema operativo compatible con los nodos híbridos instalado en los hosts en las instalaciones. Para obtener más información, consulte the section called “Cómo preparar el sistema operativo”.
- Ha creado el rol de IAM de los nodos híbridos y ha configurado el proveedor de credenciales en las instalaciones (activaciones híbridas de AWS Systems Manager o AWS IAM Roles Anywhere). Para obtener más información, consulte the section called “Cómo preparar las credenciales”.
- Ha creado el clúster de Amazon EKS habilitado para nodos híbridos. Para obtener más información, consulte the section called “Creación de un clúster”.
- Ha asociado el rol de IAM de nodos híbridos a los permisos de control de acceso basado en roles (RBAC) de Kubernetes. Para obtener más información, consulte the section called “Cómo preparar el acceso al clúster”.

Paso 1: Instalación de la CLI (**`nodeadm`**) de nodos híbridos en cada host en las instalaciones

Si incluye la CLI (`nodeadm`) de los Nodos híbridos de Amazon EKS en las imágenes prediseñadas del sistema operativo, puede omitir este paso. Para obtener más información acerca de la versión de los nodos híbridos de `nodeadm`, consulte the section called “Nodeadm de nodos híbridos”.

La versión de nodos híbridos de `nodeadm` está alojada en Amazon S3, con Amazon CloudFront como frontend. Para instalar `nodeadm` en cada host en las instalaciones, puede ejecutar el siguiente comando desde los hosts en las instalaciones.

Para los hosts x86_64:

```
curl -OL 'https://hybrid-assets.eks.amazonaws.com/releases/latest/bin/linux/amd64/
nodeadm'
```

Para los hosts ARM

```
curl -OL 'https://hybrid-assets.eks.amazonaws.com/releases/latest/bin/linux/arm64/
nodeadm'
```

Agregue permiso de archivo ejecutable al binario descargado en cada host.

```
chmod +x nodeadm
```

Paso 2: Instalación de las dependencias de los nodos híbridos con **`nodeadm`**

Si va a instalar las dependencias de los nodos híbridos en las imágenes prediseñadas del sistema operativo, puede omitir este paso. El comando `nodeadm install` se puede usar para instalar todas las dependencias necesarias para los nodos híbridos. Las dependencias de los nodos híbridos incluyen containerd, kubelet, kubectl y los componentes de AWS SSM o AWS IAM Roles Anywhere. Consulte the section called “Nodeadm de nodos híbridos” para obtener más información sobre los componentes y las ubicaciones de los archivos instalados por `nodeadm install`. Consulte the section called “Preparación de las redes” para nodos híbridos para obtener más información sobre los dominios que se deben permitir en el firewall en las instalaciones durante el proceso de `nodeadm install`.

Ejecute el siguiente comando para instalar las dependencias de los nodos híbridos en el host en las instalaciones. El comando que aparece a continuación se debe ejecutar con un usuario que tenga privilegios de raíz/sudo en el host.

> **Important**
>
> La CLI (`nodeadm`) de nodos híbridos se debe ejecutar con un usuario que tenga acceso sudo/raíz en el host.

- Sustituya `K8S_VERSION` por la versión secundaria de Kubernetes del clúster de Amazon EKS, por ejemplo `1.31`. Consulte las versiones compatibles de Amazon EKS para obtener una lista de las versiones de Kubernetes compatibles.
- Sustituya `CREDS_PROVIDER` por el proveedor de credenciales en las instalaciones que utiliza. Los valores válidos son `ssm` para AWS SSM y `iam-ra` para AWS IAM Roles Anywhere.

```
nodeadm install K8S_VERSION --credential-provider CREDS_PROVIDER
```

Paso 3: Conexión de los nodos híbridos al clúster

Antes de conectar los nodos híbridos al clúster, asegúrese de haber permitido el acceso necesario en el firewall en las instalaciones y en el grupo de seguridad del clúster para la comunicación entre el plano de control de Amazon EKS y el nodo híbrido. La mayoría de los problemas de este paso están relacionados con la configuración del firewall, la configuración del grupo de seguridad o la configuración de los roles de IAM de los nodos híbridos.

> **Important**
>
> La CLI (`nodeadm`) de nodos híbridos se debe ejecutar con un usuario que tenga acceso sudo/raíz en el host.

1. Cree un archivo `nodeConfig.yaml` en cada host con los valores de la implementación. Para obtener una descripción completa de los ajustes de configuración disponibles, consulte the section called “Nodeadm de nodos híbridos”. Si el rol de IAM de los nodos híbridos no tiene permiso para la acción `eks:DescribeCluster`, debe proporcionar el punto de conexión de la API de Kubernetes, el paquete CA del clúster y el CIDR IPv4 del servicio de Kubernetes en la sección del clúster del `nodeConfig.yaml`.
   a. Utilice el `nodeConfig.yaml` de ejemplo que aparece a continuación si utiliza activaciones híbridas de AWS SSM para el proveedor de credenciales en las instalaciones.
      i. Reemplace `CLUSTER_NAME` por el nombre del clúster.
      ii. Sustituya `AWS_REGION` por la región de AWS en la que se aloja el clúster. Por ejemplo, `us-west-2`.

iii. Sustituya `ACTIVATION_CODE` por el código de activación que recibió al crear la activación híbrida de AWS SSM. Para obtener más información, consulte the section called “Cómo preparar las credenciales”.

iv. Sustituya `ACTIVATION_ID` por el ID de activación que recibió al crear la activación híbrida de AWS SSM. Puede recuperar esta información desde la consola de AWS Systems Manager o desde el comando `aws ssm describe-activations` de AWS CLI.

```
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name: CLUSTER_NAME
    region: AWS_REGION
  hybrid:
    ssm:
      activationCode: ACTIVATION_CODE
      activationId: ACTIVATION_ID
```

b. Utilice el `nodeConfig.yaml` de ejemplo que aparece a continuación si utiliza AWS IAM Roles Anywhere para el proveedor de credenciales en las instalaciones.

i. Reemplace `CLUSTER_NAME` por el nombre del clúster.

ii. Sustituya `AWS_REGION` por la región de AWS en la que se aloja el clúster. Por ejemplo, `us-west-2`.

iii. Sustituya `NODE_NAME` por el nombre del nodo. El nombre del nodo debe coincidir con el CN del certificado del host si configuró la política de confianza del rol de IAM de nodos híbridos con la condición de recurso `"sts:RoleSessionName": "${aws:PrincipalTag/x509Subject/CN}"`. El `nodeName` que utilice no puede tener más de 64 caracteres.

iv. Sustituya `TRUST_ANCHOR_ARN` por el ARN del anclaje de veracidad que configuró en los pasos que se indican en Cómo preparar credenciales para nodos híbridos.

v. Sustituya `PROFILE_ARN` por el ARN del anclaje de veracidad que configuró en los pasos de the section called “Cómo preparar las credenciales”.

vi. Sustituya `ROLE_ARN` por el ARN del rol de IAM de nodos híbridos.

vii. Sustituya `CERTIFICATE_PATH` por la ruta en disco al certificado del nodo. Si no se especifica, el valor predeterminado es `/etc/iam/pki/server.pem`.

viii. Sustituya `KEY_PATH` por la ruta en disco a la clave privada del certificado. Si no se especifica, el valor predeterminado es `/etc/iam/pki/server.key`.

```
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. Ejecute el comando `nodeadm init` con el `nodeConfig.yaml` para conectar los nodos híbridos al clúster de Amazon EKS.

```
nodeadm init -c file://nodeConfig.yaml
```

Si el comando anterior se ejecuta correctamente, el nodo híbrido se habrá unido a su clúster de Amazon EKS. Puede verificar esto en la consola de Amazon EKS. Para ello, vaya a la pestaña Computación del clúster (asegúrese de que la entidad principal de IAM tenga permisos de visualización) o con `kubectl get nodes`.

> **Important**
>
> Los nodos tendrán el estado `Not Ready`, que es el esperado, y se debe a la falta de una CNI que se ejecute en los nodos híbridos. Si los nodos no se unieron al clúster, consulte the section called “Solución de problemas”.

Paso 4: Configuración de una CNI para los nodos híbridos

Para que los nodos híbridos estén preparados para ejecutar aplicaciones, continúe con los pasos que se indican en the section called “Cómo configurar una CNI”.

Conexión de nodos híbridos con Bottlerocket

Este tema describe cómo conectar nodos híbridos que ejecutan Bottlerocket a un clúster de Amazon EKS. Bottlerocket es una distribución de Linux de código abierto patrocinada y respaldada por AWS. Bottlerocket está diseñado específicamente para alojar cargas de trabajo de contenedores. Con Bottlerocket, puede mejorar la disponibilidad de las implementaciones en contenedores y reducir los costos operativos mediante la automatización de las actualizaciones de la infraestructura de contenedores. Bottlerocket incluye solo el software esencial para ejecutar los contenedores, lo que mejora el uso de los recursos, reduce las amenazas a la seguridad y reduce los gastos de administración.

Solo se admiten las variantes de VMware de Bottlerocket a partir de la versión v1.37.0 con los Nodos híbridos de EKS. Las variantes de VMware de Bottlerocket están disponibles para las versiones v1.28 y posteriores de Kubernetes. Las imágenes del sistema operativo para estas variantes incluyen kubelet, containerd, aws-iam-authenticator y otros requisitos previos de software para los Nodos híbridos de EKS. Puede configurar estos componentes mediante un archivo de configuración de Bottlerocket que incluya datos de usuario codificados en base64 para los contenedores de arranque y administración de Bottlerocket. Configurar estos ajustes permite que Bottlerocket utilice el proveedor de credenciales de los nodos híbridos para autenticarlos en el clúster. Después de que los nodos híbridos se unan al clúster, aparecerán con el estado `Not Ready` en la consola de Amazon EKS y en herramientas compatibles con Kubernetes, como `kubectl`. Tras completar los pasos que se indican en esta página, proceda a the section called “Cómo configurar una CNI” para preparar los nodos híbridos para ejecutar aplicaciones.

Requisitos previos

Antes de conectar los nodos híbridos al clúster de Amazon EKS, compruebe que se cumplen los requisitos previos indicados.

- Dispone de conectividad de red entre el entorno en las instalaciones con la región de AWS que aloja el clúster de Amazon EKS. Para obtener más información, consulte the section called “Preparación de las redes”.
- Ha creado el rol de IAM de los nodos híbridos y ha configurado el proveedor de credenciales en las instalaciones (activaciones híbridas de AWS Systems Manager o AWS IAM Roles Anywhere). Para obtener más información, consulte the section called “Cómo preparar las credenciales”.
- Ha creado el clúster de Amazon EKS habilitado para nodos híbridos. Para obtener más información, consulte the section called “Creación de un clúster”.

- Ha asociado el rol de IAM de nodos híbridos a los permisos de control de acceso basado en roles (RBAC) de Kubernetes. Para obtener más información, consulte the section called “Cómo preparar el acceso al clúster”.

Paso 1: creación del archivo TOML de configuración de Bottlerocket

Para configurar Bottlerocket para nodos híbridos, debe crear un archivo `settings.toml` con la configuración necesaria. El contenido del archivo TOML variará según el proveedor de credenciales que utilice (SSM o IAM Roles Anywhere). Este archivo se transferirá como datos de usuario al aprovisionar la instancia de Bottlerocket.

> **Note**
>
> Los archivos TOML que se proporcionan a continuación solo representan la configuración mínima necesaria para inicializar una máquina VMWare de Bottlerocket como un nodo en un clúster de EKS. Bottlerocket ofrece una amplia gama de opciones de configuración para abordar varios casos de uso diferentes, por lo que, para obtener más opciones de configuración además de la inicialización del nodo híbrido, consulte la documentación de Bottlerocket a fin de obtener una lista completa de todas las configuraciones documentadas para la versión de Bottlerocket que esté utilizando (por ejemplo, aquí están todas las configuraciones disponibles para Bottlerocket 1.51.x).

SSM

Si utiliza AWS Systems Manager como proveedor de credenciales, cree un archivo `settings.toml` con el siguiente contenido:

```
[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
```

Reemplace los marcadores de posición con los siguientes valores:

- `<cluster-name>`: el nombre del clúster de Amazon EKS.
- `<api-server-endpoint>`: el punto de conexión del servidor API del clúster.
- `<cluster-certificate-authority>`: el paquete CA codificado en base64 del clúster.
- `<region>`: la región de AWS donde se aloja el clúster, por ejemplo, “us-east-1”.
- `<hostname>`: el nombre de host de la instancia de Bottlerocket, que se utilizará también como nombre del nodo. Puede ser cualquier valor único de su elección, pero debe cumplir con las convenciones de nomenclatura de objetos de Kubernetes. Además, el nombre de host que utilice no puede tener más de 64 caracteres. NOTA: Al utilizar el proveedor SSM, este nombre de host y

nombre de nodo serán reemplazados por el ID de instancia administrada (por ejemplo, ID `mi-*`) después de que la instancia se registre en SSM.

- `<base64-encoded-admin-container-userdata>`: el contenido codificado en base64 de la configuración del contenedor de administración de Bottlerocket. Al habilitar el contenedor de administración, se puede conectar a la instancia de Bottlerocket mediante SSH para explorar y depurar el sistema. Aunque esta configuración no es obligatoria, recomendamos habilitarla para facilitar la resolución de problemas. Consulte la documentación del contenedor de administración de Bottlerocket para obtener más información sobre cómo autenticarse con el contenedor de administración. El contenedor de administración acepta entradas de usuario y clave SSH en formato JSON, por ejemplo,

```
{
  "user": "<ssh-user>",
  "ssh": {
    "authorized-keys": [
      "<ssh-authorized-key>"
    ]
  }
}
```

- `<base64-encoded-bootstrap-container-userdata>`: el contenido codificado en base64 de la configuración del contenedor de arranque Bottlerocket. Consulte la documentación del contenedor de arranque de Bottlerocket para obtener más información sobre su configuración. El contenedor de arranque se encarga de registrar la instancia como una instancia administrada de AWS SSM y de unirla como nodo de Kubernetes en el clúster de Amazon EKS. Los datos de usuario que se transmiten al contenedor de arranque adoptan la forma de una invocación de comando que acepta como entrada el código de activación híbrida SSM y el ID que creó previamente:

```
eks-hybrid-ssm-setup --activation-id=<activation-id> --activation-code=<activation-
code> --region=<region>
```

IAM Roles Anywhere

Si utiliza AWS IAM Roles Anywhere como proveedor de credenciales, cree un archivo `settings.toml` con el siguiente contenido:

```
[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>"
```

Reemplace los marcadores de posición con los siguientes valores:

- <cluster-name>: el nombre del clúster de Amazon EKS.
- <api-server-endpoint>: el punto de conexión del servidor API del clúster.
- <cluster-certificate-authority>: el paquete CA codificado en base64 del clúster.
- <region>: la región de AWS que aloja el clúster (por ejemplo, “us-east-1”)
- <hostname>: el nombre de host de la instancia de Bottlerocket, que se utilizará también como nombre del nodo. Puede ser cualquier valor único de su elección, pero debe cumplir con las convenciones de nomenclatura de objetos de Kubernetes. Además, el nombre de host que utilice no puede tener más de 64 caracteres. NOTA: Cuando se utiliza el proveedor IAM-RA, el nombre del nodo debe coincidir con el CN del certificado en el host si ha configurado la política de confianza del rol de IAM de Nodos híbridos con la condición de recurso "sts:RoleSessionName": "${aws:PrincipalTag/x509Subject/CN}".
- AWS: el contenido codificado en base64 del archivo de configuración de <base64-encoded-aws-config-file>. El contenido del archivo debe ser el siguiente:

```
[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>: el contenido codificado en base64 de la configuración del contenedor de administración de Bottlerocket. Al habilitar el contenedor de administración, se puede conectar a la instancia de Bottlerocket mediante SSH para explorar y depurar el sistema. Aunque esta configuración no es obligatoria, recomendamos habilitarla para facilitar la resolución de problemas. Consulte la documentación del contenedor de administración de Bottlerocket para obtener más información sobre cómo autenticarse con el contenedor de administración. El contenedor de administración acepta entradas de usuario y clave SSH en formato JSON, por ejemplo,

```
{
  "user": "<ssh-user>",
  "ssh": {
    "authorized-keys": [
      "<ssh-authorized-key>"
    ]
  }
```

```
}
```

- `<base64-encoded-bootstrap-container-userdata>`: el contenido codificado en base64 de la configuración del contenedor de arranque Bottlerocket. Consulte la documentación del contenedor de arranque de Bottlerocket para obtener más información sobre su configuración. El contenedor de arranque se encarga de crear el certificado de host de IAM Roles Anywhere y los archivos de clave privada del certificado en la instancia. Estas serán consumidas por el `aws_signing_helper` para obtener credenciales temporales para autenticarse con el clúster de Amazon EKS. Los datos de usuario que se transmiten al contenedor de arranque adoptan la forma de una invocación de comando que acepta como entrada el contenido del certificado y la clave privada que creó previamente:

```
eks-hybrid-iam-ra-setup --certificate=<certificate> --key=<private-key>
```

Paso 2: aprovisionamiento de la máquina virtual Bottlerocket vSphere con datos de usuario

Una vez que haya creado el archivo TOML, transmítalo como datos de usuario durante la creación de la máquina virtual en vSphere. Tenga en cuenta que los datos de usuario se deben configurar antes de que la máquina virtual se encienda por primera vez. Por lo tanto, deberá proporcionarlos al crear la instancia o, si desea crear la máquina virtual con anticipación, esta debe permanecer en estado apagado (poweredOff) hasta que configure los datos de usuario. Por ejemplo, si utiliza la CLI de `govc`:

Cómo crear la máquina virtual por primera vez

```
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>
```

Cómo actualizar los datos de usuario de una máquina virtual existente

```
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>
```

En las secciones anteriores, la opción `-e guestinfo.userdata.encoding="base64"` especifica que los datos de usuario están codificados en base64. La opción `-e guestinfo.userdata` transmite el contenido codificado en base64 del archivo `settings.toml` como datos de usuario a la instancia de Bottlerocket. Reemplace los marcadores de posición con los valores específicos, como la plantilla OVA de Bottlerocket y los detalles de la red.

Paso 3: verificación de la conexión del nodo híbrido

Después de que la instancia de Bottlerocket inicie, intentará unirse al clúster de Amazon EKS. Para verificar la conexión, ingrese a la consola de Amazon EKS y diríjase a la pestaña Computación del clúster o ejecute el siguiente comando:

```
kubectl get nodes
```

> **Important**
>
> Los nodos tendrán el estado `Not Ready`, que es el esperado, y se debe a la falta de una CNI que se ejecute en los nodos híbridos. Si los nodos no se unieron al clúster, consulte the section called “Solución de problemas”.

Paso 4: Configuración de una CNI para los nodos híbridos

Para que los nodos híbridos estén preparados para ejecutar aplicaciones, continúe con los pasos que se indican en the section called “Cómo configurar una CNI”.

Actualización de nodos híbridos para el clúster

Las instrucciones para actualizar los nodos híbridos son similares a aquellas de los nodos autoadministrados de Amazon EKS que se ejecutan en Amazon EC2. Le recomendamos crear nuevos nodos híbridos en la versión de Kubernetes de destino, migrar correctamente las aplicaciones existentes a los nodos híbridos de la nueva versión de Kubernetes y eliminar los nodos híbridos de la versión antigua de Kubernetes del clúster. Asegúrese de revisar las Prácticas recomendadas de Amazon EKS para actualizaciones antes de iniciar una actualización. Los Nodos híbridos de Amazon EKS cuentan con el mismo soporte para versiones de Kubernetes que los clústeres de Amazon EKS con nodos en la nube, incluidos el soporte estándar y ampliado.

Los Nodos híbridos de Amazon EKS siguen la misma política de discrepancia de versiones para los nodos que el proyecto fuente de Kubernetes. Los Nodos híbridos de Amazon EKS no pueden estar en una versión más reciente que el plano de control de Amazon EKS, y los nodos híbridos pueden tener hasta tres versiones menores de Kubernetes más antiguas que la versión menor del plano de control de Amazon EKS.

Si no dispone de capacidad de sobra para crear nuevos nodos híbridos en la versión de Kubernetes de destino para una estrategia de actualización de migración de transición, también puede utilizar la CLI de Nodos híbridos de Amazon EKS (`nodeadm`) para actualizar la versión de Kubernetes de los nodos híbridos in situ.

> **Important**
>
> Si actualiza los nodos híbridos in situ con `nodeadm`, se produce un tiempo de inactividad del nodo durante el proceso en el que se apaga la versión anterior de los componentes de Kubernetes y se instalan e inician los componentes de la nueva versión de Kubernetes.

Requisitos previos

Antes de efectuar la actualización, asegúrese de que cumple los siguientes requisitos previos.

- La versión de Kubernetes de destino para la actualización de los nodos híbridos debe ser igual o inferior a la versión del plano de control de Amazon EKS.
- Si sigue una estrategia de actualización de migración de transición, los nuevos nodos híbridos que instale en la versión de Kubernetes de destino deben cumplir los requisitos the section called “Requisitos previos”. Esto implica contar con direcciones IP dentro del CIDR de red de nodos remotos que transmitió durante la creación del clúster de Amazon EKS.

- Tanto para la migración de transición como para las actualizaciones en las instalaciones, los nodos híbridos deben tener acceso a los dominios necesarios para obtener las nuevas versiones de las dependencias de los nodos híbridos.
- Debe tener kubectl instalado en la máquina local o en la instancia que utilice para interactuar con el punto de conexión de la API de Kubernetes de Amazon EKS.
- La versión de la CNI debe ser compatible con la versión de Kubernetes a la que se actualiza. En caso contrario, actualice la versión de la CNI antes de actualizar los nodos híbridos. Para obtener más información, consulte the section called “Cómo configurar una CNI”.

Actualizaciones de migración por transición (azul/verde)

Las actualizaciones de migración por transición se refieren al proceso de crear nuevos nodos híbridos en hosts nuevos con la versión de destino de Kubernetes, migrar de forma controlada las aplicaciones existentes a los nuevos nodos híbridos con la versión de destino de Kubernetes y eliminar del clúster los nodos híbridos con la versión anterior de Kubernetes. Esta estrategia también se conoce como una migración azul/verde.

1. Siga los siguientes pasos the section called “Cómo conectar nodos híbridos” para conectar los nuevos hosts como nodos híbridos. Al ejecutar el comando `nodeadm install`, utilice la versión de Kubernetes de destino.
2. Habilite la comunicación entre los nuevos nodos híbridos en la versión de Kubernetes de destino y los nodos híbridos en la versión antigua de Kubernetes. Esta configuración permite que los pods se comuniquen entre sí mientras la carga de trabajo se migra a los nodos híbridos de la versión de Kubernetes de destino.
3. Confirme que los nodos híbridos de la versión de Kubernetes de destino se han unido correctamente al clúster y que se encuentran en estado Preparado.
4. Utilice el siguiente comando para marcar como no programables cada uno de los nodos que desea eliminar. Esto es para que los nuevos pods no se programen ni se vuelvan a programar en los nodos que se van a reemplazar. Para obtener más información, consulte kubectl cordon en la documentación de Kubernetes. Sustituya `NODE_NAME` por el nombre de los nodos híbridos en la versión anterior de Kubernetes.

```
kubectl cordon NODE_NAME
```

Puede identificar y marcar como no programables todos los nodos de una versión específica de Kubernetes (en este caso, la `1.28`) con el siguiente fragmento de código.

```
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. Si la implementación actual ejecuta menos de dos réplicas de CoreDNS en los nodos híbridos, escale horizontalmente la implementación a al menos dos réplicas. Le recomendamos ejecutar al menos dos réplicas de CoreDNS en nodos híbridos para garantizar la resistencia durante las operaciones normales.

```
kubectl scale deployments/coredns --replicas=2 -n kube-system
```

6. Vacíe cada uno de los nodos híbridos de la versión antigua de Kubernetes que desee eliminar del clúster con el siguiente comando. Para obtener más información sobre el vaciado de nodos, consulte Vaciado seguro de nodos en la documentación de Kubernetes. Sustituya `NODE_NAME` por el nombre de los nodos híbridos en la versión anterior de Kubernetes.

```
kubectl drain NODE_NAME --ignore-daemonsets --delete-emptydir-data
```

Puede identificar todos los nodos de una versión concreta de Kubernetes (en este caso, `1.28`) y vaciarlos con el siguiente fragmento de código.

```
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. Puede usar `nodeadm` para detener y eliminar los artefactos de los nodos híbridos del host. Debe ejecutar `nodeadm` con un usuario que tenga privilegios raíz/sudo. De forma predeterminada,

`nodeadm uninstall` no procederá si quedan pods en el nodo. Para obtener más información, consulte the section called “Nodeadm de nodos híbridos”.

```
nodeadm uninstall
```

8. Una vez que los artefactos de los nodos híbridos estén detenidos y desinstalados, elimine el recurso de nodo del clúster.

```
kubectl delete node node-name
```

   Puede identificar todos los nodos de una versión concreta de Kubernetes (en este caso, `1.28`) y eliminarlos con el siguiente fragmento de código.

```
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. Según la CNI que elija, es posible que, aún después de ejecutar los pasos anteriores, queden artefactos en los nodos híbridos. Para obtener más información, consulte the section called “Cómo configurar una CNI”.

Actualizaciones locales

El proceso de actualización in situ consiste en utilizar `nodeadm upgrade` para actualizar la versión de Kubernetes para nodos híbridos sin utilizar nuevos hosts físicos o virtuales ni una estrategia de migración de transición. El proceso `nodeadm upgrade` apaga los componentes de Kubernetes antiguos existentes que se ejecutan en el nodo híbrido, desinstala los componentes de Kubernetes antiguos existentes, instala los nuevos componentes de Kubernetes de destino e inicia los nuevos componentes de Kubernetes de destino. Se recomienda enfáticamente actualizar un nodo a la vez para minimizar el impacto en las aplicaciones que se ejecutan en los nodos híbridos. La duración de este proceso depende del ancho de banda y la latencia de la red.

1. Utilice el siguiente comando para marcar como no programable el nodo que está en proceso de actualizar. Esto es para que los nuevos pods no se programen o reprogramen en el nodo que se actualiza. Para obtener más información, consulte kubectl cordon en la documentación de Kubernetes. Sustituya `NODE_NAME` por el nombre del nodo híbrido que se actualiza

```
kubectl cordon NODE_NAME
```

2. Vacíe el nodo que se actualiza con el siguiente comando. Para obtener más información sobre el vaciado de nodos, consulte Vaciado seguro de nodos en la documentación de Kubernetes. Sustituya `NODE_NAME` por el nombre del nodo híbrido que se actualiza.

```
kubectl drain NODE_NAME --ignore-daemonsets --delete-emptydir-data
```

3. Ejecute `nodeadm upgrade` en el nodo híbrido que se actualiza. Debe ejecutar `nodeadm` con un usuario que tenga privilegios raíz/sudo. El nombre del nodo se conserva durante la actualización para los proveedores de credenciales de AWS SSM y AWS IAM Roles Anywhere. No es posible cambiar de proveedor de credenciales durante el proceso de actualización. Consulte the section called “Nodeadm de nodos híbridos” para conocer los valores de configuración de `nodeConfig.yaml`. Sustituya `K8S_VERSION` por la versión de Kubernetes de destino a la que se actualiza.

```
nodeadm upgrade K8S_VERSION -c file://nodeConfig.yaml
```

4. Para permitir que los pods se programen en el nodo después de la actualización, escriba lo siguiente. Sustituya `NODE_NAME` por el nombre del nodo.

```
kubectl uncordon NODE_NAME
```

5. Observe el estado de los nodos híbridos y espere a que se apaguen y reinicien en la nueva versión de Kubernetes con el estado Preparado.

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

Aplicación de revisiones de actualizaciones de seguridad para nodos híbridos

Este tema describe el procedimiento para aplicar revisiones de actualizaciones de seguridad en el lugar para paquetes y dependencias específicos que se ejecutan en los nodos híbridos. Como buena

práctica, recomendamos actualizar periódicamente los nodos híbridos para recibir actualizaciones relacionadas con vulnerabilidades (CVE) y revisiones de seguridad.

Para conocer los pasos para actualizar la versión de Kubernetes, consulte the section called "Actualización de nodos híbridos".

Un ejemplo de software que podría requerir revisiones de seguridad es `containerd`.

`Containerd`

`containerd` es el tiempo de ejecución de contenedores estándar de Kubernetes y una dependencia fundamental para los nodos híbridos de EKS. Se encarga de administrar el ciclo de vida de los contenedores, incluida la extracción de imágenes y la ejecución de los contenedores. En un nodo híbrido, puede instalar `containerd` mediante la CLI de nodeadm o de forma manual. Según el sistema operativo del nodo, `nodeadm` instalará `containerd` desde el paquete distribuido por el sistema operativo o desde el paquete de Docker.

Si se publica un CVE en `containerd`, tendrá las siguientes opciones para actualizar a la versión revisada de `containerd` en los nodos híbridos.

Paso 1: Cómo comprobar si la revisión se ha publicado en los administradores de paquetes

Para verificar si la revisión de la CVE de `containerd` ha sido publicada en el administrador de paquetes de cada sistema operativo, puede consultar los boletines de seguridad correspondientes:

- Amazon Linux 2023
- RHEL
- Ubuntu 20.04
- Ubuntu 22.04
- Ubuntu 24.04

Si utiliza el repositorio de Docker como origen de `containerd`, puede consultar los Anuncios de seguridad de Docker para verificar la disponibilidad de la versión revisada en el repositorio de Docker.

Paso 2: Cómo elegir el método para instalar la revisión

Existen tres métodos para aplicar revisiones e instalar actualizaciones de seguridad en el lugar en los nodos. El método que puede utilizar depende de si la revisión está disponible en el administrador de paquetes del sistema operativo.

1. Instale revisiones con las `nodeadm upgrade` publicadas en los administradores de paquetes. Consulte el Paso 2.a.
2. Instale revisiones directamente con los administradores de paquetes. Consulte el Paso 2.b.
3. Instale revisiones personalizadas que no estén publicadas en los administradores de paquetes. Tenga en cuenta que existen consideraciones especiales para las revisiones personalizadas de `containerd`. Consulte el Paso 2.c.

Paso 2.a: Aplicación de revisiones con **`nodeadm upgrade`**

Una vez que haya confirmado que la revisión de la CVE de `containerd` ha sido publicada en los repositorios del sistema operativo o de Docker (ya sea Apt o RPM), puede utilizar el comando `nodeadm upgrade` para actualizar a la última versión de `containerd`. Como no se trata de una actualización de versión de Kubernetes, debe proporcionar la versión actual de Kubernetes al comando de actualización `nodeadm`.

```
nodeadm upgrade K8S_VERSION --config-source file:///root/nodeConfig.yaml
```

Paso 2.b: Aplicación de revisiones con los administradores de paquetes del sistema operativo

Como alternativa, también puede actualizar a través del administrador de paquetes correspondiente y usarlo para actualizar el paquete de `containerd` de la siguiente manera.

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
```

Paso 2.c: Revisión de CVE de **`Containerd`** no publicada en los administradores de paquetes

Si la versión revisada de `containerd` solo está disponible por otros medios, en lugar de en el administrador de paquetes; por ejemplo, en las publicaciones de GitHub, puede instalar `containerd` desde el sitio oficial de GitHub.

1. Si la máquina ya se ha unido al clúster como un nodo híbrido, entonces debe ejecutar el comando `nodeadm uninstall`.
2. Instale los binarios oficiales de `containerd`. Puede seguir los pasos de instalación oficiales disponibles en GitHub.
3. Ejecute el comando `nodeadm install` con el argumento `--containerd-source` establecido en `none`, lo que omitirá la instalación de `containerd` a través de `nodeadm`. Puede usar el valor de `none` en el origen de `containerd` para cualquier sistema operativo que el nodo ejecute.

   ```
   nodeadm install K8S_VERSION --credential-provider CREDS_PROVIDER --containerd-source
    none
   ```

Cómo eliminar nodos híbridos

En este tema se explica cómo eliminar nodos híbridos del clúster de Amazon EKS. Debe eliminar los nodos híbridos con las herramientas compatibles con Kubernetes que elija, como kubectl. El cobro por los nodos híbridos se detiene cuando el objeto del nodo se elimina del clúster de Amazon EKS. Para obtener más información sobre los precios de los nodos híbridos, consulte los Precios de Amazon EKS.

> **Important**
>
> La eliminación de nodos interrumpe las cargas de trabajo en ejecución en el nodo. Antes de eliminar los nodos híbridos, le recomendamos vaciar primero el nodo para trasladar los

pods a otro nodo activo. Para obtener más información sobre el vaciado de nodos, consulte Vaciado seguro de nodos en la documentación de Kubernetes.

Ejecute los pasos de kubectl que se indican a continuación desde la máquina o instancia local que utilice para interactuar con el punto de conexión de la API de Kubernetes del clúster de Amazon EKS. Si utiliza un archivo `kubeconfig` específico, utilice la marca `--kubeconfig`.

Paso 1: Enumeración de los nodos

```
kubectl get nodes
```

Paso 2: Vaciado del nodo

Consulte Vaciado de kubectl en la documentación de Kubernetes para obtener más información sobre el comando `kubectl drain`.

```
kubectl drain --ignore-daemonsets <node-name>
```

Paso 3: Detención y desinstalación de artefactos de nodos híbridos

Puede usar la CLI (`nodeadm`) de los Nodos híbridos de Amazon EKS para detener y eliminar los artefactos de nodos híbridos del host. Debe ejecutar `nodeadm` con un usuario que tenga privilegios raíz/sudo. De forma predeterminada, `nodeadm uninstall` no procederá si quedan pods en el nodo. Si utiliza AWS Systems Manager (SSM) como proveedor de credenciales, el comando `nodeadm uninstall` anula el registro del host como instancia administrada por AWS SSM. Para obtener más información, consulte the section called “Nodeadm de nodos híbridos”.

```
nodeadm uninstall
```

Paso 4: Cómo eliminar el nodo del clúster

Una vez que los artefactos de los nodos híbridos estén detenidos y desinstalados, elimine el recurso de nodo del clúster.

```
kubectl delete node <node-name>
```

Paso 5: Cómo comprobar si hay artefactos restantes

Según la CNI que elija, es posible que, aún después de ejecutar los pasos anteriores, queden artefactos en los nodos híbridos. Para obtener más información, consulte the section called “Cómo configurar una CNI”.

Configuración de redes de aplicaciones, complementos y webhooks para nodos híbridos

Después de crear un clúster de EKS para nodos híbridos, configure capacidades adicionales para las redes de aplicaciones (CNI, BGP, entrada, equilibrio de carga, políticas de red), complementos, webhooks y configuraciones de proxy. Para ver la lista completa de los complementos de EKS y de la comunidad que son compatibles con los nodos híbridos, consulte the section called “Configuración de complementos”.

Información del clúster de EKS: EKS incluye comprobaciones de información para detectar errores en la configuración de los nodos híbridos que puedan afectar a la funcionalidad del clúster o de las cargas de trabajo. Para obtener más información sobre información del clúster, consulte the section called “Información sobre clústeres”.

A continuación, se enumeran las capacidades y complementos comunes que puede utilizar con los nodos híbridos:

- Interfaz de red de contenedores (CNI): AWS admite Cilium como CNI para nodos híbridos. Para obtener más información, consulte the section called “Cómo configurar una CNI”. Tenga en cuenta que la CNI de AWS VPC no se puede utilizar con nodos híbridos.
- CoreDNS y **`kube-proxy`**: CoreDNS y `kube-proxy` se instalan automáticamente cuando los nodos híbridos se unen al clúster de EKS. Estos complementos se pueden administrar como complementos de EKS tras la creación del clúster.
- Entrada y equilibrio de carga: puede utilizar el Controlador del equilibrador de carga de AWS y el Equilibrador de carga de aplicación (ALB) o el Equilibrador de carga de red (NLB) con el tipo de destino `ip` para las cargas de trabajo que se ejecutan en nodos híbridos. AWS es compatible con las características integradas de equilibrio de carga de entrada, puerta de enlace y el servicio de Kubernetes de Cilium para las cargas de trabajo que se ejecutan en nodos híbridos. Para obtener más información, consulte the section called “Configuración de Ingress” y the section called “Configuración de servicios de LoadBalancer”.
- Métricas: puede utilizar raspadores sin agente de Amazon Managed Service para Prometheus (AMP), AWS Distro para Open Telemetry (ADOT) y el agente de observabilidad de Amazon

CloudWatch con nodos híbridos. Para utilizar los raspadores sin agente de AMP para métricas de pods en nodos híbridos, los pods deben ser accesibles desde la VPC que se utiliza para el clúster de EKS.

- Registros: puede habilitar el registro del plano de control de EKS para clústeres habilitados para nodos híbridos. Puede utilizar el complemento ADOT de EKS y el complemento de agente de observabilidad de Amazon CloudWatch para EKS para el registro de nodos híbridos y pods.
- Pod Identities y los IRSA: puede usar Pod Identities de EKS y roles de IAM para cuentas de servicio (IRSA) con aplicaciones que se ejecutan en nodos híbridos para permitir un acceso detallado para los pods que se ejecutan en nodos híbridos con otros servicios de AWS.
- Webhooks: si ejecuta webhooks, consulte the section called “Configuración de webhooks” para conocer las consideraciones y los pasos para ejecutarlos opcionalmente en nodos en la nube, en caso de que no sea posible enrutar las redes de pods en las instalaciones.
- Proxy: si utiliza un servidor proxy en el entorno en las instalaciones para el tráfico que sale del centro de datos o del entorno periférico, puede configurar los nodos híbridos y el clúster de modo que utilicen el servidor proxy. Para obtener más información, consulte the section called “Configuración del proxy”.

Temas

- Configuración de una CNI para nodos híbridos
- Cómo configurar complementos para nodos híbridos
- Configuración de webhooks para nodos híbridos
- Configuración del proxy para nodos híbridos
- Configuración del BGP de Cilium para nodos híbridos
- Configuración de Kubernetes Ingress para nodos híbridos
- Configuración de servicios del tipo LoadBalancer para nodos híbridos
- Configuración de políticas de red de Kubernetes para nodos híbridos

Configuración de una CNI para nodos híbridos

Cilium es la interfaz de red de contenedores (CNI) compatible con AWS para los Nodos híbridos de Amazon EKS. Debe instalar una CNI para que los nodos híbridos estén listos para atender cargas de trabajo. Los nodos híbridos aparecen con el estado `Not Ready` hasta que una CNI está en ejecución. Puede administrar la CNI con las herramientas que elija, como Helm. Las instrucciones de esta página tratan la administración del ciclo de vida de Cilium (instalar, actualizar, eliminar).

Consulte the section called “Información general sobre Cilium Ingress y Cilium Gateway”, the section called “LoadBalancer del tipo de servicio” y the section called “Configuración de políticas de red” para saber cómo configurar Cilium para las políticas de entrada, equilibrio de carga y red.

Cilium no es compatibles con AWS cuando se ejecuta en nodos en la nube de AWS. La CNI de Amazon VPC no es compatible con los nodos híbridos y está configurada con antiafinidad para la etiqueta `eks.amazonaws.com/compute-type: hybrid`.

La documentación de Calico que aparecía anteriormente en esta página se ha trasladado al repositorio de ejemplos de Nodos híbridos de EKS.

Compatibilidad de versiones

Las versiones `v1.17.x` y `v1.18.x` de Cilium son compatibles con los Nodos híbridos de EKS para todas las versiones de Kubernetes compatibles con Amazon EKS.

> **Note**
>
> Requisito del kernel de Cilium v1.18.3: debido a los requisitos del kernel (kernel de Linux >= 5.10), Cilium v1.18.3 no es compatible con:

- Ubuntu 20.04
- Red Hat Enterprise Linux (RHEL) 8

Para conocer los requisitos del sistema, consulte los requisitos del sistema de Cilium.

Consulte la compatibilidad de versiones de Kubernetes para ver las versiones que admite Amazon EKS. Los Nodos híbridos de EKS tienen la misma compatibilidad de versiones de Kubernetes que los clústeres de Amazon EKS con nodos en la nube.

Capacidades compatibles

AWS mantiene versiones de Cilium para los Nodos híbridos de EKS que se basan en el proyecto de código abierto Cilium. Para recibir soporte de AWS para Cilium, debe utilizar las versiones de Cilium mantenidas por AWS y las versiones compatibles de Cilium.

AWS ofrece soporte técnico para las configuraciones predeterminadas de las siguientes capacidades de Cilium para su uso con Nodos híbridos de EKS. Si planea utilizar la funcionalidad fuera del ámbito

del soporte de AWS, es recomendable obtener soporte comercial alternativo para Cilium o contar con expertos internos que puedan resolver problemas y contribuir con correcciones al proyecto de Cilium.

Característica Cilium	Compatible con AWS
Conformidad de la red Kubernetes	Sí
Conectividad del clúster principal	Sí
Familia de IP	IPv4
Administración del ciclo de vida	Helm
Modos de redes	Encapsulación mediante VXLAN
Administración de direcciones IP (IPAM)	Ámbito del clúster de IPAM de Cilium
Política de red	Política de red de Kubernetes
Protocolo de puerta de enlace fronteriza (BGP)	Cilium BGP Control Plane
Kubernetes Ingress	Cilium Ingress, Cilium Gateway
Asignación de IP de LoadBalancer de servicio	Cilium Load Balancer IPAM
Anuncio de dirección IP de LoadBalancer de servicio	Cilium BGP Control Plane
Reemplazo de kube-proxy	Sí

Consideraciones sobre Cilium

- Repositorio de Helm: AWS aloja el gráfico de Helm de Cilium en Amazon Elastic Container Registry Público (Amazon ECR Público) en Amazon EKS Cilium/Cilium. Las versiones disponibles incluyen lo siguiente:
  - Cilium v1.17.9: `oci://public.ecr.aws/eks/cilium/cilium:1.17.9-0`

- Cilium v1.18.3: `oci://public.ecr.aws/eks/cilium/cilium:1.18.3-0`

  Los comandos de este tema utilizan este repositorio. Tenga en cuenta que ciertos comandos `helm repo` no son válidos para los repositorios de Helm en Amazon ECR Público, por lo que no puede hacer referencia a este repositorio desde un nombre de repositorio de Helm local. En su lugar, use el URI completo en la mayoría de los comandos.

- De forma predeterminada, Cilium está configurado para ejecutarse en modo de superposición/túnel con VXLAN como método de encapsulación. Este modo es el que impone menos requisitos a la red física subyacente.
- De forma predeterminada, Cilium enmascara la dirección IP de origen de todo el tráfico de pods que sale del clúster, asignándole la dirección IP del nodo. Si desactiva el enmascarado, los CIDR de los pods deben ser enrutables en la red en las instalaciones.
- Si ejecuta webhooks en nodos híbridos, los CIDR de los pods deben ser enrutable en la red en las instalaciones. Si los CIDR de los pods no son enrutables en la red en las instalaciones, se recomienda ejecutar los webhooks en los nodos en la nube del mismo clúster. Para obtener más información, consulte the section called “Configuración de webhooks” y the section called “Preparación de las redes”.
- AWS recomienda usar la funcionalidad del BGP integrada de Cilium para hacer que los CIDR de los pods sean enrutables en la red en las instalaciones. Para obtener más información sobre cómo configurar el BGP de Cilium con nodos híbridos, consulte the section called “Configuración de BGP”.
- La administración de direcciones IP (IPAM) predeterminada en Cilium se llama Ámbito del clúster, donde el operador de Cilium asigna direcciones IP para cada nodo en función de los CIDR de pods configurados por el usuario.

Cómo instalar Cilium en nodos híbridos

Procedimiento

1. Cree un archivo YAML denominado `cilium-values.yaml`. En el siguiente ejemplo se configura Cilium para que se ejecute únicamente en nodos híbridos. Para ello, establece afinidad para la etiqueta `eks.amazonaws.com/compute-type: hybrid` para el operador y el agente de Cilium.
   - Configure `clusterPoolIpv4PodCIDRList` con los mismos CIDR de los pods que configuró para las redes de pods remotas del clúster de EKS. Por ejemplo, `10.100.0.0/24`. El operador de Cilium asigna segmentos de direcciones IP desde el espacio de IP

`clusterPoolIpv4PodCIDRList` configurado. El CIDR de los pods no debe superponerse con el CIDR de los nodos en las instalaciones, el CIDR de las VPC o el CIDR de los servicios de Kubernetes.

- Configure `clusterPoolIpv4MaskSize` según los pods requeridos por nodo. Por ejemplo, `25` para un tamaño de segmento /25 de 128 pods por nodo.
- No cambie `clusterPoolIpv4PodCIDRList` ni `clusterPoolIpv4MaskSize` después de implementar Cilium en su clúster. Consulte Expanding the cluster pool para obtener más información.
- Si está ejecutando Cilium en el modo de reemplazo de kube-proxy, configure `kubeProxyReplacement: "true"` en sus valores de Helm y asegúrese de que no tenga ninguna implementación de kube-proxy existente en ejecución en los mismos nodos que Cilium.
- En el siguiente ejemplo se desactiva el proxy de capa 7 (L7) de Envoy que Cilium utiliza para las políticas de red y el ingreso de la capa 7. Para obtener más información, consulte the section called “Configuración de políticas de red” y the section called “Información general sobre Cilium Ingress y Cilium Gateway”.
- En el siguiente ejemplo se configura `loadBalancer.serviceTopology: true` para que la distribución del tráfico del servicio funcione correctamente si la configura para sus servicios. Para obtener más información, consulte the section called “Configurar la distribución de tráfico del servicio”.
- Para obtener una lista completa de los valores de Helm para Cilium, consulte la referencia de Helm en la documentación de Cilium.

```
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. Instale Cilium en el clúster.
   - Sustituya `CILIUM_VERSION` por una versión de Cilium (por ejemplo, `1.17.9-0` o `1.18.3-0`). Se recomienda utilizar la versión más reciente del parche para la versión secundaria de Cilium.
   - Asegúrese de que los nodos cumplan los requisitos del kernel para la versión que elija. Cilium v1.18.3 requiere un kernel de Linux >= 5.10.
   - Si va a utilizar un archivo kubeconfig específico, utilice la marca `--kubeconfig` con el comando de instalación de Helm.

```
helm install cilium oci://public.ecr.aws/eks/cilium/cilium \
    --version CILIUM_VERSION \
    --namespace kube-system \
    --values cilium-values.yaml
```

3. Confirme que la instalación de Cilium se haya realizado correctamente con los siguientes comandos. Debería ver la implementación del `cilium-operator` y el `cilium-agent` en ejecución en cada uno de los nodos híbridos. Además, los nodos híbridos se deben encontrar en el estado `Ready`. Para obtener información sobre cómo configurar el BGP de Cilium para anunciar los CIDR de los pods en la red en las instalaciones, continúe con the section called “Configuración de BGP”.

```
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
```

Cómo actualizar Cilium en nodos híbridos

Antes de actualizar la implementación de Cilium, revise detenidamente la documentación de actualización de Cilium y las notas de actualización para comprender los cambios en la versión de destino de Cilium.

1. Asegúrese de haber instalado `helm` CLI en el entorno de línea de comandos. Consulte la documentación de Helm para consultar las instrucciones de instalación.
2. Ejecute la comprobación previa a la actualización de Cilium. Sustituya `CILIUM_VERSION` por la versión de destino de Cilium. Le recomendamos ejecutar la versión más reciente del parche para la versión secundaria de Cilium. Puede encontrar la versión más reciente de la revisión de una versión secundaria determinada de Cilium en la sección Versiones estables de la documentación de Cilium.

```
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. Después de aplicar `cilium-preflight.yaml`, asegúrese de que la cantidad de pods `READY` sea la misma que la cantidad de pods de Cilium en ejecución.

```
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. Una vez que la cantidad de pods en READY sea igual, asegúrese de que la implementación de verificación previa de Cilium también esté marcada como READY 1/1. Si aparece READY 0/1, consulte la sección Validación del CNP y resuelva los problemas relacionados con la implementación antes de continuar con la actualización.

```
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. Elimine la verificación previa

```
helm uninstall cilium-preflight --namespace kube-system
```

6. Antes de ejecutar el comando `helm upgrade`, conserve los valores de la implementación en `existing-cilium-values.yaml` o utilice las opciones de línea de comandos de `--set` para la configuración al ejecutar el comando de actualización. La operación de actualización sobrescribe el ConfigMap de Cilium, por lo que es fundamental que se transmitan los valores de configuración al realizar la actualización.

```
helm get values cilium --namespace kube-system -o yaml > existing-cilium-values.yaml
```

7. Durante las operaciones normales del clúster, todos los componentes de Cilium deben ejecutar la misma versión. En los siguientes pasos se describe cómo actualizar todos los componentes de una versión estable a una versión estable posterior. Al actualizar de una versión secundaria a otra secundaria, se recomienda actualizar primero a la versión de revisión más reciente para la versión secundaria de Cilium existente. Para minimizar las interrupciones, la opción `upgradeCompatibility` se debe establecer en la versión inicial de Cilium que instaló en este clúster.

```
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. (Opcional) Si necesita revertir la actualización debido a problemas, ejecute los siguientes comandos.

```
helm history cilium --namespace kube-system
helm rollback cilium [REVISION] --namespace kube-system
```

Elimine Cilium de los nodos híbridos

1. Ejecute el siguiente comando para desinstalar todos los componentes de Cilium del clúster. Tenga en cuenta que desinstalar el CNI podría afectar el estado de los nodos y pods, y no debería realizarse en clústeres de producción.

```
helm uninstall cilium --namespace kube-system
```

   Las interfaces y rutas configuradas por Cilium no se eliminan de forma predeterminada cuando se elimina la CNI del clúster; consulte la edición de GitHub para obtener más información.
2. Para limpiar los archivos de configuración y recursos en disco, si utiliza los directorios de configuración estándar, puede eliminar los archivos como se indica el script `cni-uninstall.sh` proporcionado en el repositorio de Cilium en GitHub.
3. Para eliminar las definiciones de recursos personalizadas (CRD) de Cilium del clúster, puede ejecutar los siguientes comandos.

```
kubectl get crds -oname | grep "cilium" | xargs kubectl delete
```

Cómo configurar complementos para nodos híbridos

En esta página se presentan consideraciones importantes para la ejecución de complementos de AWS y de la comunidad en Nodos híbridos de Amazon EKS. Para obtener más información sobre los complementos de Amazon EKS y los procesos para crearlos, actualizarlos o eliminarlos del clúster, consulte the section called “Complementos de Amazon EKS”. Salvo que se indique lo contrario en esta página, los procesos para crear, actualizar y eliminar complementos de Amazon EKS son los mismos tanto para los clústeres con nodos híbridos como para los clústeres de Amazon EKS con nodos en ejecución en la nube de AWS. Solo los complementos incluidos en esta página han sido validados para garantizar su compatibilidad con Nodos híbridos de Amazon EKS.

Los siguientes complementos de AWS son compatibles con Nodos híbridos de Amazon EKS.

Complemento de AWS	Versiones del complemento compatibles
kube-proxy	v1.25.14-eksbuild.2 y versiones posteriores
CoreDNS	v1.9.3-eksbuild.7 y versiones posteriores
AWS Distro para OpenTelemetry (ADOT)	v0.102.1-eksbuild.2 y versiones posteriores
Agente de observabilidad de CloudWatch	v2.2.1-eksbuild.1 y versiones posteriores
Agente de Pod Identity de EKS	• v1.3.3-eksbuild.1 y versiones posteriores, excepto para Bottlerocket • v1.3.7-eksbuild.2 y versiones posteriores para Bottlerocket
Agente de supervisión de nodos	v1.2.0-eksbuild.1 y versiones posteriores
Controlador de instantáneas CSI	v8.1.0-eksbuild.1 y versiones posteriores
Conector de AWS Private CA para Kubernetes	v1.6.0-eksbuild.1 y versiones posteriores

Los siguientes complementos de la comunidad son compatibles con Nodos híbridos de Amazon EKS. Para obtener más información sobre los complementos de la comunidad, consulte the section called “Complementos de la comunidad”.

Complemento de la comunidad	Versiones del complemento compatibles
Servidor de métricas de Kubernetes	v0.7.2-eksbuild.1 y versiones superiores
cert-manager	v1.17.2-eksbuild.1 y versiones posteriores
Exportador de nodos de Prometheus	v1.9.1-eksbuild.2 y versiones posteriores
kube-state-metrics	v2.15.0-eksbuild.4 y versiones posteriores
DNS externo	v0.19.0-eksbuild.1 y versiones posteriores

Además de los complementos de Amazon EKS que se indican en las tablas anteriores, Amazon Managed Service para Prometheus Collector y el Controlador del equilibrador de carga de AWS para el ingreso de la aplicación (HTTP) y el equilibrio de carga (TCP/UDP) son compatibles con los nodos híbridos.

Existen complementos de AWS y complementos de la comunidad que no son compatibles con los Nodos híbridos de Amazon EKS. Las versiones más recientes de estos complementos incluyen una regla de antiafinidad para la etiqueta predeterminada `eks.amazonaws.com/compute-type: hybrid` aplicada a los nodos híbridos. Esto impide que se ejecuten en nodos híbridos cuando se implementan en los clústeres. Si tiene clústeres con nodos híbridos y nodos que se ejecutan en la nube de AWS, puede implementar estos complementos en el clúster en los nodos que se ejecutan en la nube de AWS. La CNI de Amazon VPC no es compatible con los nodos híbridos, y Cilium y Calico se admiten como interfaces de red de contenedores (CNI) para los Nodos híbridos de Amazon EKS. Para obtener más información, consulte the section called “Cómo configurar una CNI”.

Complementos de AWS

Las siguientes secciones describen las diferencias entre ejecutar complementos de AWS compatibles en nodos híbridos y ejecutarlos en otros tipos de computación de Amazon EKS.

kube-proxy y CoreDNS

EKS instala kube-proxy y CoreDNS como complementos autoadministrados de forma predeterminada cuando se crea un clúster de EKS mediante la API de AWS y los SDK de AWS, incluida AWS CLI. Puede sobrescribir estos complementos con complementos de Amazon EKS después de la creación del clúster. Consulte la documentación de EKS para obtener más información sobre the section called “kube-proxy” ythe section called “CoreDNS”. Si ejecuta un clúster en modo mixto con nodos híbridos y nodos en la nube de AWS, AWS le recomienda tener al menos una réplica de CoreDNS en los nodos híbridos y al menos una réplica de CoreDNS en los nodos en la nube de AWS. Consulte the section called “Configuración de réplicas de CoreDNS” para conocer los pasos de configuración.

Agente de observabilidad de CloudWatch

El operador del agente de observabilidad de CloudWatch utiliza webhooks. Si se ejecuta el operador en nodos híbridos, el CIDR de pods en las instalaciones debe ser enrutable en la red en las instalaciones, y se debe configurar el clúster de EKS con la red de pods remota. Para obtener más información, consulte Configuración de webhooks para nodos híbridos.

Las métricas a nivel de nodo no se encuentran disponibles para los nodos híbridos porque Información de contenedores de CloudWatch depende de la disponibilidad del servicio de metadatos de instancias (IMDS) para las métricas a nivel de nodo. Las métricas de nivel del clúster, de la carga de trabajo, del pod y del contenedor están disponibles para los nodos híbridos.

Tras instalar el complemento según los pasos descritos en Instalación del agente de CloudWatch con Observabilidad de Amazon CloudWatch, se debe actualizar el manifiesto del complemento para que el agente se pueda ejecutar correctamente en nodos híbridos. Edite el recurso `amazoncloudwatchagents` en el clúster para agregar la variable de entorno `RUN_WITH_IRSA`, tal y como se muestra a continuación.

```
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
          ...
```

Recopilador administrado de Amazon Managed para Prometheus para nodos híbridos

Un recopilador administrado de Amazon Managed Service para Prometheus (AMP) consta de un raspador que descubre y recopila métricas de los recursos en un clúster de Amazon EKS. AMP administra el raspador en su nombre, por lo que no tendrá que administrar instancias, agentes o raspadores.

Puede utilizar los recopiladores administrados de AMP sin ninguna configuración adicional específica para los nodos híbridos. Sin embargo, se debe poder acceder a los puntos de conexión métricos de las aplicaciones en los nodos híbridos desde la VPC, incluidas las rutas desde la VPC a los CIDR de la red de pods remotos y los puertos abiertos en el firewall en las instalaciones. Además, el clúster debe tener acceso privado al punto de conexión del clúster.

Siga los pasos que se indican en Uso de un recopilador administrado de AWS en la Guía del usuario de Amazon Managed Service para Prometheus.

AWS Distro para OpenTelemetry (ADOT)

Puede utilizar el complemento AWS Distro para OpenTelemetry (ADOT) para recopilar métricas, registros y datos de trazado de las aplicaciones en ejecución en nodos híbridos. ADOT utiliza webhooks de admisión para modificar y validar las solicitudes del recurso personalizado del recolector. Si ejecuta el operador ADOT en nodos híbridos, el CIDR de pods en las instalaciones debe ser enrutable en la red en las instalaciones y debe configurar el clúster de EKS con la red de pods remota. Para obtener más información, consulte Configuración de webhooks para nodos híbridos.

Siga los pasos que se indican en Introducción a AWS Distro para OpenTelemetry con complementos de EKS en la documentación de AWS Distro para OpenTelemetry.

AWS Controlador del equilibrador de carga de

Puede utilizar el controlador del equilibrador de carga de AWS y el equilibrador de carga de aplicación (ALB) o el equilibrador de carga de red (NLB) con el tipo de destino `ip` para cargas de trabajo en nodos híbridos. Las direcciones IP de destino utilizadas con el ALB o NLB deben ser enrutables desde AWS. El controlador del equilibrador de carga de AWS también utiliza webhooks. Si ejecuta el operador del controlador de equilibrador de carga de AWS en nodos híbridos, el CIDR de pods en las instalaciones debe ser enrutable en la red en las instalaciones y debe configurar el clúster de EKS con la red de pods remota. Para obtener más información, consulte Configuración de webhooks para nodos híbridos.

Para instalar el Controlador del equilibrador de carga de AWS, siga los pasos que se indican en the section called “Equilibrador de carga de aplicación de AWS” o the section called “Equilibrador de carga de red de AWS”.

Para el ingreso con ALB, debe especificar las anotaciones que aparecen a continuación. Para obtener más información, consulte the section called “Equilibrio de carga de aplicaciones”.

```
alb.ingress.kubernetes.io/target-type: ip
```

Para el equilibrio de carga con ALB, debe especificar las anotaciones que aparecen a continuación. Para obtener más información, consulte the section called “Equilibrio de carga de red”.

```
service.beta.kubernetes.io/aws-load-balancer-type: "external"
service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: "ip"
```

Agente de Pod Identity de EKS

> **Note**
>
> Para implementar correctamente el complemento del agente de EKS Pod Identity en los nodos híbridos que ejecutan Bottlerocket, asegúrese de que la versión de Bottlerocket sea la v1.39.0 como mínimo. El agente de Pod Identity no es compatible con versiones anteriores de Bottlerocket en entornos de nodos híbridos.

El DaemonSet original del Agente de Pod Identity de Amazon EKS depende de la disponibilidad del IMDS de EC2 en el nodo para obtener las credenciales de AWS necesarias. Dado que IMDS no está disponible en nodos híbridos, a partir de la versión 1.3.3-eksbuild.1, el complemento del agente de Pod Identity implementa opcionalmente un DaemonSet que monta las credenciales necesarias. Los nodos híbridos que ejecutan Bottlerocket requieren un método diferente para montar las credenciales y, a partir de la versión 1.3.7-eksbuild.2, el complemento del agente de Pod Identity implementa opcionalmente un DaemonSet dirigido específicamente a nodos híbridos de Bottlerocket. En las siguientes secciones se describe el proceso de activación de los DaemonSets opcionales.

Ubuntu, RHEL y AL2023

1. Para utilizar el agente de Pod Identity en nodos híbrido de Ubuntu, RHEL y Al2023, configure `enableCredentialsFile: true` en la sección híbrida de la configuración de `nodeadm`, tal como se muestra a continuación:

   ```
   apiVersion: node.eks.aws/v1alpha1
   kind: NodeConfig
   spec:
       hybrid:
           enableCredentialsFile: true # <-- Add this
   ```

Esto configurará `nodeadm` para crear un archivo de credenciales que se configurará en el nodo bajo `/eks-hybrid/.aws/credentials`, que utilizarán los pods de `eks-pod-identity-agent`. Este archivo de credenciales contendrá credenciales de AWS temporales que se actualizarán periódicamente.

2. Tras actualizar la configuración de `nodeadm` en cada nodo, ejecute el siguiente comando `nodeadm init` con el `nodeConfig.yaml` para unir los nodos híbridos al clúster de Amazon EKS. Si los nodos se han unido al clúster anteriormente, vuelva a ejecutar el comando `nodeadm init`.

```
nodeadm init -c file://nodeConfig.yaml
```

3. Instale `eks-pod-identity-agent` con la compatibilidad con nodos híbridos activada, mediante la AWS CLI o la Consola de administración de AWS.

   a. AWS CLI: desde la máquina que utilice para administrar el clúster, ejecute el siguiente comando para instalar `eks-pod-identity-agent` con la compatibilidad con nodos híbridos habilitada. Reemplace `my-cluster` por el nombre del clúster.

```
aws eks create-addon \
    --cluster-name my-cluster \
    --addon-name eks-pod-identity-agent \
    --configuration-values '{"daemonsets":{"hybrid":{"create": true}}}'
```

   b. Consola de administración de AWS: si va a instalar el complemento del agente de Pod Identity a través de la consola de AWS, agregue lo siguiente a la configuración opcional para implementar el DaemonSet dirigido a los nodos híbridos.

```
{"daemonsets":{"hybrid":{"create": true}}}
```

Bottlerocket

1. Para usar el agente de Pod Identity en los nodos híbridos de Bottlerocket, agregue la marca `--enable-credentials-file=true` al comando utilizado para los datos de usuario del contenedor de arranque de Bottlerocket, tal como se describe en the section called “Conexión de nodos híbridos con Bottlerocket”.

   a. Si utiliza el proveedor de credenciales de SSM, el comando debería tener este aspecto:

```
eks-hybrid-ssm-setup --activation-id=<activation-id> --activation-
code=<activation-code> --region=<region> --enable-credentials-file=true
```

b. Si utiliza el proveedor de credenciales de IAM Roles Anywhere, el comando debería tener este aspecto:

```
eks-hybrid-iam-ra-setup --certificate=<certificate> --key=<private-key> --enable-
credentials-file=true
```

Esto configurará el script de arranque para crear un archivo de credenciales en el nodo de `/var/eks-hybrid/.aws/credentials`, que utilizarán los pods `eks-pod-identity-agent`. Este archivo de credenciales contendrá credenciales de AWS temporales que se actualizarán periódicamente.

2. Instale `eks-pod-identity-agent` con la compatibilidad con nodos híbridos de Bottlerocket activada, mediante la AWS CLI o la Consola de administración de AWS.

a. AWS CLI: desde la máquina que utilice para administrar el clúster, ejecute el siguiente comando para instalar `eks-pod-identity-agent` con la compatibilidad con nodos híbridos de Bottlerocket activada. Reemplace `my-cluster` por el nombre del clúster.

```
aws eks create-addon \
    --cluster-name my-cluster \
    --addon-name eks-pod-identity-agent \
    --configuration-values '{"daemonsets":{"hybrid-bottlerocket":{"create":
 true}}}'
```

b. Consola de administración de AWS: si va a instalar el complemento del agente de Pod Identity a través de la consola de AWS, agregue lo siguiente a la configuración opcional para implementar el DaemonSet dirigido a los nodos híbridos de Bottlerocket.

```
{"daemonsets":{"hybrid-bottlerocket":{"create": true}}}
```

Controlador de instantáneas CSI

A partir de la versión `v8.1.0-eksbuild.2`, el complemento de controlador de instantáneas de CSI aplica una regla de antiafinidad flexible para los nodos híbridos y prefiere que el controlador `deployment` se ejecute en EC2 en la misma región de AWSque el plano de control de Amazon

EKS. La coubicación de `deployment` en la misma región de AWS que el plano de control de Amazon EKS mejora la latencia.

Complementos de la comunidad

Las secciones siguientes describen las diferencias entre ejecutar complementos de la comunidad compatibles en nodos híbridos y en otros tipos de computación de Amazon EKS.

Servidor de métricas de Kubernetes

El plano de control necesita acceder a la dirección IP del pod de Metrics Server (o a la dirección IP del nodo si se habilita hostNetwork). Por lo tanto, a menos que ejecute Metrics Server en modo hostNetwork, debe configurar una red de pods remota al crear el clúster de Amazon EKS y debe hacer que las direcciones IP de los pods sean enrutables. Implementar el protocolo de puerta de enlace fronteriza (BGP) con la CNI es una forma común de hacer que las direcciones IP de los pods sean enrutables.

cert-manager

`cert-manager` utiliza webhooks. Si ejecuta `cert-manager` en nodos híbridos, el CIDR de pods en las instalaciones debe ser enrutable en la red en las instalaciones y debe configurar el clúster de EKS con la red de pods remotos. Para obtener más información, consulte Configuración de webhooks para nodos híbridos.

Configuración de webhooks para nodos híbridos

Esta página presenta consideraciones importantes para la ejecución de webhooks con nodos híbridos. Los webhooks se utilizan en aplicaciones de Kubernetes y en proyectos de código abierto, como Controlador del equilibrador de carga de AWS y el agente de observabilidad de CloudWatch, para realizar funciones de mutación y validación en tiempo de ejecución.

Redes de pods enrutables

Si puede configurar el CIDR del pod en las instalaciones para que sea enrutable en la red en las instalaciones, puede ejecutar webhooks en los nodos híbridos. Existen varias técnicas que puede utilizar para hacer que el CIDR de los pods en las instalaciones sea enrutable en la red en las instalaciones, como el protocolo de puerta de enlace fronteriza (BGP), rutas estáticas u otras soluciones de enrutamiento personalizadas. BGP es la solución recomendada, ya que es más escalable y fácil de administrar que las soluciones alternativas que requieren configuración manual o personalizada de rutas. AWS admite las capacidades de BGP de Cilium y Calico para anunciar los

CIDR de los pods. Consulte the section called “Cómo configurar una CNI” y the section called “CIDR de pod remoto enrutable” para obtener más información.

Redes de pods no enrutables

Si no puede hacer que el CIDR del pod en las instalaciones sea enrutable en la red en las instalaciones y necesita ejecutar webhooks, le recomendamos que ejecute todos los webhooks en los nodos de la nube en el mismo clúster de EKS que los nodos híbridos.

Consideraciones para clústeres en modo mixto

Los clústeres en modo mixto se definen como clústeres de EKS que tienen tanto nodos híbridos como nodos que se ejecutan en la nube de AWS. Al ejecutar un clúster en modo mixto, tenga en cuenta las siguientes recomendaciones:

- Ejecute la CNI de la VPC en los nodos en la nube de AWS y Cilium o Calico en nodos híbridos. Cilium y Calico no son compatibles con AWS cuando se ejecutan en nodos en la nube de AWS.
- Configure los webhooks para que se ejecuten en los nodos en la nube de AWS. Consulte the section called “Configuración de webhooks para complementos” para saber cómo configurar los webhooks para AWS y los complementos de la comunidad.
- Si las aplicaciones requieren que los pods que se ejecutan en nodos en la nube de AWS se comuniquen directamente con los pods en nodos híbridos (“comunicación este-oeste”), y utiliza la CNI de la VPC en los nodos en la nube de AWS, junto con Cilium o Calico en los nodos híbridos, entonces el CIDR de los pods en las instalaciones debe ser enrutable en la red en las instalaciones.
- Ejecute al menos una réplica de CoreDNS en los nodos de la nube de AWS y al menos una réplica de CoreDNS en los nodos híbridos.
- Configure la distribución de tráfico del servicio para mantener el tráfico del servicio local en la zona de la que se origina. Para obtener más información sobre la distribución de tráfico del servicio, consulte the section called “Configurar la distribución de tráfico del servicio”.
- Si utiliza equilibradores de carga de aplicaciones (ALB) o equilibradores de carga de red (NLB) de AWS para el tráfico de las cargas de trabajo que se ejecutan en nodos híbridos, las direcciones IP de destino utilizadas con el ALB o NLB deben ser enrutable desde AWS.
- El complemento Metrics Server requiere conectividad desde el plano de control de EKS hasta la dirección IP del pod de Metrics Server. Si ejecuta el complemento Metrics Server en nodos híbridos, el CIDR de pods en las instalaciones debe ser enrutable dentro de la red en las instalaciones.

- Para recopilar métricas de nodos híbridos con los recolectores administrados por Amazon Managed Service para Prometheus (AMP), el CIDR de pods en las instalaciones debe ser enrutable en la red en las instalaciones. De otro modo, puede utilizar el recopilador administrado de AMP para las métricas del plano de control de EKS y de los recursos que se ejecutan en la nube de AWS, y el complemento AWS Distro para OpenTelemetry (ADOT) para recopilar métricas de los nodos híbridos.

Configurar clústeres en modo mixto

Para ver los webhooks de mutación y validación que se ejecutan en el clúster, puede consultar el tipo de recurso Extensiones en el panel de Recursos de la consola de EKS, o usar los siguientes comandos. EKS también reporta métricas de webhooks en el panel de observabilidad del clúster. Consulte the section called “Panel de observabilidad” para obtener más información.

```
kubectl get mutatingwebhookconfigurations
```

```
kubectl get validatingwebhookconfigurations
```

Configurar la distribución de tráfico del servicio

Cuando ejecute clústeres de modo mixto, le recomendamos que utilice la distribución de tráfico del servicio para mantener el tráfico del servicio local en la zona de la que se origina. La distribución de tráfico del servicio (disponible para versiones de Kubernetes 1.31 y posteriores en EKS) es la solución recomendada en lugar del enrutamiento consciente de la topología, ya que ofrece un comportamiento más predecible. Con la distribución de tráfico del servicio, los puntos de conexión en buen estado dentro de una zona recibirán todo el tráfico correspondiente a esa zona. Con el enrutamiento consciente de la topología, cada servicio debe cumplir varias condiciones en esa zona para que se aplique el enrutamiento personalizado; de lo contrario, el tráfico se distribuye de manera equitativa entre todos los puntos de conexión.

Si utiliza Cilium como CNI, debe ejecutar la CNI con la opción `enable-service-topology` establecida en `true` para habilitar la distribución del tráfico de servicios. Puede transmitir esta configuración con el indicador de instalación de Helm `--set loadBalancer.serviceTopology=true` o puede actualizar una instalación existente con el comando de la CLI de Cilium `cilium config set enable-service-topology true`. El agente de Cilium que se ejecuta en cada nodo se debe reiniciar después de actualizar la configuración de una instalación existente.

En la siguiente sección se muestra un ejemplo de cómo configurar la distribución de tráfico del servicio para el servicio CoreDNS, y le recomendamos que habilite la misma para todos los servicios del clúster a fin de evitar el tráfico no deseado entre entornos.

Configuración de réplicas de CoreDNS

Si ejecuta un clúster en modo mixto tanto con nodos híbridos como con nodos en la nube de AWS, se recomienda contar con al menos una réplica de CoreDNS en los nodos híbridos y otra en los nodos ubicados en la nube de AWS. Para evitar problemas de latencia y de red en un clúster en modo mixto, puede configurar el servicio CoreDNS de modo que prefiera la réplica de CoreDNS más cercana mediante la distribución de tráfico del servicio.

La distribución de tráfico del servicio (disponible para versiones de Kubernetes 1.31 y posteriores en EKS) es la solución recomendada en lugar del enrutamiento consciente de la topología, ya que ofrece un comportamiento más predecible. En la distribución de tráfico del servicio, los puntos de conexión en buen estado dentro de una zona recibirán todo el tráfico correspondiente a esa zona. En el enrutamiento consciente de la topología, cada servicio debe cumplir varias condiciones en esa zona para que se aplique el enrutamiento personalizado; de lo contrario, el tráfico se distribuye de manera equitativa entre todos los puntos de conexión. A continuación, se describen los pasos para configurar la distribución de tráfico del servicio.

Si utiliza Cilium como CNI, debe ejecutar la CNI con la opción `enable-service-topology` establecida en `true` para habilitar la distribución del tráfico de servicios. Puede transmitir esta configuración con el indicador de instalación de Helm `--set loadBalancer.serviceTopology=true` o puede actualizar una instalación existente con el comando de la CLI de Cilium `cilium config set enable-service-topology true`. El agente de Cilium que se ejecuta en cada nodo se debe reiniciar después de actualizar la configuración de una instalación existente.

1. Agregue una etiqueta de zona de topología para cada uno de los nodos híbridos, por ejemplo `topology.kubernetes.io/zone: onprem`. O bien, puede establecer la etiqueta en la fase `nodeadm init`. Para ello, especifíquela en la configuración `nodeadm` (consulte the section called “Configuración de nodos para personalizar kubelet (opcional)”). Nota: Los nodos que se ejecutan en la nube de AWS reciben automáticamente una etiqueta de zona de topología que corresponde a la zona de disponibilidad (AZ) del nodo.

   ```
   kubectl label node hybrid-node-name topology.kubernetes.io/zone=zone
   ```

2. Agregue `podAntiAffinity` a la implementación de CoreDNS con la clave de zona de topología. De otro modo, puede configurar la implementación de CoreDNS durante la instalación con complementos de EKS.

```
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. Añada el ajuste `trafficDistribution: PreferClose` a la configuración del servicio de `kube-dns` para habilitar la distribución de tráfico del servicio.

```
kubectl patch svc kube-dns -n kube-system --type=merge -p '{
  "spec": {
    "trafficDistribution": "PreferClose"
  }
}'
```

4. Para confirmar que la distribución del tráfico del servicio está habilitada, puede consultar los segmentos de puntos de conexión del servicio `kube-dns`. Los segmentos de puntos de conexión deben mostrar las `hints` de las etiquetas de zona topológica, lo que confirma que la distribución del tráfico del servicio está habilitada. Si no ve la `hints` para cada dirección de punto de conexión, significa que la distribución del tráfico del servicio no está habilitada.

```
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
```

Configuración de webhooks para complementos

Los siguientes complementos utilizan webhooks y son compatibles para su uso con nodos híbridos.

- Controlador del equilibrador de carga de AWS
- Agente de observabilidad de Amazon CloudWatch
- AWS Distro para OpenTelemetry (ADOT)
- `cert-manager`

Consulte las secciones que aparecen a continuación para configurar los webhooks que estos complementos utilizan para que se ejecuten en nodos en la nube de AWS.

Controlador del equilibrador de carga de AWS

Para utilizar el controlador del equilibrador de carga de AWS en una configuración de clúster de modo mixto, debe ejecutar el controlador en nodos en la nube de AWS. Para ello, agregue lo siguiente a la configuración de valores de Helm o especifique los valores mediante la configuración del complemento de EKS.

```
affinity:
  nodeAffinity:
    requiredDuringSchedulingIgnoredDuringExecution:
      nodeSelectorTerms:
      - matchExpressions:
        - key: eks.amazonaws.com/compute-type
          operator: NotIn
          values:
          - hybrid
```

Agente de observabilidad de Amazon CloudWatch

El complemento del agente de observabilidad de CloudWatch tiene un operador de Kubernetes que utiliza webhooks. Para ejecutar el operador en nodos en la nube de AWS dentro de una configuración de clúster en modo mixto, edite la configuración del operador del agente de observabilidad de CloudWatch. No es posible configurar la afinidad del operador durante la instalación con Helm y los complementos de EKS (consulte incidencia n.º 2431 containers-roadmap).

```
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 para OpenTelemetry (ADOT)

El complemento de AWS Distro para OpenTelemetry (ADOT) tiene un operador de Kubernetes que utiliza webhooks. Para ejecutar el operador en nodos en la nube de AWS dentro de una configuración de clúster en modo mixto, agregue lo siguiente a la configuración de valores de Helm o especifique los valores mediante la configuración del complemento de EKS.

```
affinity:
  nodeAffinity:
    requiredDuringSchedulingIgnoredDuringExecution:
      nodeSelectorTerms:
      - matchExpressions:
        - key: eks.amazonaws.com/compute-type
          operator: NotIn
          values:
          - hybrid
```

Si el CIDR de pods no es enrutable en la red en las instalaciones, el recopilador de ADOT se debe ejecutar en los nodos híbridos para obtener las métricas de los nodos híbridos y de las cargas de trabajo que se ejecutan en estos. Para ello, edite la definición de recurso personalizado (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
```

Puede configurar el recopilador de ADOT para que obtenga únicamente métricas de los nodos híbridos y de los recursos que se ejecutan en estos. Para ello, agregeue las siguientes `relabel_configs` a cada `scrape_configs` en la configuración de CRD del recopilador de ADOT.

```
relabel_configs:
  - action: keep
    regex: hybrid
    source_labels:
    - __meta_kubernetes_node_label_eks_amazonaws_com_compute_type
```

El complemento ADOT tiene como requisito previo instalar `cert-manager` para los certificados TLS utilizados por el webhook del operador de ADOT. `cert-manager` también ejecuta webhooks y se puede configurar de manera que se ejecute en nodos en la nube de AWS con la siguiente configuración de valores de Helm.

```
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

El complemento `cert-manager` ejecuta webhooks y se puede configurar de manera que se ejecute en nodos en la nube de AWS con la siguiente configuración de valores de Helm.

```
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
```

Configuración del proxy para nodos híbridos

Si utiliza un servidor proxy en el entorno en las instalaciones para el tráfico que sale del centro de datos o del entorno periférico, debe configurar por separado los nodos y el clúster para usar el servidor proxy.

Clúster

En el clúster, debe configurar `kube-proxy` para usar el servidor proxy. Debe configurar `kube-proxy` después de crear el clúster de Amazon EKS.

Nodos

En los nodos, debe configurar el sistema operativo, `containerd`, `kubelet` y el agente Amazon SSM para usar el servidor proxy. Puede realizar estos cambios durante el proceso de creación de las imágenes del sistema operativo o antes de ejecutar `nodeadm init` en cada nodo híbrido.

Configuración a nivel de nodo

Debe aplicar las siguientes configuraciones ya sea en las imágenes de sistema operativo o antes de ejecutar `nodeadm init` en cada nodo híbrido.

Configuración del proxy **containerd**

`containerd` es el tiempo de ejecución de administración de contenedores predeterminado para Kubernetes. Si utiliza un proxy para acceder a Internet, debe configurar `containerd` de modo que pueda extraer las imágenes de contenedor necesarias para Kubernetes y Amazon EKS.

Cree un archivo en cada nodo híbrido llamado `http-proxy.conf` en el directorio `/etc/systemd/system/containerd.service.d` con el siguiente contenido. Sustituya `proxy-domain` y `port` por los valores correspondientes al entorno.

```
[Service]
Environment="HTTP_PROXY=http://proxy-domain:port"
Environment="HTTPS_PROXY=http://proxy-domain:port"
Environment="NO_PROXY=localhost"
```

Configuración de **containerd** a partir de datos de usuarios

Será necesario crear el directorio `containerd.service.d` para este archivo. Tendrá que volver a cargar systemd para recuperar el archivo de configuración sin necesidad de reiniciar. En AL2023, es probable que el servicio ya esté en ejecución cuando se ejecute el script, por lo que también tendrá que reiniciarlo.

```
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
```

Configuración del proxy **kubelet**

`kubelet` es el agente de nodos de Kubernetes que se ejecuta en cada nodo de Kubernetes y es responsable de administrar el nodo y los pods que se ejecutan en este. Si utiliza un proxy en el entorno en las instalaciones, debe configurar el `kubelet` para que pueda comunicarse con los puntos de conexión públicos o privados del clúster de Amazon EKS.

Cree un archivo en cada nodo híbrido llamado `http-proxy.conf` en el directorio `/etc/systemd/system/kubelet.service.d/` con el siguiente contenido. Sustituya `proxy-domain` y `port` por los valores correspondientes al entorno.

```
[Service]
Environment="HTTP_PROXY=http://proxy-domain:port"
Environment="HTTPS_PROXY=http://proxy-domain:port"
Environment="NO_PROXY=localhost"
```

Configuración de **`kubelet`** a partir de datos de usuarios

Se debe crear el directorio `kubelet.service.d` para este archivo. Tendrá que volver a cargar systemd para recuperar el archivo de configuración sin necesidad de reiniciar. En AL2023, es probable que el servicio ya esté en ejecución cuando se ejecute el script, por lo que también tendrá que reiniciarlo.

```
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
```

Configuración del proxy **`ssm`**

`ssm` es uno de los proveedores de credenciales que se pueden utilizar para inicializar un nodo híbrido. `ssm` es responsable de autenticarse con AWS y generar credenciales temporales que son utilizadas por `kubelet`. Si utiliza un proxy en el entorno en las instalaciones y usa `ssm` como proveedor de credenciales en el nodo, debe configurar `ssm` para que pueda comunicarse con los puntos de conexión del servicio de Amazon SSM.

Cree un archivo en cada nodo híbrido llamado `http-proxy.conf` en la siguiente ruta, según el sistema operativo

- Ubuntu - `/etc/systemd/system/snap.amazon-ssm-agent.amazon-ssm-agent.service.d/http-proxy.conf`

- Amazon Linux 2023 y Red Hat Enterprise Linux - `/etc/systemd/system/amazon-ssm-agent.service.d/http-proxy.conf`

Rellene el archivo con el siguiente contenido. Sustituya `proxy-domain` y `port` por los valores correspondientes al entorno.

```
[Service]
Environment="HTTP_PROXY=http://proxy-domain:port"
Environment="HTTPS_PROXY=http://proxy-domain:port"
Environment="NO_PROXY=localhost"
```

Configuración de **`ssm`** a partir de datos de usuarios

Es necesario crear el directorio para el archivo de servicio systemd de `ssm`. La ruta del directorio depende del sistema operativo utilizado en el nodo.

- Ubuntu - `/etc/systemd/system/snap.amazon-ssm-agent.amazon-ssm-agent.service.d`
- Amazon Linux 2023 y Red Hat Enterprise Linux - `/etc/systemd/system/amazon-ssm-agent.service.d`

Reemplace el nombre del servicio systemd en el comando de reinicio a continuación, en función del sistema operativo utilizado en el nodo.

- Ubuntu - `snap.amazon-ssm-agent.amazon-ssm-agent`
- Amazon Linux 2023 y 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
```

Configuración del proxy del sistema operativo

Si utiliza un proxy para acceder a Internet, debe configurar el sistema operativo de modo que pueda extraer las dependencias de los nodos híbridos del administrador de paquetes del sistema operativo.

Ubuntu

1. Configure `snap` para usar el proxy con los siguientes comandos:

```
sudo snap set system proxy.https=http://proxy-domain:port
sudo snap set system proxy.http=http://proxy-domain:port
```

2. Para habilitar el proxy para `apt`, cree un archivo llamado `apt.conf` en el directorio `/etc/apt/`. Sustituya proxy-domain y puerto por los valores correspondientes al entorno.

```
Acquire::http::Proxy "http://proxy-domain:port";
Acquire::https::Proxy "http://proxy-domain:port";
```

Amazon Linux 2023

1. Configure `dnf` para utilizar el proxy. Cree un archivo `/etc/dnf/dnf.conf` con los valores proxy-domain y de puerto correspondientes al entorno.

```
proxy=http://proxy-domain:port
```

Red Hat Enterprise Linux

1. Configure `yum` para utilizar el proxy. Cree un archivo `/etc/yum.conf` con los valores proxy-domain y de puerto correspondientes al entorno.

```
proxy=http://proxy-domain:port
```

Configuración del proxy de IAM Roles Anywhere

El servicio de proveedor de credenciales de IAM Roles Anywhere se ocupa de actualizar las credenciales cuando se utiliza IAM Roles Anywhere con la marca `enableCredentialsFile` (consulte the section called “Agente de Pod Identity de EKS”). Si utiliza un proxy en el entorno en las instalaciones, debe configurar el servicio para que pueda comunicarse con los puntos de conexión de IAM Roles Anywhere.

Cree un archivo llamado `http-proxy.conf` en el directorio `/etc/systemd/system/aws_signing_helper_update.service.d/` con el siguiente contenido. Sustituya `proxy-domain` y `port` por los valores correspondientes al entorno.

```
[Service]
Environment="HTTP_PROXY=http://proxy-domain:port"
Environment="HTTPS_PROXY=http://proxy-domain:port"
Environment="NO_PROXY=localhost"
```

Configuración de todo el clúster

Las configuraciones de esta sección se deben aplicar después de crear el clúster de Amazon EKS y antes de ejecutar `nodeadm init` en cada nodo híbrido.

Configuración del proxy kube-proxy

Amazon EKS instala automáticamente `kube-proxy` en cada nodo híbrido como DaemonSet cuando los nodos híbridos se unen al clúster. `kube-proxy` permite el enrutamiento entre servicios respaldados por pods en clústeres de Amazon EKS. Para configurar cada host, `kube-proxy` requiere la resolución DNS para el punto de conexión del clúster de Amazon EKS.

1. Edite el DaemonSet `kube-proxy` con el siguiente comando

   ```
   kubectl -n kube-system edit ds kube-proxy
   ```

   Esto abrirá la definición del DaemonSet `kube-proxy` en el editor configurado.
2. Agregue las variables de entorno para `HTTP_PROXY` y `HTTPS_PROXY`. Tenga en cuenta que la variable de entorno `NODE_NAME` ya debe existir en la configuración. Sustituya `proxy-domain` y `port` por los valores correspondientes al entorno.

   ```
   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
```

Configuración del BGP de Cilium para nodos híbridos

En este tema se describe cómo configurar el protocolo de puerta de enlace fronteriza (BGP) de Cilium para los nodos híbridos de Amazon EKS. La funcionalidad del BGP de Cilium se denomina Cilium BGP Control Plane y se puede utilizar para anunciar las direcciones de los pods y los servicios en la red en las instalaciones. Para ver métodos alternativos para hacer que los CIDR de los pods sean enrutables en la red en las instalaciones, consulte the section called “CIDR de pod remoto enrutable”.

Configuración del BGP de Cilium

Requisitos previos

- Se instaló Cilium según las instrucciones de the section called “Cómo configurar una CNI”.

Procedimiento

1. Para usar el BGP con Cilium a fin de anunciar las direcciones de los pods o los servicios en la red en las instalaciones, debe haber instalado Cilium con `bgpControlPlane.enabled: true`. Si activa BGP para una implementación de Cilium existente, debe reiniciar el operador de Cilium para aplicar la configuración de BGP si BGP no se activó anteriormente. Puede configurar `operator.rollOutPods` como `true` en sus valores de Helm para reiniciar el operador de Cilium como parte del proceso de instalación o actualización de Helm.

```
helm upgrade cilium oci://public.ecr.aws/eks/cilium/cilium \
  --namespace kube-system \
```

```
  --reuse-values \
  --set operator.rollOutPods=true \
  --set bgpControlPlane.enabled=true
```

2. Confirme que el operador y los agentes de Cilium se hayan reiniciado y estén funcionando.

```
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. Cree un archivo llamado `cilium-bgp-cluster.yaml` con una definición `CiliumBGPClusterConfig`. Es posible que deba obtener la siguiente información de su administrador de red.
   - Configure `localASN` con el ASN para los nodos que ejecutan Cilium.
   - Configure `peerASN` con el ASN de su enrutador en las instalaciones.
   - Configure `peerAddress` con la IP del enrutador en las instalaciones con la que se emparejará cada nodo que ejecute Cilium.

```
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. Aplique la configuración de clúster del BGP de Cilium al clúster.

```
kubectl apply -f cilium-bgp-cluster.yaml
```

5. Cree un archivo denominado `cilium-bgp-peer.yaml` con el recurso `CiliumBGPPeerConfig` que define una configuración de emparejamiento del BGP. Varios pares pueden compartir la misma configuración y proporcionar una referencia al recurso `CiliumBGPPeerConfig` común. Consulte BGP Peer configuration en la documentación de Cilium para obtener una lista completa de las opciones de configuración.

   Los valores de las siguientes configuraciones de emparejamiento de Cilium deben coincidir con los del router en las instalaciones con el que se empareja.

   - Configure `holdTimeSeconds` para determinar cuánto tiempo espera un emparejamiento del BGP a recibir un mensaje de mantenimiento o actualización antes de declarar la sesión inactiva. El valor predeterminado es de 90 segundos.
   - Configure `keepAliveTimeSeconds` para determinar si aún se puede acceder a un emparejamiento del BGP y si la sesión del BGP está activa. El valor predeterminado es de 30 segundos.
   - Configure `restartTimeSeconds` para determinar el momento en que se espera que el plano de control del BGP de Cilium restablezca la sesión del BGP tras un reinicio. El valor predeterminado es de 120 segundos.

```
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. Aplique la configuración de emparejamiento del BGP de Cilium al clúster.

```
kubectl apply -f cilium-bgp-peer.yaml
```

7. Cree un archivo llamado `cilium-bgp-advertisement-pods.yaml` con un recurso `CiliumBGPAdvertisement` para anunciar los CIDR de pods en su red en las instalaciones.
   - El recurso `CiliumBGPAdvertisement` se utiliza para definir varios tipos de anuncios y los atributos asociados a estos. En el siguiente ejemplo, se configura Cilium para que anuncie únicamente los CIDR de pods. Consulte los ejemplos de the section called “LoadBalancer del tipo de servicio” y the section called “Equilibrio de carga en el clúster de Cilium” para obtener más información sobre la configuración de Cilium para anunciar las direcciones de los servicios.
   - Cada nodo híbrido que ejecuta el agente de Cilium se empareja con el router ascendente compatible con el BGP. Cada nodo anuncia el rango de CIDR del pod que posee cuando el `advertisementType` de Cilium está configurado en `PodCIDR` como en el ejemplo siguiente. Consulte BGP Advertisements configuration en la documentación de Cilium para obtener más información.

```
apiVersion: cilium.io/v2alpha1
kind: CiliumBGPAdvertisement
metadata:
  name: bgp-advertisement-pods
  labels:
    advertise: bgp
spec:
  advertisements:
    - advertisementType: "PodCIDR"
```

8. Aplique la configuración de anuncio de BGP de Cilium al clúster.

```
kubectl apply -f cilium-bgp-advertisement-pods.yaml
```

9. Puede confirmar que el emparejamiento de BGP funcionó con la CLI de Cilium mediante el comando `cilium bgp peers`. Debería ver los valores correctos en la salida del entorno y el estado de la sesión como `established`. Consulte la Guía de solución de problemas y

operaciones en la documentación de Cilium para obtener más información sobre la solución de problemas.

En los ejemplos siguientes, hay cinco nodos híbridos que ejecutan el agente de Cilium y cada nodo anuncia el rango de CIDR de pods que posee.

```
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
```

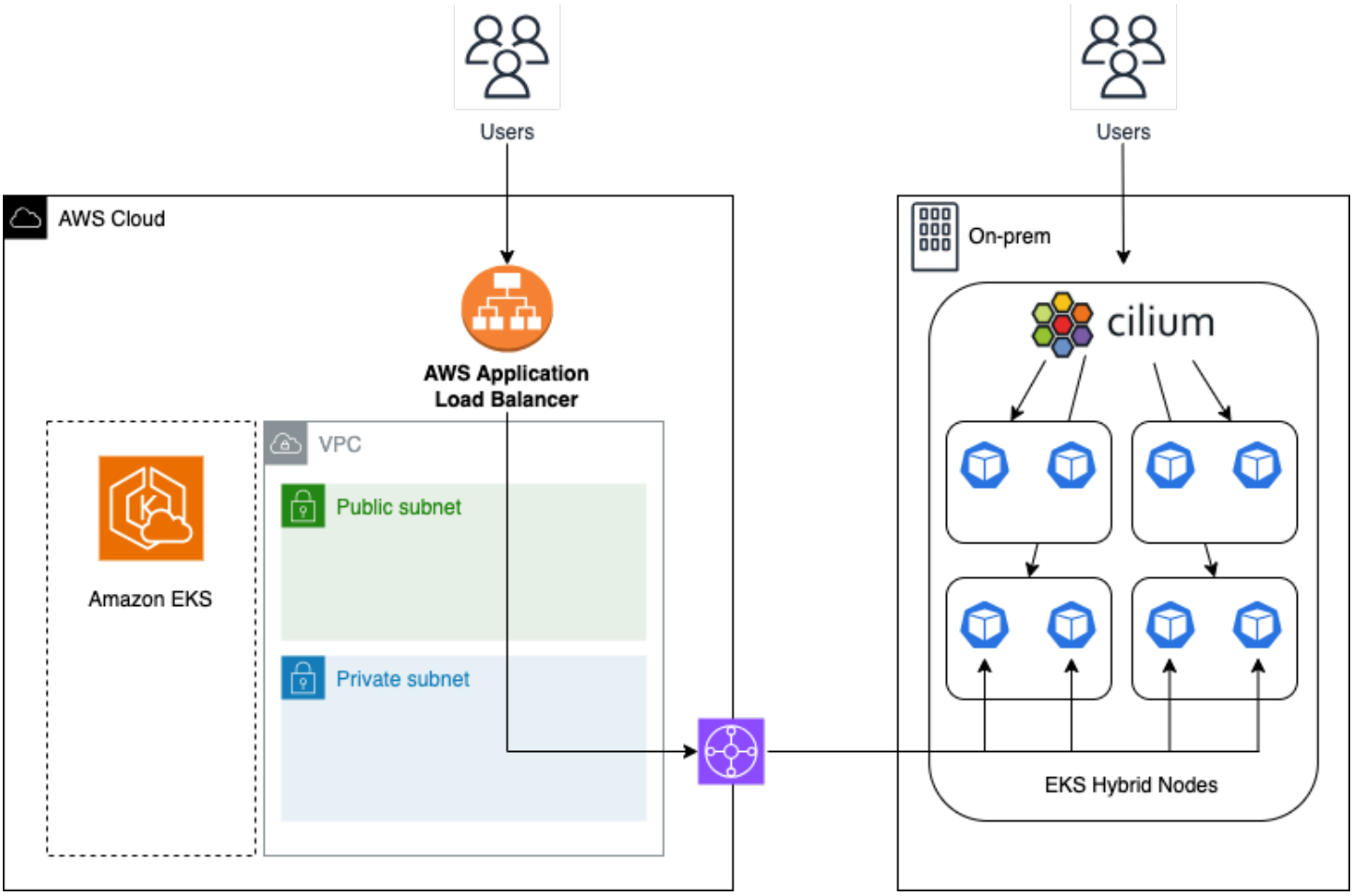
```
Node                   VRouter       Prefix           NextHop   Age         Attrs
mi-026d6a261e355fba7   NODES_ASN     10.86.2.0/26     0.0.0.0   1h16m46s   [{Origin:
 i} {Nexthop: 0.0.0.0}]
mi-082f73826a163626e   NODES_ASN     10.86.2.192/26   0.0.0.0   1h16m46s   [{Origin:
 i} {Nexthop: 0.0.0.0}]
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}]
```

Configuración de Kubernetes Ingress para nodos híbridos

En este tema se describe cómo configurar Kubernetes Ingress para las cargas de trabajo que se ejecutan en Nodos híbridos de Amazon EKS. Kubernetes Ingress expone las rutas HTTP y HTTPS desde fuera del clúster a los servicios del clúster. Para utilizar los recursos de Ingress, se necesita un controlador de Kubernetes Ingress a fin de configurar la infraestructura de red y los componentes que sirven al tráfico de la red.

AWS es compatible con el Equilibrador de carga de aplicación (ALB) de AWS y Cilium for Kubernetes Ingress para cargas de trabajo que se ejecutan en Nodos híbridos de EKS. La decisión de utilizar ALB o Cilium para la entrada se basa en el origen del tráfico de aplicaciones. Si el tráfico de aplicaciones se origina en una región de AWS, AWS recomienda utilizar el ALB de AWS y el Controlador del equilibrador de carga de AWS. Si el tráfico de aplicaciones se origina en un entorno local en las instalaciones o en la periferia, AWS recomienda utilizar las funciones de entrada integradas de Cilium, que se pueden utilizar con o sin una infraestructura de equilibradores de carga en su entorno.



Equilibrador de carga de aplicación de AWS

Puede usar el Controlador del equilibrador de carga de AWS y el Equilibrador de carga de aplicación (ALB) con el tipo de destino `ip` para cargas de trabajo que se ejecutan en nodos híbridos. Cuando se utiliza el tipo de destino `ip`, el ALB reenvía el tráfico directamente a los pods, sin pasar por la ruta de red de la capa de servicio. Para que el ALB llegue a los destinos de IP de pods en nodos híbridos, el CIDR de pods en las instalaciones debe ser enrutable en la red en las instalaciones. Además, el Controlador del equilibrador de carga de AWS utiliza webhooks y requiere una comunicación directa desde el plano de control de EKS. Para obtener más información, consulte the section called "Configuración de webhooks".

Consideraciones

- Consulte the section called "Equilibrio de carga de aplicaciones" y the section called "Instalación con Helm" para obtener más información sobre el Equilibrador de carga de aplicación de AWS y Controlador del equilibrador de carga de AWS.

- Consulte Best Practices for Load Balancing para obtener información sobre cómo elegir entre el Equilibrador de carga de aplicación de AWS y el Equilibrador de carga de red de AWS.
- Consulte AWS Load Balancer Controller Ingress annotations para obtener la lista de anotaciones que se pueden configurar para los recursos de entrada con el Equilibrador de carga de aplicación de AWS.

Requisitos previos

- Se instaló Cilium según las instrucciones de the section called “Cómo configurar una CNI”.
- Cilium BGP Control Plane está activado según las instrucciones de the section called “Configuración de BGP”. Si no desea usar el BGP, debe usar un método alternativo para que los CIDR de pods en las instalaciones sean enrutables dentro de la red en las instalaciones. Si no hace que los CIDR de sus pods en las instalaciones sean enrutables, el ALB no podrá registrar los destinos de IP de su pod ni contactar con ellos.
- Si tiene Helm instalado en su entorno de línea de comandos, consulte las instrucciones de configuración de Helm para obtener más información.
- Si tiene eksctl instalado en su entorno de línea de comandos, consulte las instrucciones de instalación de eksctl para obtener más información.

Procedimiento

1. Descargue una política de IAM para el Controlador del equilibrador de carga de AWS que le permita realizar llamadas a las API de AWS en su nombre.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-
controller/refs/heads/main/docs/install/iam_policy.json
```

2. Cree una política de IAM con la política descargada en el paso anterior.

```
aws iam create-policy \
    --policy-name AWSLoadBalancerControllerIAMPolicy \
    --policy-document file://iam_policy.json
```

3. Sustituya el valor del nombre del clúster (`CLUSTER_NAME`), la región de AWS (`AWS_REGION`) y el ID de la cuenta de AWS (`AWS_ACCOUNT_ID`) por su configuración y ejecute el siguiente comando.

```
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. Agregue el repositorio de gráficos de Helm y actualice el repositorio local para asegurarse de que cuenta con los gráficos más recientes.

```
helm repo add eks https://aws.github.io/eks-charts
```

```
helm repo update eks
```

5. Instale el Controlador del equilibrador de carga de AWS. Sustituya el valor del nombre del clúster (`CLUSTER_NAME`), la región de AWS (`AWS_REGION`), el ID de VPC (`VPC_ID`) y la versión del gráfico de Helm del Controlador del equilibrador de carga de AWS (`AWS_LBC_HELM_VERSION`) por su configuración y ejecute el siguiente comando. Si ejecuta un clúster en modo mixto con nodos híbridos y nodos en la nube de AWS, puede ejecutar el Controlador del equilibrador de carga de AWS en los nodos en la nube según las instrucciones de the section called “Controlador del equilibrador de carga de AWS”.
   - Para encontrar la versión más reciente del gráfico de Helm, ejecute `helm search repo eks/aws-load-balancer-controller --versions`.

```
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. Compruebe que el Controlador del equilibrador de carga de AWS se haya instalado correctamente.

```
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. Cree una aplicación de muestra. En el siguiente ejemplo, se utiliza la aplicación de microservicios de ejemplo Istio BookInfo.

```
kubectl apply -f https://raw.githubusercontent.com/istio/istio/refs/heads/master/
samples/bookinfo/platform/kube/bookinfo.yaml
```

8. Cree un archivo denominado `my-ingress-alb.yaml` con el siguiente contenido.

```
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. Aplique la configuración de entrada al clúster.

```
kubectl apply -f my-ingress-alb.yaml
```

10. El aprovisionamiento del ALB para el recurso de entrada puede tardar unos minutos. Una vez aprovisionado el ALB, el recurso de entrada tendrá asignada una dirección que se corresponderá

con el nombre de DNS de la implementación del ALB. La dirección tendrá el formato <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 Acceda al servicio con la dirección del 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"
}
```

Información general sobre Cilium Ingress y Cilium Gateway

Las capacidades de entrada de Cilium están integradas en la arquitectura de Cilium y se pueden administrar con API de Gateway o la API de Kubernetes Ingress. Si no tiene recursos de entrada existentes, AWS recomienda que comience con la API de Gateway, ya que es una forma más expresiva y flexible de definir y administrar los recursos de red de Kubernetes. La API de Kubernetes Gateway tiene como objetivo estandarizar la forma en que se definen y administran los recursos de red de entrada, equilibrador de carga y malla de servicios en los clústeres de Kubernetes.

Al activar las características de entrada o puerta de enlace de Cilium, el operador de Cilium reconcilia los objetos de entrada y puerta de enlace del clúster y los proxies de Envoy de cada nodo procesan

el tráfico de red de capa 7 (L7). Cilium no aprovisiona directamente la infraestructura de entrada y puerta de enlace, como los equilibradores de carga. Si planea usar Cilium Ingress o Gateway con un equilibrador de carga, debe usar las herramientas del equilibrador de carga, normalmente un controlador de entrada o puerta de enlace, para implementar y administrar la infraestructura del equilibrador de carga.

Para el tráfico de entrada o puerta de enlace, Cilium gestiona el tráfico de la red principal y el cumplimiento de las políticas de L3/L4, y los proxies de Envoy integrados procesan el tráfico de red de L7. Con Cilium Ingress o Gateway, Envoy es responsable de aplicar las reglas de enrutamiento de capa 7, las políticas y la manipulación de solicitudes, la administración avanzada del tráfico, como la división y duplicación del tráfico, y la terminación y el origen del TLS. Los proxies de Envoy de Cilium se implementan como un DaemonSet (`cilium-envoy`) independiente de forma predeterminada, lo que permite a Envoy y el agente de Cilium actualizarse, escalarse y administrarse por separado.

Para obtener más información sobre cómo funcionan Cilium Ingress y Cilium Gateway, consulte las páginas Cilium Ingress y Cilium Gateway en la documentación de Cilium.

Comparación entre Cilium Ingress y Gateway

En la siguiente tabla se resumen las características de Cilium Ingress y Cilium Gateway de la versión 1.17.x de Cilium.

Característica	Ingress	Puerta de enlace
LoadBalancer del tipo de servicio	Sí	Sí
NodePort del tipo de servicio	Sí	No[1]
Red del host	Sí	Sí
Equilibrador de carga compartido	Sí	Sí
Equilibrador de carga dedicado	Sí	No[2]
Políticas de red	Sí	Sí
Protocolos	Capa 7 (HTTP(S), gRPC)	Capa 7 (HTTP(S), gRPC)[3]
Acceso directo de TLS	Sí	Sí

Característica	Ingress	Puerta de enlace
Administración del tráfico	Enrutamiento de rutas y hosts	Enrutamiento de rutas y hosts, redirección y reescritura de URL, división de tráfico, modificación de encabezados

[1] Está prevista la compatibilidad de Cilium Gateway con servicios NodePort en la versión 1.18.x (#27273) de Cilium

[2] Compatibilidad de Cilium Gateway con equilibradores de carga dedicados (#25567)

[3] Compatibilidad de Cilium Gateway con TCP/UDP (#21929)

Instalación de Cilium Gateway

Consideraciones

- Cilium debe configurarse con `nodePort.enabled` establecido en `true`, como se muestra en los ejemplos siguientes. Si utiliza la característica de reemplazo de kube-proxy de Cilium, no necesita configurar `nodePort.enabled` en `true`.
- Cilium debe configurarse con `envoy.enabled` establecido en `true`, como se muestra en los ejemplos siguientes.
- Cilium Gateway se puede implementar en modo de equilibrador de carga (predeterminado) o en modo de red host.
- Cuando se utiliza Cilium Gateway en modo de equilibrador de carga, la anotación `service.beta.kubernetes.io/aws-load-balancer-type: "external"` debe estar configurada en el recurso de puerta de enlace para evitar que el proveedor de nube de AWS heredado cree un Equilibrador de carga clásico para el servicio del tipo LoadBalancer que Cilium crea para el recurso de puerta de enlace.
- Cuando se utiliza Cilium Gateway en el modo de red host, el servicio del tipo LoadBalancer está desactivado. El modo de red host es útil para entornos que no tienen una infraestructura de equilibrador de carga. Consulte the section called “Red del host” para obtener más información.

Requisitos previos

1. Si tiene Helm instalado en su entorno de línea de comandos, consulte las instrucciones de configuración de Helm.
2. Se instaló Cilium según las instrucciones de the section called “Cómo configurar una CNI”.

Procedimiento

1. Instale las definiciones de recursos personalizados (CRD) de la API de Kubernetes Gateway.

```
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. Cree un archivo denominado `cilium-gateway-values.yaml` con el siguiente contenido. En el siguiente ejemplo se configura Cilium Gateway para usar el modo de equilibrador de carga predeterminado y usar un DaemonSet `cilium-envoy` independiente para los proxies de Envoy configurados para que se ejecuten solo en nodos híbridos.

```
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. Aplique el archivo de valores de Helm al clúster.

```
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. Confirme que el operador, el agente y los pods de Envoy de Cilium estén en ejecución.

```
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
```

Configuración de Cilium Gateway

Cilium Gateway se activa en objetos de puerta de enlace mediante la configuración de `gatewayClassName` como `cilium`. El servicio que Cilium crea para los recursos de puerta de enlace se puede configurar con los campos del objeto de puerta de enlace. Las anotaciones habituales que utilizan los controladores de puerta de enlace para configurar la infraestructura del equilibrador de carga se pueden configurar con el campo `infrastructure` del objeto de puerta de enlace. Cuando se utiliza el IPAM LoadBalancer de Cilium (consulte el ejemplo de the section called “LoadBalancer del tipo de servicio”), la dirección IP que se utilizará para el servicio del tipo LoadBalancer se puede configurar en el campo `addresses` del objeto de puerta de enlace. Para

obtener más información sobre la configuración de la puerta de enlace, consulte la especificación de la API de Kubernetes Gateway.

```
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 y la especificación de Kubernetes Gateway admiten los recursos GatewayClass, de puerta de enlace, HTTPRoute, GRPCRoute y ReferenceGrant.

- Consulte las especificaciones de HTTPRoute y GRPCRoute para ver la lista de campos disponibles.
- Consulte los ejemplos de la sección the section called “Implementación de Cilium Gateway” a continuación y los ejemplos de la documentación de Cilium para saber cómo utilizar y configurar estos recursos.

Implementación de Cilium Gateway

1. Cree una aplicación de muestra. En el siguiente ejemplo, se utiliza la aplicación de microservicios de ejemplo Istio BookInfo.

   ```
   kubectl apply -f https://raw.githubusercontent.com/istio/istio/refs/heads/master/
   samples/bookinfo/platform/kube/bookinfo.yaml
   ```

2. Compruebe que la aplicación se esté ejecutando correctamente.

   ```
   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. Cree un archivo denominado `my-gateway.yaml` con el siguiente contenido. En el ejemplo siguiente se utiliza la anotación `service.beta.kubernetes.io/aws-load-balancer-type: "external"` para evitar que el proveedor de nube de AWS heredado cree un Equilibrador de carga clásico para el servicio del tipo LoadBalancer que Cilium crea para el recurso de puerta de enlace.

```
---
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. Aplique el recurso de puerta de enlace al clúster.

```
kubectl apply -f my-gateway.yaml
```

5. Confirme que se hayan creado el recurso de puerta de enlace y el servicio correspondiente. En esta fase, se espera que el campo ADDRESS del recurso de puerta de enlace no se complete con una dirección IP o un nombre de host y que el servicio del tipo LoadBalancer para el recurso de puerta de enlace tampoco tenga una dirección IP o un nombre de host asignados.

```
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. Continúe con the section called “LoadBalancer del tipo de servicio” para configurar el recurso de puerta de enlace a fin de que utilice una dirección IP asignada por Cilium Load Balancer IPAM, y con the section called “NodePort del tipo de servicio” o the section called “Red del host” para configurar el recurso de puerta de enlace a fin de que utilice direcciones de red de NodePort o host.

Instalación de Cilium Ingress

Consideraciones

- Cilium debe configurarse con `nodePort.enabled` establecido en `true`, como se muestra en los ejemplos siguientes. Si utiliza la característica de reemplazo de kube-proxy de Cilium, no necesita configurar `nodePort.enabled` en `true`.
- Cilium debe configurarse con `envoy.enabled` establecido en `true`, como se muestra en los ejemplos siguientes.
- Si se establece `ingressController.loadbalancerMode` en `dedicated`, Cilium crea servicios dedicados para cada recurso de entrada. Si se establece `ingressController.loadbalancerMode` en `shared`, Cilium crea un servicio compartido del tipo LoadBalancer para todos los recursos de entrada del clúster. Cuando se utiliza el modo de equilibrador de carga `shared`, la configuración del servicio compartido como `labels`, `annotations`, `type` y `loadBalancerIP` se configura en la sección `ingressController.service` de valores de Helm. Consulte la referencia de valores de Helm de Cilium para obtener más información.
- Si se establece `ingressController.default` en `true`, Cilium se configura como el controlador de entrada predeterminado para el clúster y creará entradas incluso cuando no se especifique `ingressClassName` en los recursos de entrada.
- Cilium Ingress se puede implementar en modo de equilibrador de carga (predeterminado), puerto de nodos o red host. Cuando Cilium se instala en el modo de red host, los modos de servicio del tipo LoadBalancer y NodePort están desactivados. Para obtener más información, consulte the section called “Red del host”.
- Establezca siempre `ingressController.service.annotations` como `service.beta.kubernetes.io/aws-load-balancer-type: "external"` en los valores de Helm para evitar que el proveedor de nube de AWS heredado cree un Equilibrador de carga clásico para el servicio de `cilium-ingress` predeterminado creado por el gráfico de Helm de Cilium.

Requisitos previos

1. Si tiene Helm instalado en su entorno de línea de comandos, consulte las instrucciones de configuración de Helm.
2. Se instaló Cilium según las instrucciones de the section called “Cómo configurar una CNI”.

Procedimiento

1. Cree un archivo denominado `cilium-ingress-values.yaml` con el siguiente contenido. En el siguiente ejemplo se configura Cilium Ingress para usar el modo `dedicated` del equilibrador de carga predeterminado y usar un DaemonSet de `cilium-envoy` independiente para los proxies de Envoy configurados para que se ejecuten solo en nodos híbridos.

```
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. Aplique el archivo de valores de Helm al clúster.

```
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. Confirme que el operador, el agente y los pods de Envoy de Cilium estén en ejecución.

```
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
```

Configuración de Cilium Ingress

Cilium Ingress se activa en objetos de entrada mediante la configuración de `ingressClassName` como `cilium`. Los servicios que Cilium crea para los recursos de entrada se pueden configurar con anotaciones en los objetos de entrada cuando se utiliza el modo `dedicated` del equilibrador de carga y en la configuración de Cilium o Helm cuando se utiliza el modo `shared` del equilibrador de carga. Los controladores de entrada suelen utilizar estas anotaciones para configurar la infraestructura del equilibrador de carga u otros atributos del servicio, como el tipo de servicio, el modo del equilibrador de carga, los puertos y los accesos directos a TLS. Las anotaciones clave se describen a continuación. Para obtener una lista completa de las anotaciones admitidas, consulte las anotaciones de Cilium Ingress en la documentación de Cilium.

Anotación	Descripción
`ingress.cilium.io/loadbalancer-mode`	`dedicated` : servicio dedicado del tipo LoadBalancer para cada recurso de entrada (predeterminado). `shared`: servicio único del tipo LoadBalancer para todos los recursos de entrada.
`ingress.cilium.io/service-type`	`LoadBalancer` : el servicio será del tipo LoadBalancer (predeterminado). `NodePort`: el servicio será del tipo NodePort.
`service.beta.kubernetes.io/aws-load-balancer-type`	`"external"` : evite que un proveedor de nube de AWS heredado aprovisione el Equilibrador de carga clásico para servicios del tipo LoadBalancer.

Anotación	Descripción
`lbipam.cilium.io/ips`	Lista de direcciones IP para asignar desde el IPAM LoadBalancer de Cilium

La especificación de Cilium y Kubernetes Ingress admiten reglas de coincidencia exactas, de prefijo y específicas de implementación para las rutas de entrada. Cilium admite las expresiones regulares como regla de coincidencia específica de la implementación. Para obtener más información, consulte Ingress path types and precedence y Path types examples en la documentación de Cilium, así como los ejemplos de la sección the section called “Implementación de Cilium Ingress” de esta página.

A continuación, se muestra un ejemplo de objeto de Cilium Ingress.

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-ingress
  annotations:
    service.beta.kuberentes.io/...
    service.kuberentes.io/...
spec:
  ingressClassName: cilium
  rules:
  ...
```

Implementación de Cilium Ingress

1. Cree una aplicación de muestra. En el siguiente ejemplo, se utiliza la aplicación de microservicios de ejemplo Istio BookInfo.

   ```
   kubectl apply -f https://raw.githubusercontent.com/istio/istio/refs/heads/master/
   samples/bookinfo/platform/kube/bookinfo.yaml
   ```

2. Compruebe que la aplicación se esté ejecutando correctamente.

   ```
   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. Cree un archivo denominado `my-ingress.yaml` con el siguiente contenido. En el ejemplo siguiente se utiliza la anotación `service.beta.kubernetes.io/aws-load-balancer-type: "external"` para evitar que el proveedor de nube de AWS heredado cree un Equilibrador de carga clásico para el servicio del tipo LoadBalancer que Cilium crea para el recurso de entrada.

```
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. Aplique el recurso de entrada al clúster.

```
kubectl apply -f my-ingress.yaml
```

5. Confirme que se hayan creado el recurso de entrada y el servicio correspondiente. En esta fase, se espera que el campo `ADDRESS` del recurso de entrada no se complete con una dirección IP o un nombre de host y que el servicio compartido o dedicado del tipo LoadBalancer para el recurso de entrada tampoco tenga una dirección IP o un nombre de host asignados.

```
kubectl get ingress my-ingress
```

```
NAME         CLASS    HOSTS   ADDRESS   PORTS   AGE
my-ingress   cilium   *                 80      8s
```

Para el modo del equilibrador de carga `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
```

Para el modo del equilibrador de carga `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. Continúe con the section called “LoadBalancer del tipo de servicio” para configurar el recurso de entrada a fin de que utilice una dirección IP asignada por Cilium Load Balancer IPAM, y con the section called “NodePort del tipo de servicio” o the section called “Red del host” para configurar el recurso de entrada a fin de que utilice direcciones de red de NodePort o host.

LoadBalancer del tipo de servicio

Infraestructura del equilibrador de carga existente

De forma predeterminada, tanto para Cilium Ingress como para Cilium Gateway, Cilium crea los servicios de Kubernetes del tipo LoadBalancer para los recursos de entrada o puerta de enlace. Los atributos de los servicios que crea Cilium se pueden configurar mediante los recursos de entrada y puerta de enlace. Al crear recursos de entrada o puerta de enlace, la dirección IP expuesta externamente o los nombres de host de dicha entrada o puerta de enlace se asignan desde la

infraestructura del equilibrador de carga, que se suele aprovisionar mediante un controlador de entrada o puerta de enlace.

Muchos controladores de entrada y puerta de enlace utilizan anotaciones para detectar y configurar la infraestructura del equilibrador de carga. Las anotaciones de estos controladores de entrada y puerta de enlace se configuran en los recursos de entrada o puerta de enlace, como se muestra en los ejemplos anteriores. Consulte la documentación del controlador de entrada o puerta de enlace para ver las anotaciones compatibles, así como la documentación de Kubernetes Ingress y la documentación de Kubernetes Gateway para ver una lista de controladores populares.

> **Important**
>
> Cilium Ingress y Gateway no se pueden usar con el Controlador del equilibrador de carga de AWS y los Equilibradores de carga de red de AWS con Nodos híbridos de EKS. Al intentar usarlos juntos, los destinos no se registran, ya que el NLB intenta conectarse directamente a las IP del pod que respaldan el servicio del tipo LoadBalancer cuando el `target-type` del NLB está configurado en `ip` (requisito para usar el NLB con cargas de trabajo que se ejecutan en Nodos híbridos de EKS).

Sin infraestructura del equilibrador de carga

Si no tienen ninguna infraestructura del equilibrador de carga y el controlador de entrada o puerta de enlace correspondiente en su entorno, los recursos de entrada o puerta de enlace y los servicios correspondientes del tipo LoadBalancer se pueden configurar para utilizar las direcciones IP asignadas por Load Balancer IP address management (LB IPAM) de Cilium. LB IPAM de Cilium se puede configurar con rangos de direcciones IP conocidos de su entorno en las instalaciones y puede utilizar la compatibilidad integrada con el protocolo de puerta de enlace fronteriza (BGP) de Cilium o los anuncios de L2 para anunciar las direcciones IP del LoadBalancer en su red en las instalaciones.

En el siguiente ejemplo se muestra cómo configurar LB IPAM de Cilium con una dirección IP para utilizarla en los recursos de entrada o puerta de enlace, y cómo configurar Cilium BGP Control Plane para anunciar la dirección IP del LoadBalancer en la red en las instalaciones. La característica LB IPAM de Cilium está activada de forma predeterminada, pero no se activa hasta que se crea un recurso `CiliumLoadBalancerIPPool`.

Requisitos previos

- Se ha instalado Cilium Ingress o Gateway según las instrucciones de the section called “Instalación de Cilium Ingress” o the section called “Instalación de Cilium Gateway”.
- Se han implementado los recursos de Cilium Ingress o Gateway con una aplicación de ejemplo según las instrucciones de the section called “Implementación de Cilium Ingress” o the section called “Implementación de Cilium Gateway”.
- Cilium BGP Control Plane está activado según las instrucciones de the section called “Configuración de BGP”. Si no desea utilizar el BGP, puede omitir este requisito previo, pero no podrá acceder a su recurso de entrada o puerta de enlace hasta que la dirección IP del LoadBalancer asignada por LB IPAM de Cilium se pueda enrutar en su red en las instalaciones.

Procedimiento

1. Opcionalmente, parchee el recurso de entrada o puerta de enlace para solicitar una dirección IP específica para usarla en el servicio del tipo LoadBalancer. Si no solicita ninguna dirección IP específica, Cilium asignará una dirección IP del rango de direcciones IP configurado en el recurso `CiliumLoadBalancerIPPool` en el paso siguiente. En los comandos siguientes, sustituya `LB_IP_ADDRESS` por la dirección IP para solicitar el servicio del tipo LoadBalancer.

   Puerta de enlace

   ```
   kubectl patch gateway -n default my-gateway --type=merge -p '{
     "spec": {
       "addresses": [{"type": "IPAddress", "value": "LB_IP_ADDRESS"}]
     }
   }'
   ```

   Ingreso

   ```
   kubectl patch ingress my-ingress --type=merge -p '{
     "metadata": {"annotations": {"lbipam.cilium.io/ips": "LB_IP_ADDRESS"}}
   }'
   ```

2. Cree un archivo llamado `cilium-lbip-pool-ingress.yaml` con un recurso `CiliumLoadBalancerIPPool` para configurar el rango de direcciones IP del equilibrador de carga para los recursos de entrada o puerta de enlace.

- Si utiliza Cilium Ingress, Cilium aplica automáticamente la etiqueta `cilium.io/ingress: "true"` a los servicios que crea para los recursos de entrada. Puede utilizar esta etiqueta en el campo `serviceSelector` de la definición del recurso `CiliumLoadBalancerIPPool` para seleccionar los servicios aptos para LB IPAM.
- Si utiliza Cilium Gateway, puede utilizar la etiqueta `gateway.networking.k8s.io/gateway-name` en los campos `serviceSelector` de la definición del recurso `CiliumLoadBalancerIPPool` para seleccionar los recursos de puerta de enlace aptos para LB IPAM.
- Reemplace `LB_IP_CIDR` por el rango de direcciones IP para utilizar las direcciones IP del equilibrador de carga. Para seleccionar una sola dirección IP, use un CIDR `/32`. Para obtener más información, consulte LoadBalancer IP Address Management en la documentación de Cilium.

```
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. Aplique el recurso `CiliumLoadBalancerIPPool` al clúster.

```
kubectl apply -f cilium-lbip-pool-ingress.yaml
```

4. Confirme que se haya asignado una dirección IP desde LB IPAM de Cilium para el recurso de entrada o puerta de enlace.

   Puerta de enlace

```
kubectl get gateway my-gateway
```

```
NAME         CLASS    ADDRESS         PROGRAMMED   AGE
my-gateway   cilium   LB_IP_ADDRESS     True         6m41s
```

Ingreso

```
kubectl get ingress my-ingress
```

```
NAME         CLASS    HOSTS   ADDRESS         PORTS   AGE
my-ingress   cilium   *       LB_IP_ADDRESS   80      10m
```

5. Cree un archivo llamado `cilium-bgp-advertisement-ingress.yaml` con un recurso `CiliumBGPAdvertisement` para anunciar la dirección IP del LoadBalancer para los recursos de entrada o puerta de enlace. Si no utiliza el BGP de Cilium, puede saltarse este paso. La dirección IP del LoadBalancer utilizada para el recurso de entrada o puerta de enlace debe poder enrutarse en la red en las instalaciones para que pueda consultar el servicio en el siguiente paso.

```
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. Aplique el recurso `CiliumBGPAdvertisement` al clúster.

```
kubectl apply -f cilium-bgp-advertisement-ingress.yaml
```

7. Acceda al servicio mediante la dirección IP asignada desde LB IPAM de Cilium.

```
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"
}
```

NodePort del tipo de servicio

Si no tiene ninguna infraestructura del equilibrador de carga y el controlador de entrada correspondiente en su entorno, o si autoadministra su infraestructura del equilibrador de carga o utiliza un equilibrador de carga basado en DNS, puede configurar Cilium Ingress para crear servicios del tipo NodePort para los recursos de entrada. Cuando se utiliza NodePort con Cilium Ingress, el servicio del tipo NodePort se expone en un puerto de cada nodo en el rango de puertos 30000-32767. En este modo, cuando el tráfico llega a cualquier nodo del clúster de NodePort, se reenvía a un pod que respalda el servicio, que puede estar en el mismo nodo o en un nodo diferente.

> **Note**
>
> Está prevista la compatibilidad de Cilium Gateway con servicios NodePort en la versión 1.18.x (#27273) de Cilium

Requisitos previos

- Se ha instalado Cilium Ingress según las instrucciones de the section called “Instalación de Cilium Ingress”.

- Se han implementado los recursos de Cilium Ingress con una aplicación de ejemplo según las instrucciones de the section called “Implementación de Cilium Ingress”.

Procedimiento

1. Parchee el recurso `my-ingress` de entrada existente para cambiarlo del tipo de servicio LoadBalancer a NodePort.

```
kubectl patch ingress my-ingress --type=merge -p '{
    "metadata": {"annotations": {"ingress.cilium.io/service-type": "NodePort"}}
}'
```

   Si no ha creado el recurso de entrada, puede aplicar la siguiente definición de entrada al clúster para crearlo. Tenga en cuenta que la siguiente definición de entrada utiliza la aplicación de ejemplo Istio BookInfo descrita en the section called “Implementación de Cilium Ingress”.

```
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. Confirme que el servicio del recurso de entrada se haya actualizado para usar el tipo de servicio NodePort. Anote el puerto del protocolo HTTP en la salida. En el siguiente ejemplo, el puerto HTTP es `32353`, que se utilizará en un paso posterior para consultar el servicio. La ventaja de

utilizar Cilium Ingress con un servicio del tipo NodePort es que puede aplicar un enrutamiento basado en rutas y hosts, así como políticas de red para el tráfico de entrada, lo que no puede hacer con un servicio estándar del tipo NodePort sin entrada.

```
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. Obtenga las direcciones IP de los nodos en el clúster.

```
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. Acceda al servicio del tipo NodePort con las direcciones IP de sus nodos y el NodePort capturado anteriormente. En el siguiente ejemplo, la dirección IP del nodo utilizada es `10.80.146.32`, mientras que el NodePort es `32353`. Sustitúyalos por los valores correspondientes al entorno.

```
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"
}
```

Red del host

Al igual que en el servicio del tipo NodePort, si no tiene ninguna infraestructura del equilibrador de carga ni un controlador de entrada o puerta de enlace, o si autoadministra el equilibrio de carga con un equilibrador de carga externo, puede configurar Cilium Ingress y Cilium Gateway para exponer los recursos de entrada y puerta de enlace directamente en la red host. Cuando el modo de red host está activado para un recurso de entrada o puerta de enlace, los modos de servicio del tipo LoadBalancer y NodePort se desactivan automáticamente, el modo de red host se excluye mutuamente con estos modos alternativos para cada recurso de entrada o puerta de enlace. En comparación con el modo del servicio del tipo NodePort, el modo de red host ofrece flexibilidad adicional para el rango de puertos que se pueden usar (no está restringido al rango NodePort 30000-32767) y puede configurar un subconjunto de nodos donde los proxies de Envoy se ejecutan en la red host.

Requisitos previos

- Se ha instalado Cilium Ingress o Gateway según las instrucciones de the section called “Instalación de Cilium Ingress” o the section called “Instalación de Cilium Gateway”.

Procedimiento

Puerta de enlace

1. Cree un archivo llamado `cilium-gateway-host-network.yaml` con el siguiente contenido.

```
gatewayAPI:
  enabled: true
  hostNetwork:
    enabled: true
    # uncomment to restrict nodes where Envoy proxies run on the host network
    # nodes:
    #   matchLabels:
    #     role: gateway
```

2. Aplique la configuración de Cilium Gateway de red host al clúster.

```
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
```

Si no ha creado el recurso de puerta de enlace, puede aplicar la siguiente definición de puerta de enlace al clúster para crearlo. La siguiente definición de puerta de enlace utiliza la aplicación de ejemplo Istio BookInfo descrita en the section called “Implementación de Cilium Gateway”. En el siguiente ejemplo, el recurso de puerta de enlace está configurado para usar el puerto 8111 del oyente HTTP, que es el puerto de oyente compartido para los proxies de Envoy que se ejecutan en la red host. Si utiliza un puerto privilegiado (inferior a 1023) para el recurso de puerta de enlace, consulte la documentación de Cilium para obtener instrucciones.

```
---
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
```

Puede observar la configuración de Envoy de Cilium aplicada con el siguiente comando.

```
kubectl get cec cilium-gateway-my-gateway -o yaml
```

Puede obtener el puerto de oyente de Envoy para el servicio `cilium-gateway-my-gateway` con el siguiente comando. En este ejemplo, el puerto de oyente compartido es `8111`.

```
kubectl get cec cilium-gateway-my-gateway -o jsonpath={.spec.services[0].ports[0]}
```

3. Obtenga las direcciones IP de los nodos en el clúster.

```
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. Acceda al servicio mediante las direcciones IP de sus nodos y el puerto de oyente del recurso `cilium-gateway-my-gateway`. En el siguiente ejemplo, la dirección IP del nodo utilizada es `10.80.146.32`, mientras que el puerto de oyente es `8111`. Sustitúyalos por los valores correspondientes al entorno.

```
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

Debido a un problema de Cilium con el proceso ascendente (#34028), Cilium Ingress en modo de red host requiere el uso de `loadbalancerMode: shared`, que crea un solo servicio del tipo ClusterIP para todos los recursos de entrada en el clúster. Si utiliza un puerto privilegiado (inferior a 1023) para el recurso de entrada, consulte la documentación de Cilium para obtener instrucciones.

1. Cree un archivo llamado `cilium-ingress-host-network.yaml` con el siguiente contenido.

```
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. Aplique la configuración de Cilium Ingress de red host al clúster.

```
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
```

Si no ha creado el recurso de entrada, puede aplicar la siguiente definición de entrada al clúster para crearlo. La siguiente definición de entrada utiliza la aplicación de ejemplo Istio BookInfo descrita en the section called “Implementación de Cilium Ingress”.

```
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
```

Puede observar la configuración de Envoy de Cilium aplicada con el siguiente comando.

```
kubectl get cec -n kube-system cilium-ingress -o yaml
```

Puede obtener el puerto de oyente de Envoy para el servicio `cilium-ingress` con el siguiente comando. En este ejemplo, el puerto de oyente compartido es `8111`.

```
kubectl get cec -n kube-system cilium-ingress -o
 jsonpath={.spec.services[0].ports[0]}
```

3. Obtenga las direcciones IP de los nodos en el clúster.

```
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. Acceda al servicio mediante las direcciones IP de sus nodos y el `sharedListenerPort` del recurso `cilium-ingress`. En el siguiente ejemplo, la dirección IP del nodo utilizada es `10.80.146.32`, mientras que el puerto de oyente es `8111`. Sustitúyalos por los valores correspondientes al entorno.

```
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"
}
```

Configuración de servicios del tipo LoadBalancer para nodos híbridos

En este tema se describe cómo configurar el equilibrio de carga de capa 4 (L4) para las aplicaciones que se ejecutan en Nodos híbridos de Amazon EKS. Los servicios de Kubernetes del tipo LoadBalancer se utilizan para exponer las aplicaciones de Kubernetes externas al clúster. Los servicios del tipo LoadBalancer se suelen utilizar con la infraestructura del equilibrador de carga físico en la nube o en un entorno en las instalaciones para atender el tráfico de la carga de trabajo.

Esta infraestructura de equilibrador de carga suele aprovisionarse con un controlador específico del entorno.

AWS es compatible con Equilibradores de carga de red de AWS y Cilium para servicios del tipo LoadBalancer que se ejecutan en Nodos híbridos de EKS. La decisión de utilizar NLB o Cilium se basa en el origen del tráfico de aplicaciones. Si el tráfico de aplicaciones se origina en una región de AWS, AWS recomienda utilizar el NLB de AWS y el Controlador del equilibrador de carga de AWS. Si el tráfico de aplicaciones se origina en un entorno local en las instalaciones o en la periferia, AWS recomienda utilizar las funciones de equilibrio de carga integradas de Cilium, que se pueden utilizar con o sin una infraestructura de equilibradores de carga en su entorno.

Para obtener información sobre el equilibrio de carga de tráfico de aplicaciones de capa 7 (L7), consulte the section called “Configuración de Ingress”. Para obtener información general sobre el equilibrio de carga con EKS, consulte las prácticas recomendadas para el equilibrio de carga.

Equilibrador de carga de red de AWS

Puede usar el Controlador del equilibrador de carga de AWS y el NLB con el tipo de destino `ip` para cargas de trabajo que se ejecutan en nodos híbridos. Cuando se utiliza el tipo de destino `ip`, el NLB reenvía el tráfico directamente a los pods, sin pasar por la ruta de red de la capa de servicio. Para que el NLB llegue a los destinos de IP de pods en nodos híbridos, los CIDR de pods en las instalaciones deben ser enrutables en la red en las instalaciones. Además, el Controlador del equilibrador de carga de AWS utiliza webhooks y requiere una comunicación directa desde el plano de control de EKS. Para obtener más información, consulte the section called “Configuración de webhooks”.

- Consulte the section called “Equilibrio de carga de red” para ver los requisitos de configuración de la subred, así como the section called “Instalación con Helm” y Best Practices for Load Balancing para obtener información adicional sobre el Equilibrador de carga de red de AWS y el Controlador del equilibrador de carga de AWS.
- Consulte AWS Load Balancer Controller NLB configurations para obtener configuraciones que se pueden aplicar a servicios del tipo `LoadBalancer` con el Equilibrador de carga de red de AWS.

Requisitos previos

- Se instaló Cilium según las instrucciones de the section called “Cómo configurar una CNI”.
- Cilium BGP Control Plane está activado según las instrucciones de the section called “Configuración de BGP”. Si no desea usar el BGP, debe usar un método alternativo para que los

CIDR de pods en las instalaciones sean enrutables dentro de la red en las instalaciones. Consulte the section called “CIDR de pod remoto enrutable” para obtener más información.

- Si tiene Helm instalado en su entorno de línea de comandos, consulte las instrucciones de configuración de Helm.
- Si tiene eksctl instalado en su entorno de línea de comandos, consulte las instrucciones de configuración de eksctl.

Procedimiento

1. Descargue una política de IAM para el Controlador del equilibrador de carga de AWS que le permita realizar llamadas a las API de AWS en su nombre.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-
controller/refs/heads/main/docs/install/iam_policy.json
```

2. Cree una política de IAM con la política descargada en el paso anterior.

```
aws iam create-policy \
    --policy-name AWSLoadBalancerControllerIAMPolicy \
    --policy-document file://iam_policy.json
```

3. Sustituya los valores del nombre del clúster (`CLUSTER_NAME`), la región de AWS (`AWS_REGION`) y el ID de la cuenta de AWS (`AWS_ACCOUNT_ID`) por su configuración y ejecute el siguiente comando.

```
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. Agregue el repositorio de gráficos de Helm eks-charts. AWS mantiene este repositorio en GitHub.

```
helm repo add eks https://aws.github.io/eks-charts
```

5. Actualice el repositorio de Helm local para asegurarse de que cuenta con los gráficos más recientes.

```
helm repo update eks
```

6. Instale el Controlador del equilibrador de carga de AWS. Sustituya los valores del nombre del clúster (`CLUSTER_NAME`), la región de AWS (`AWS_REGION`), el ID de VPC (`VPC_ID`) y la versión del gráfico de Helm del Controlador del equilibrador de carga de AWS (`AWS_LBC_HELM_VERSION`) por su configuración. Para encontrar la versión más reciente del gráfico de Helm, ejecute `helm search repo eks/aws-load-balancer-controller --versions`. Si ejecuta un clúster en modo mixto con nodos híbridos y nodos en la nube de AWS, puede ejecutar el Controlador del equilibrador de carga de AWS en los nodos en la nube según las instrucciones de the section called “Controlador del equilibrador de carga de AWS”.

```
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. Compruebe que el Controlador del equilibrador de carga de AWS se haya instalado correctamente.

```
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. Defina una aplicación de ejemplo en un archivo denominado `tcp-sample-app.yaml`. El siguiente ejemplo usa una implementación simple de NGINX con un puerto TCP.

```
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. Aplique la implementación al clúster.

```
kubectl apply -f tcp-sample-app.yaml
```

10. Defina un servicio del tipo LoadBalancer para la implementación en un archivo llamado `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. Aplique la configuración del servicio al clúster.

```
kubectl apply -f tcp-sample-service.yaml
```

12 El aprovisionamiento del NLB para el servicio puede tardar unos minutos. Una vez aprovisionado el NLB, el servicio tendrá asignada una dirección que se corresponderá con el nombre de DNS de la implementación del NLB.

```
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 Acceda al servicio con la dirección del NLB.

```
curl k8s-default-tcpsampl-xxxxxxxxxx-xxxxxxxxxxxxxxxx.elb.<region>.amazonaws.com
```

A continuación se muestra un ejemplo de salida.

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
[...]
```

14 Limpie los recursos de que ha creado.

```
kubectl delete -f tcp-sample-service.yaml
kubectl delete -f tcp-sample-app.yaml
```

Equilibrio de carga en el clúster de Cilium

Cilium se puede utilizar como un equilibrador de cargas integrado en el clúster para las cargas de trabajo que se ejecutan en Nodos híbridos de EKS, lo que puede resultar útil para entornos que no tienen una infraestructura de equilibrador de carga. Las capacidades de equilibrio de carga de Cilium constan de una combinación de características de Cilium, que incluyen la sustitución de kube-proxy, Load Balancer IP Address Management (IPAM) y BGP Control Plane. Las responsabilidades de estas características se detallan a continuación:

- Sustitución del kube-proxy de Cilium: gestiona el enrutamiento del tráfico del servicio a los pods de backend.
- Cilium Load Balancer IPAM: administra las direcciones IP que se pueden asignar a servicios del tipo `LoadBalancer`.
- Cilium BGP Control Plane: anuncia direcciones IP asignadas por Load Balancer IPAM a la red en las instalaciones.

Si no utiliza la sustitución de kube-proxy de Cilium, puede seguir utilizando Cilium Load Balancer IPAM y BGP Control Plane para asociar y asignar direcciones IP para los servicios del tipo LoadBalancer. Si no utilizas la sustitución de kube-proxy de Cilium, el equilibrio de carga para los servicios y los pods de backend se gestiona mediante reglas de kube-proxy e iptables de forma predeterminada en EKS.

Requisitos previos

- Cilium se instala según las instrucciones de the section called “Cómo configurar una CNI” con o sin la sustitución de kube-proxy activada. La sustitución de kube-proxy de Cilium requiere ejecutar un sistema operativo con un kernel de Linux al menos tan reciente como las versiones 4.19.57, 5.1.16 o 5.2.0. Todas las versiones recientes de los sistemas operativos compatibles para su uso con nodos híbridos cumplen estos requisitos, con la excepción de Red Hat Enterprise Linux (RHEL) 8.x.
- Cilium BGP Control Plane está activado según las instrucciones de the section called “Configuración de BGP”. Si no desea usar el BGP, debe usar un método alternativo para que los CIDR de pods en las instalaciones sean enrutables dentro de la red en las instalaciones. Consulte the section called “CIDR de pod remoto enrutable” para obtener más información.
- Si tiene Helm instalado en su entorno de línea de comandos, consulte las instrucciones de configuración de Helm.

Procedimiento

1. Cree un archivo llamado `cilium-lbip-pool-loadbalancer.yaml` con un recurso `CiliumLoadBalancerIPPool` para configurar el rango de direcciones IP del equilibrador de carga para los servicios del tipo LoadBalancer.
   - Reemplace `LB_IP_CIDR` por el rango de direcciones IP para utilizar las direcciones IP del equilibrador de carga. Para seleccionar una sola dirección IP, use un CIDR `/32`. Para obtener

más información, consulte LoadBalancer IP Address Management en la documentación de Cilium.

- El campo `serviceSelector` está configurado para que coincida con el nombre del servicio que creará en un paso posterior. Con esta configuración, las IP de este grupo solo se asignarán a los servicios con el nombre `tcp-sample-service`.

```
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. Aplique el recurso `CiliumLoadBalancerIPPool` al clúster.

```
kubectl apply -f cilium-lbip-pool-loadbalancer.yaml
```

3. Confirme que haya al menos una dirección IP disponible en el grupo.

```
kubectl get ciliumloadbalancerippools.cilium.io
```

```
NAME               DISABLED   CONFLICTING   IPS AVAILABLE   AGE
tcp-service-pool   false      False         1               24m
```

4. Cree un archivo llamado `cilium-bgp-advertisement-loadbalancer.yaml` con un recurso `CiliumBGPAdvertisement` para anunciar la dirección IP del equilibrador de carga para el servicio que creará en el siguiente paso. Si no utiliza el BGP de Cilium, puede saltarse este paso. La dirección IP del equilibrador de carga utilizada para el servicio debe poder enrutarse en la red en las instalaciones para que pueda consultar el servicio en el paso final.

   - El campo `advertisementType` está configurado como `Service` y `service.addresses` está configurado como `LoadBalancerIP` para anunciar solo `LoadBalancerIP` para los servicios del tipo `LoadBalancer`.
   - El campo `selector` está configurado para que coincida con el nombre del servicio que creará en un paso posterior. Con esta configuración, solo se anunciará `LoadBalancerIP` para servicios con el nombre `tcp-sample-service`.

```
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. Aplique el recurso `CiliumBGPAdvertisement` al clúster. Si no utiliza el BGP de Cilium, puede saltarse este paso.

```
kubectl apply -f cilium-bgp-advertisement-loadbalancer.yaml
```

6. Defina una aplicación de ejemplo en un archivo denominado `tcp-sample-app.yaml`. El siguiente ejemplo usa una implementación simple de NGINX con un puerto TCP.

```
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. Aplique la implementación al clúster.

```
kubectl apply -f tcp-sample-app.yaml
```

8. Defina un servicio del tipo LoadBalancer para la implementación en un archivo llamado `tcp-sample-service.yaml`.
   - Puede solicitar una dirección IP específica del grupo de IP del equilibrador de carga con la anotación `lbipam.cilium.io/ips` en el objeto del servicio. Puede eliminar esta anotación si no desea solicitar una dirección IP específica para el servicio.
   - El campo de especificaciones `loadBalancerClass` es obligatorio para evitar que el proveedor de nube de AWS heredado cree un Equilibrador de carga clásico para el servicio. En el siguiente ejemplo, está configurado como `io.cilium/bgp-control-plane` para usar BGP Control Plane de Cilium como clase del equilibrador de carga. Como alternativa, este campo se puede configurar como `io.cilium/l2-announcer` para utilizar la característica L2 Announcements de Cilium (actualmente en versión beta y no compatible oficialmente con AWS).

```
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. Aplique el servicio al clúster. El servicio se creará con una dirección IP externa que podrá utilizar para acceder a la aplicación.

```
kubectl apply -f tcp-sample-service.yaml
```

10. Compruebe que el servicio se haya creado correctamente y que tenga asignada una IP diferente al `CiliumLoadBalancerIPPool` creado en el paso anterior.

```
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. Si utiliza Cilium en el modo de sustitución de kube-proxy, puede confirmar que Cilium se encarga del equilibrio de carga del servicio mediante la ejecución del siguiente comando. En el siguiente resultado, las direcciones `10.86.2.x` son las direcciones IP de los pods de backend del servicio.

```
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. Confirme que Cilium anuncie la dirección IP en la red en las instalaciones mediante BGP. En el siguiente ejemplo, hay cinco nodos híbridos, cada uno de los cuales anuncia `LB_IP_ADDRESS` del servicio `tcp-sample-service` en la red en las instalaciones.

```
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 Acceda al servicio con la dirección IP del equilibrador de carga asignada.

```
curl LB_IP_ADDRESS
```

A continuación se muestra un ejemplo de salida.

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
[...]
```

14 Limpie los recursos de que ha creado.

```
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
```

Configuración de políticas de red de Kubernetes para nodos híbridos

AWS admite las políticas de red de Kubernetes (capa 3 o 4) para el tráfico de entrada y salida de los pods cuando se utiliza Cilium como CNI con Nodos híbridos EKS. Si ejecuta clústeres de EKS con nodos en la nube de AWS, AWS es compatible con las políticas de red de la CNI de Amazon VPC para Kubernetes.

En este tema se explica cómo configurar las políticas de red de Cilium y Kubernetes con Nodos híbridos de EKS. Para obtener información detallada sobre las políticas de red de Kubernetes, consulte Kubernetes Network Policies en la documentación de Kubernetes.

Configurar políticas de red

Consideraciones

- AWS es compatible con las políticas de red ascendentes de Kubernetes y la especificación para la entrada y salida de pods. AWS actualmente no es compatible con `CiliumNetworkPolicy` ni `CiliumClusterwideNetworkPolicy`.
- El valor de Helm `policyEnforcementMode` se puede utilizar para controlar el comportamiento de aplicación predeterminado de las políticas de Cilium. El comportamiento predeterminado permite todo el tráfico de entrada y salida. Cuando una política de red selecciona un punto de conexión, pasa a un estado de denegación predeterminado, en el que solo se permite el tráfico explícitamente permitido. Consulte la documentación de Cilium para obtener más información sobre el modo de políticas predeterminados y los modos de aplicación de políticas.
- Si cambia `policyEnforcementMode` para una instalación de Cilium existente, debe reiniciar el DaemonSet del agente de Cilium para aplicar el nuevo modo de aplicación de políticas.
- Utilice `namespaceSelector` y `podSelector` para permitir o denegar el tráfico hacia o desde los espacios de nombres y los pods con etiquetas coincidentes. `namespaceSelector` y `podSelector` se pueden usar con `matchLabels` o `matchExpressions` para seleccionar espacios de nombres y pods en función de sus etiquetas.
- Utilice `ingress.ports` y `egress.ports` para permitir o denegar el tráfico hacia o desde puertos y protocolos.
- El campo `ipBlock` no se puede utilizar para permitir o denegar de forma selectiva el tráfico hacia o desde las direcciones IP de los pods (#9209). El uso de selectores `ipBlock` para las IP de los nodos es una característica en versión beta de Cilium y no es compatible con AWS.
- Consulte NetworkPolicy resource en la documentación de Kubernetes para obtener información sobre los campos disponibles para las políticas de red de Kubernetes.

Requisitos previos

- Se instaló Cilium según las instrucciones de the section called “Cómo configurar una CNI”.
- Si tiene Helm instalado en su entorno de línea de comandos, consulte las instrucciones de configuración de Helm.

Procedimiento

El siguiente procedimiento configura las políticas de red para una aplicación de microservicios de ejemplo, de modo que los componentes solo pueden comunicarse con otros componentes necesarios para que la aplicación funcione. El procedimiento utiliza la aplicación de microservicios de ejemplo Istio BookInfo.

La aplicación BookInfo consta de cuatro microservicios independientes con las siguientes relaciones:

- productpage. El microservicio productpage llama a los microservicios details y reviews para rellenar la página.
- details. El microservicio details contiene información sobre libros.
- reviews. El microservicio reviews contiene reseñas de libros. También llama al microservicio ratings.
- ratings. El microservicio ratings contiene información de clasificación de libros que acompaña a la reseña de un libro.

1. Cree la aplicación de ejemplo.

```
kubectl apply -f https://raw.githubusercontent.com/istio/istio/refs/heads/master/
samples/bookinfo/platform/kube/bookinfo.yaml
```

2. Confirme que la aplicación se esté ejecutando correctamente y anote la dirección IP del pod del microservicio productpage. Utilizará la dirección IP de este pod para consultar cada microservicio en los pasos siguientes.

```
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. Cree un pod que se utilizará en todo momento para probar las políticas de red. Tenga en cuenta que el pod se crea en el espacio de nombres `default` con la etiqueta `access: true`.

```
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. Pruebe el acceso al microservicio productpage. En el siguiente ejemplo, utilizamos la dirección IP del pod productpage (`10.86.2.193`) para consultar el microservicio. Sustitúyala por la dirección IP del pod productpage de su entorno.

```
curl -s http://10.86.2.193:9080/productpage | grep -o "<title>.*</title>"
```

```
<title>Simple Bookstore App</title>
```

5. Para salir del pod curl de prueba, puede escribir `exit`. Para volver a conectarse al pod, ejecute el siguiente comando.

```
kubectl attach curl-pod -c curl-pod -i -t
```

6. Para demostrar los efectos de las políticas de red en los siguientes pasos, primero creamos una política de red que deniegue todo el tráfico de los microservicios de BookInfo. Cree un archivo llamado `network-policy-deny-bookinfo.yaml` que defina la política de red de denegación.

```
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. Aplique la política de red de denegación al clúster.

```
kubectl apply -f network-policy-default-deny-bookinfo.yaml
```

8. Pruebe el acceso a la aplicación BookInfo. En el siguiente ejemplo, utilizamos la dirección IP del pod productpage (`10.86.2.193`) para consultar el microservicio. Sustitúyala por la dirección IP del pod productpage de su entorno.

```
curl http://10.86.2.193:9080/productpage --max-time 10
```

```
curl: (28) Connection timed out after 10001 milliseconds
```

9. Cree un archivo llamado `network-policy-productpage.yaml` que defina la política de red de productpage. La política tiene las siguientes reglas:
   - Permite la entrada de tráfico desde los pods con la etiqueta `access: true` (el pod curl creado en el paso anterior).
   - Permite el tráfico TCP de salida en el puerto `9080` para los microservicios details, reviews y ratings.
   - Permite el tráfico TCP/UDP de salida en el puerto `53` para CoreDNS que se ejecuta en el espacio de nombres `kube-system`.

```
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. Aplique la política de red de productpage a su clúster.

```
kubectl apply -f network-policy-productpage.yaml
```

11. Conéctese al pod curl y pruebe el acceso a la aplicación BookInfo. Ahora se permite el acceso al microservicio productpage, pero se sigue denegando el acceso a los demás microservicios porque siguen sujetos a la política de red de denegación. En los siguientes ejemplos, utilizamos la dirección IP del pod productpage (`10.86.2.193`) para consultar el microservicio. Sustitúyala por la dirección IP del pod productpage de su entorno.

```
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. Cree un archivo llamado `network-policy-details.yaml` que defina la política de red de details. La política solo permite la entrada de tráfico desde el microservicio productpage.

```
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. Cree un archivo llamado `network-policy-reviews.yaml` que defina la política de red de reviews. La política solo permite el tráfico de entrada desde el microservicio productpage y solo el tráfico de salida hacia el microservicio ratings y 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. Cree un archivo llamado `network-policy-ratings.yaml` que defina la política de red de ratings. La política solo permite la entrada de tráfico desde los microservicios productpage y reviews.

```
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. Aplique al clúster las políticas de red de details, reviews y ratings.

```
kubectl apply -f network-policy-details.yaml
kubectl apply -f network-policy-reviews.yaml
kubectl apply -f network-policy-ratings.yaml
```

16. Conéctese al pod curl y pruebe el acceso a la aplicación BookInfo. En los siguientes ejemplos, utilizamos la dirección IP del pod productpage (`10.86.2.193`) para consultar el microservicio. Sustitúyala por la dirección IP del pod productpage de su entorno.

```
kubectl attach curl-pod -c curl-pod -i -t
```

Pruebe el microservicio 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"}
```

Pruebe el microservicio 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."}]}
```

Pruebe el microservicio ratings.

```
curl -s http://10.86.2.193:9080/api/v1/products/1/ratings
```

```
{"id": 1, "ratings": {"Reviewer1": 5, "Reviewer2": 4}}
```

17 Limpie los recursos que ha creado en este procedimiento.

```
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
```

Conceptos de nodos híbridos

Con los nodos híbridos de Amazon EKS, puede unir máquinas físicas o virtuales que se ejecutan en entornos en las instalaciones o periféricos a clústeres de Amazon EKS que se ejecutan en la nube de AWS. Este enfoque aporta muchas ventajas, pero también introduce nuevos conceptos y arquitecturas de red para quienes estén familiarizados con la ejecución de clústeres de Kubernetes en un único entorno de red.

En las siguientes secciones, se profundiza en los conceptos de Kubernetes y redes para los nodos híbridos de EKS y se detalla cómo fluye el tráfico a través de la arquitectura híbrida. Para comprender estas secciones, es necesario que tenga los conocimientos básicos de redes de Kubernetes, como los conceptos de pods, nodos, servicios, plano de control de Kubernetes, kubelet y kube-proxy.

Recomendamos leer estas páginas en orden, empezando por the section called “Conceptos relacionados con las redes”, luego por the section called “Conceptos de Kubernetes” y, por último, por the section called “Flujos de tráfico”.

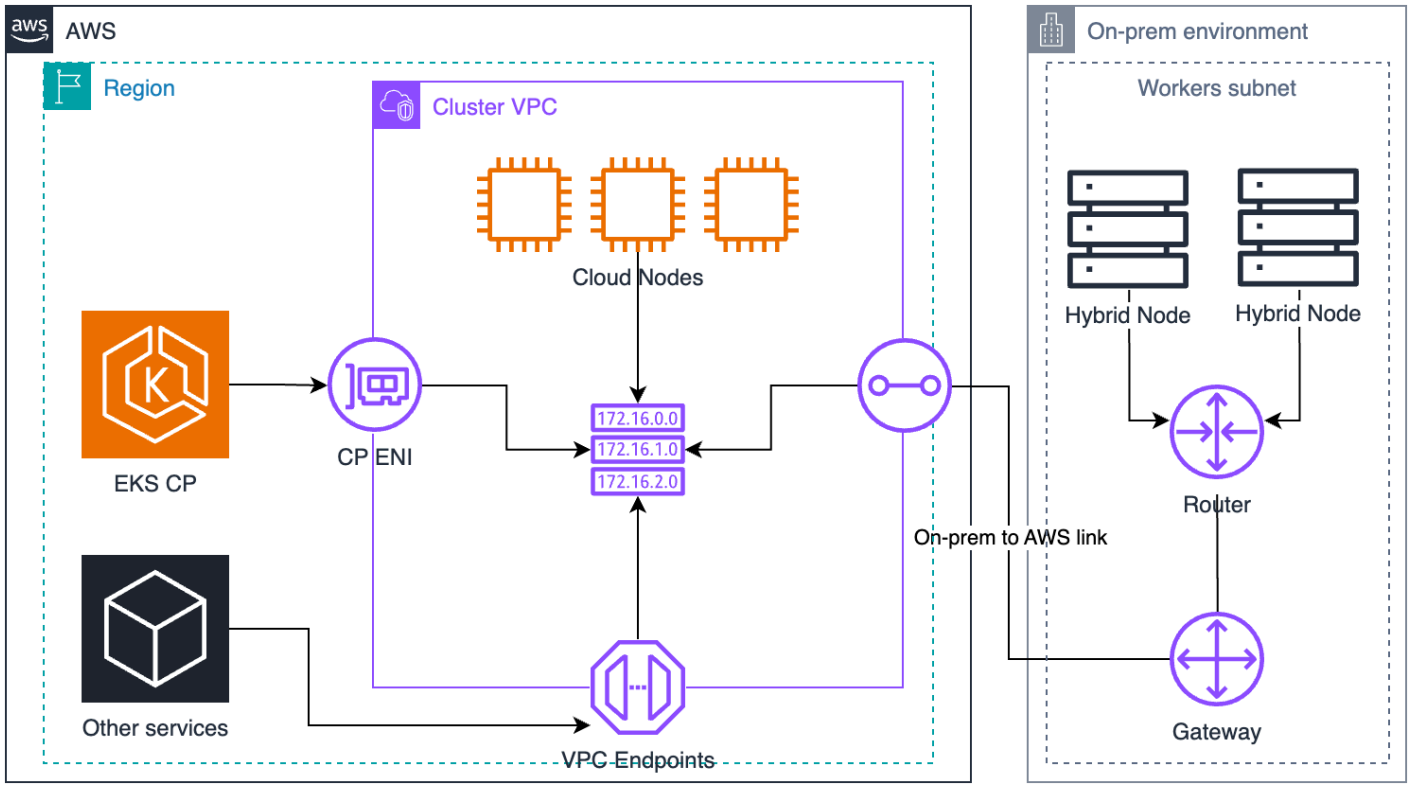
Temas

- Conceptos relacionados con las redes para nodos híbridos
- Conceptos de Kubernetes para nodos híbridos
- Flujos de tráfico de red para nodos híbridos

Conceptos relacionados con las redes para nodos híbridos

En esta sección, se detallan los conceptos básicos sobre redes y las restricciones que debe tener en cuenta al diseñar la topología de red para los nodos híbridos de EKS.

Conceptos relacionados con las redes para los nodos híbridos de EKS



VPC como hub de red

Todo el tráfico que cruza los límites de la nube pasa por su VPC. Esto incluye el tráfico entre el plano de control de EKS o los pods que se ejecutan en AWS hasta los nodos híbridos o los pods que se ejecutan en ellos. Puede pensar en la VPC de su clúster como el hub de red entre los nodos híbridos y el resto del clúster. Esta arquitectura le proporciona un control total del tráfico y su enrutamiento, pero también le confiere la responsabilidad de configurar correctamente las rutas, los grupos de seguridad y los firewalls para la VPC.

Del plano de control de EKS a la VPC

El plano de control de EKS conecta las interfaces de red elástica (ENI) a su VPC. Estas ENI gestionan el tráfico hacia y desde el servidor de la API de EKS. Usted controla la ubicación de las ENI del plano de control de EKS al configurar el clúster, ya que EKS conecta las ENI a las subredes por las que pasa durante la creación del clúster.

EKS asocia los grupos de seguridad a las ENI que EKS conecta a las subredes. Estos grupos de seguridad permiten el tráfico hacia y desde el plano de control de EKS. Esto es importante para los nodos híbridos de EKS porque debe permitir el tráfico de los nodos híbridos y los pods que se ejecutan en ellos hacia las ENI del plano de control de EKS.

Redes de nodos remotos

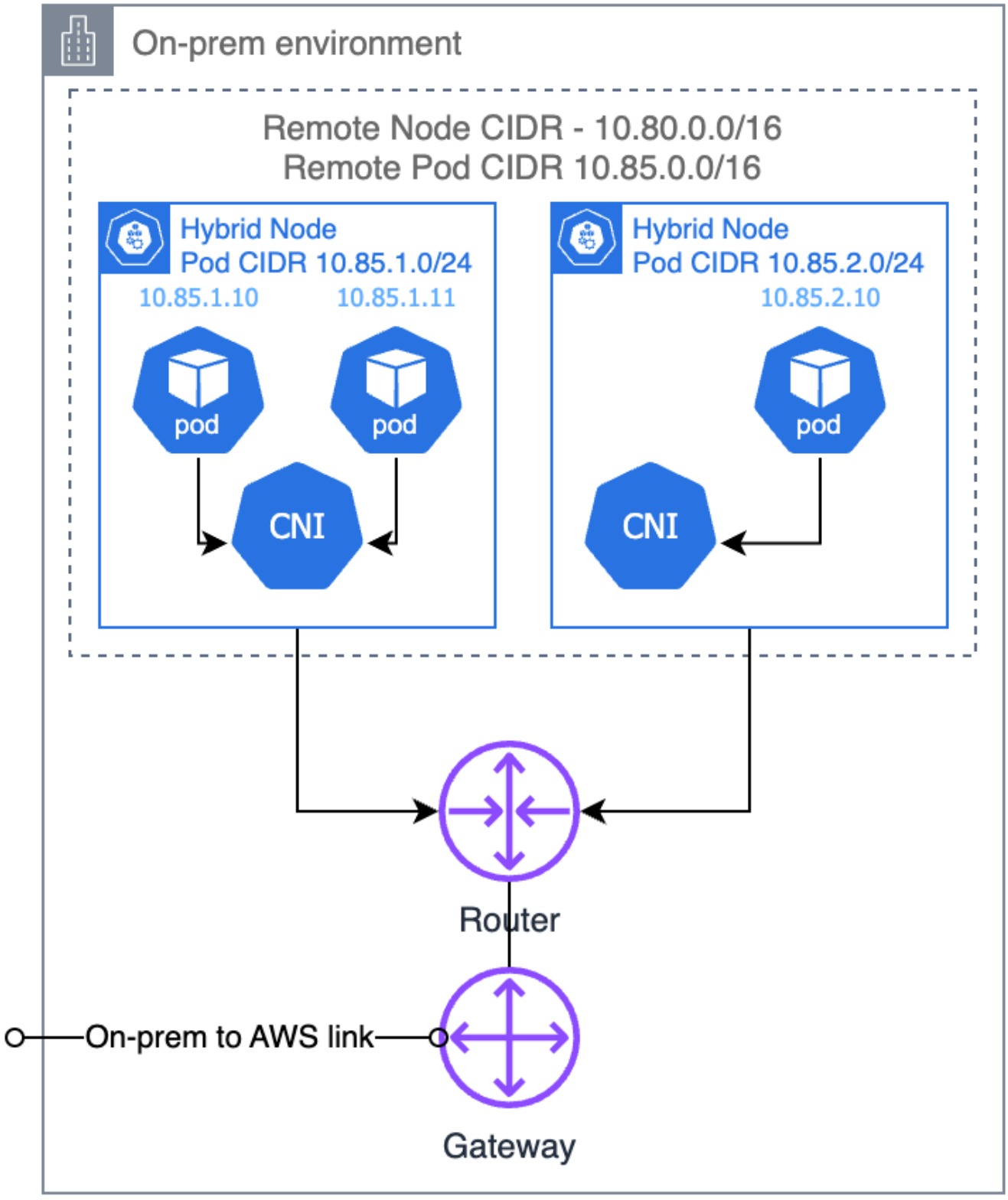
Las redes de nodos remotos, específicamente los CIDR de nodos remotos, son los rangos de IP asignados a las máquinas que se utilizan como nodos híbridos. Cuando aprovisiona nodos híbridos, residen en su centro de datos en las instalaciones o ubicación periférica, que es un dominio de red diferente al del plano de control de EKS y la VPC. Cada nodo híbrido tiene una o varias direcciones IP de un CIDR de nodo remoto que es distinta de las subredes de la VPC.

Usted configura el clúster de EKS con estos CIDR de nodos remotos para que EKS sepa enrutar todo el tráfico destinado a las IP de los nodos híbridos a través de la VPC de su clúster, como las solicitudes a la API de kubelet. Las conexiones a la API `kubelet` se utilizan en los comandos `kubectl attach`, `kubectl cp`, `kubectl exec`, `kubectl logs` y `kubectl port-forward`.

Redes de pods remotos

Las redes de pods remotos son los rangos de IP asignados a los pods que se ejecutan en los nodos híbridos. Por lo general, se configura la CNI con estos rangos, y la funcionalidad del administrador de direcciones IP (IPAM) de la CNI se encarga de asignar un segmento de estos rangos a cada nodo híbrido. Al crear un pod, la CNI asigna una IP al pod desde el segmento asignado al nodo en el que se programó el pod.

El clúster de EKS se configura con estos CIDR de pod remotos para que el plano de control de EKS sepa cómo enrutar todo el tráfico destinado a los pods que se ejecutan en los nodos híbridos a través de la VPC del clúster, por ejemplo, la comunicación con webhooks.



Desde las instalaciones a la VPC

La red en las instalaciones que utilice para los nodos híbridos debe enrutarse a la VPC que utilice para el clúster de EKS. Existen varias opciones de conectividad desde la red a la VPC de Amazon disponibles para conectar la red en las instalaciones a una VPC. También puede utilizar su propia solución de VPN.

Es importante que configure el enrutamiento correctamente en el lado de la nube de AWS, en la VPC y en la red en las instalaciones, de modo que ambas redes enruten el tráfico correcto a través de la conexión de las dos redes.

En la VPC, todo el tráfico que va a las redes del nodo remoto y del pod remoto debe enrutarse a través de la conexión a la red en las instalaciones (denominada “puerta de enlace”). Si algunas de sus subredes tienen tablas de enrutamiento diferentes, debe configurar cada una de ellas con las rutas de los nodos híbridos y los pods que se ejecutan en ellos. Esto es válido para las subredes a las que están conectadas las ENI del plano de control de EKS y para las subredes que contienen nodos o pods de EC2 que deben comunicarse con los nodos híbridos.

En la red en las instalaciones, debe configurar la red para permitir el tráfico hacia y desde la VPC del clúster de EKS y los demás servicios de AWS necesarios para los nodos híbridos. El tráfico del clúster de EKS atraviesa la puerta de enlace en ambas direcciones.

Limitaciones de las redes

Red totalmente enrutada

La principal limitación es que el plano de control de EKS y todos los nodos, ya sean nodos en la nube o híbridos, deben formar una red totalmente enrutada. Esto significa que todos los nodos deben poder comunicarse entre sí en la capa tres, mediante la dirección IP.

El plano de control de EKS y los nodos de la nube ya son accesibles entre sí porque se encuentran en una red plana (la VPC). Sin embargo, los nodos híbridos se encuentran en un dominio de red diferente. Por este motivo, debe configurar un enrutamiento adicional en la VPC y en la red en las instalaciones para enrutar el tráfico entre los nodos híbridos y el resto del clúster. Si se puede acceder a los nodos híbridos entre sí y desde la VPC, los nodos híbridos pueden estar en una sola red plana o en varias redes segmentadas.

CIDR de pod remoto enrutable

Para que el plano de control de EKS se comunique con los pods que se ejecutan en nodos híbridos (por ejemplo, webhooks o el servidor de métricas) o para que los pods que se ejecutan en nodos

de la nube se comuniquen con los pods que se ejecutan en nodos híbridos (comunicación entre la carga de trabajo de este a oeste), el CIDR del pod remoto debe poder enrutarse desde la VPC. Esto significa que la VPC debe poder enrutar el tráfico de los CIDR del pod a través de la puerta de enlace a la red en las instalaciones y que la red local debe poder enrutar el tráfico de un pod al nodo correcto.

Es importante tener en cuenta la distinción entre los requisitos de enrutamiento de los pods en la VPC y en las instalaciones. La VPC solo necesita saber que todo el tráfico que vaya a un pod remoto debe pasar por la puerta de enlace. Si solo tiene un CIDR de pod remoto, solo necesita una ruta.

Este requisito se aplica a todos los saltos de la red en las instalaciones hasta el enrutador local en la misma subred que los nodos híbridos. Este es el único enrutador que debe conocer el segmento CIDR del pod asignado a cada nodo para asegurarse de que el tráfico de un pod en particular llegue al nodo en el que se programó el pod.

Si bien no es necesario, puede optar por propagar estas rutas para los CIDR de los pods en las instalaciones desde su enrutador en las instalaciones hasta las tablas de enrutamiento de la VPC. Aunque no es común que suceda, si los CIDR de los pods en las instalaciones cambian con frecuencia y las tablas de enrutamiento de la VPC deben actualizarse para reflejar los cambios en los CIDR de los pods, le recomendamos que propague los CIDR de los pods en las instalaciones a las tablas de enrutamiento de la VPC.

Tenga en cuenta que la restricción para hacer que los CIDR de los pods en las instalaciones sean enrutables es opcional. Si no necesita ejecutar webhooks en los nodos híbridos ni hacer que los pods de los nodos en la nube se comuniquen con los pods de los nodos híbridos, no es necesario configurar el enrutamiento de los CIDR de los pods de la red en las instalaciones.

¿Por qué los CIDR de los pods en las instalaciones deben poder enrutarse con nodos híbridos?

Al usar EKS con la CNI de la VPC para los nodos en la nube, la CNI de la VPC asigna las IP directamente de la VPC a los pods. Esto significa que no es necesario ningún enrutamiento especial, ya que tanto los pods en la nube como el plano de control de EKS pueden acceder directamente a las IP de los pods.

Cuando se ejecuta en las instalaciones (y con otros CNI en la nube), los pods se suelen ejecutar dentro de una red superpuesta aislada, y el CNI se encarga de suministrar el tráfico entre pods. Por lo general, esto se hace mediante encapsulación: la CNI convierte el tráfico de pod a pod en tráfico de nodo a nodo y se encarga de encapsular y desencapsular ambos extremos. De esta manera, no se requiere configuración adicional en los nodos ni en los enrutadores.

La red con nodos híbridos es única porque presenta una combinación de ambas topologías: el plano de control de EKS y los nodos en la nube (con la CNI de la VPC) esperan una red plana que incluya nodos y pods, mientras que los pods que se ejecutan en nodos híbridos están en una red superpuesta mediante VXLAN para la encapsulación (de forma predeterminada en Cilium). Los pods que se ejecutan en nodos híbridos pueden llegar al plano de control de EKS y a los pods que se ejecutan en nodos en la nube, siempre que la red en las instalaciones pueda enrutarse a la VPC. Sin embargo, sin el enrutamiento de los CIDR de los pods en la red en las instalaciones, cualquier tráfico que regrese a la IP de un pod en las instalaciones se eliminará eventualmente si la red no sabe cómo llegar a la red superpuesta y enrutarse a los nodos correctos.

Conceptos de Kubernetes para nodos híbridos

En esta página, se detallan los conceptos clave de Kubernetes que sustentan la arquitectura del sistema de nodos híbridos de EKS.

Plano de control de EKS en la VPC

Las IP de las ENI del plano de control de EKS se almacenan en el objeto `kubernetes` `Endpoints` del espacio de nombres `default`. Cuando EKS crea nuevas ENI o elimina las antiguas, EKS actualiza este objeto para que la lista de IP esté siempre actualizada.

Puede utilizar estos puntos de conexión a través del servicio de `kubernetes`, también en el espacio de nombres `default`. A este servicio, del tipo `ClusterIP`, siempre se le asigna la primera IP del CIDR del servicio del clúster. Por ejemplo, para el servicio de CIDR `172.16.0.0/16`, la IP del servicio será `172.16.0.1`.

Por lo general, así es como los pods (independientemente de si se ejecutan en la nube o en nodos híbridos) acceden al servidor de la API de Kubernetes de EKS. Los pods utilizan la IP del servicio como IP de destino, que se traduce en las IP reales de una de las ENI del plano de control de EKS. La excepción principal es `kube-proxy`, porque configura la traducción.

Punto de conexión del servidor de la API de EKS

La IP del servicio de `kubernetes` no es la única forma de acceder al servidor de la API de EKS. EKS también crea un nombre de DNS Route53 cuando crea el clúster. Este es el campo `endpoint` de su clúster de EKS al llamar a la acción de la API `DescribeCluster` de EKS.

```
{
    "cluster": {
        "endpoint": "https://xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx.gr7.us-
west-2.eks.amazonaws.com",
```

```
        "name": "my-cluster",
        "status": "ACTIVE"
    }
}
```

En un clúster de acceso a puntos de conexión públicos o públicos y privados, sus nodos híbridos convertirán este nombre de DNS en una IP pública de forma predeterminada, enrutable a través de Internet. En un clúster de acceso a puntos de conexión privados, el nombre de DNS se convierte en las IP privadas de las ENI del plano de control de EKS.

Así es como `kubelet` y `kube-proxy` acceden al servidor de la API de Kubernetes. Si desea que todo el tráfico del clúster de Kubernetes fluya a través de la VPC, debe configurar el clúster en modo de acceso privado o modificar el servidor de DNS en las instalaciones para resolver el punto de conexión del clúster de EKS en las IP privadas de las ENI del plano de control de EKS.

Punto de conexión de **`kubelet`**

El `kubelet` expone varios puntos de conexión REST, lo que permite que otras partes del sistema interactúen con cada nodo y recopilen información de él. En la mayoría de los clústeres, la mayor parte del tráfico al servidor `kubelet` proviene del plano de control, pero algunos agentes de supervisión también pueden interactuar con él.

A través de esta interfaz, el `kubelet` gestiona varias solicitudes: buscar registros (`kubectl logs`), ejecutar comandos dentro de contenedores (`kubectl exec`) y reenviar el tráfico de puertos (`kubectl port-forward`). Cada una de estas solicitudes interactúa con el tiempo de ejecución del contenedor subyacente durante todo el `kubelet`, lo que resulta perfecto para los administradores y desarrolladores del clúster.

El consumidor más habitual de esta API es el servidor de la API de Kubernetes. Cuando utiliza alguno de los comandos de `kubectl` mencionados anteriormente, `kubectl` realiza una solicitud de API al servidor de la API, que luego llama a la API de `kubelet` del nodo en el que se está ejecutando el pod. Esta es la razón principal por la que es necesario poder acceder a la IP del nodo desde el plano de control de EKS y por la que, aunque sus pods estén en ejecución, no podrá acceder a sus registros o `exec` si la ruta del nodo está mal configurada.

IP de los nodos

Cuando el plano de control EKS se comunica con un nodo, utiliza una de las direcciones indicadas en el estado del objeto `Node` (`status.addresses`).

En el caso de los nodos en la nube de EKS, es habitual que el kubelet registre la IP privada de la instancia EC2 como `InternalIP` durante el registro del nodo. A continuación, el administrador de controladores en la nube (CCM) valida esta IP para asegurarse de que pertenece a la instancia EC2. Además, el CCM suele añadir las IP públicas (como `ExternalIP`) y los nombres de DNS (`InternalDNS` y `ExternalDNS`) de la instancia al estado del nodo.

Sin embargo, no hay ningún CCM para los nodos híbridos. Al registrar un nodo híbrido con la CLI (`nodeadm`) de los nodos híbridos de EKS, se configura el kubelet para que indique la IP de la máquina directamente en el estado del nodo, sin el 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
```

Si su máquina tiene varias IP, el kubelet seleccionará una de ellas siguiendo su propia lógica. Puede controlar la IP seleccionada con la advertencia `--node-ip`, que puede introducir en la configuración de `nodeadm` en `spec.kubelet.flags`. Solo la IP indicada en el objeto `Node` necesita una ruta desde la VPC. Sus máquinas pueden tener otras IP a las que no se pueda acceder desde la nube.

kube-proxy

`kube-proxy` es responsable de implementar la abstracción del servicio en la capa de red de cada nodo. Actúa como proxy de red y equilibrador de carga para el tráfico destinado a los servicios de Kubernetes. Al vigilar continuamente el servidor de la API de Kubernetes para detectar cambios relacionados con los servicios y los puntos de conexión, `kube-proxy` actualiza dinámicamente las reglas de red del host subyacente para garantizar que el tráfico se dirija correctamente.

En el modo `iptables`, `kube-proxy` programa varias cadenas `netfilter` para administrar el tráfico del servicio. Las reglas forman la siguiente jerarquía:

1. Cadena KUBE-SERVICES: el punto de entrada de todo el tráfico del servicio. Tiene reglas que coinciden con cada `ClusterIP` y puerto del servicio.

2. Cadenas KUBE-SVC-XXX: las cadenas específicas del servicio tienen reglas de equilibrio de carga para cada servicio.
3. Cadenas KUBE-SEP-XXX: las cadenas específicas del punto de conexión tienen las reglas DNAT reales.

Analicemos qué ocurre con un `test-server` de servicio en el espacio de nombres `default`:
* ClusterIP del servicio: `172.16.31.14` * Puerto del servicio: `80` * Pods de respaldo: `10.2.0.110`, `10.2.1.39` y `10.2.2.254`

Cuando inspeccionamos las reglas `iptables` (con `iptables-save 0 grep -A10 KUBE-SERVICES`):

1. En la cadena KUBE-SERVICES, encontramos una regla que coincide con el servicio:

```
-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
```

   - Esta regla coincide con los paquetes destinados a 172.16.31.14:80
   - El comentario indica para qué sirve esta regla: `default/test-server cluster IP`
   - Los paquetes coincidentes pasan a la cadena `KUBE-SVC-XYZABC123456`
2. La cadena KUBE-SVC-XYZABC123456 tiene reglas de equilibrio de carga basadas en probabilidades:

```
-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
```

   - Primera regla: 33,3 % de probabilidad de saltar a `KUBE-SEP-POD1XYZABC`
   - Segunda regla: 50 % de probabilidades de que el tráfico restante (el 33,3 % del total) salte a `KUBE-SEP-POD2XYZABC`
   - Última regla: todo el tráfico restante (el 33,3 % del total) salta a `KUBE-SEP-POD3XYZABC`
3. Las cadenas KUBE-SEP-XXX individuales realizan la DNAT (NAT de destino):

```
-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
```

- Estas reglas de DNAT reescriben la IP y el puerto de destino para dirigir el tráfico a pods específicos.
- Cada regla gestiona aproximadamente el 33,3 % del tráfico, lo que proporciona un equilibrio de carga uniforme entre `10.2.0.110`, `10.2.1.39` y `10.2.2.254`.

Esta estructura de cadena de varios niveles permite que `kube-proxy` implemente de manera eficiente el equilibrio y la redirección de la carga de servicios mediante la manipulación de paquetes a nivel del núcleo, sin necesidad de un proceso proxy en la ruta de datos.

Impacto en las operaciones de Kubernetes

Si hay un `kube-proxy` roto en un nodo, impide que ese nodo enrute correctamente el tráfico del servicio, lo que provoca tiempos de espera o fallos en las conexiones de los pods que dependen de los servicios del clúster. Esto puede ser especialmente perjudicial cuando se registra un nodo por primera vez. La CNI necesita hablar con el servidor de la API de Kubernetes para obtener información, como el CIDR del pod del nodo, antes de poder configurar cualquier red de pods. Para ello, utiliza la IP del servicio de `kubernetes`. Sin embargo, si `kube-proxy` no se ha podido iniciar o no se han establecido las reglas `iptables` correctas, las solicitudes que se envían a la IP del servicio de `kubernetes` no se traducen a las IP reales de las ENI del plano de control de EKS. Como consecuencia, la CNI entrará en un círculo vicioso y ninguno de los pods podrá funcionar correctamente.

Sabemos que los pods utilizan la IP del servicio de `kubernetes` para comunicarse con el servidor de la API de Kubernetes, pero primero `kube-proxy` debe establecer reglas `iptables` para que funcione.

¿Cómo se comunica `kube-proxy` con el servidor de la API?

`kube-proxy` debe configurarse para usar las IP reales del servidor de la API de Kubernetes o un nombre de DNS que las resuelva. En el caso de EKS, este servicio configura el valor de `kube-proxy` predeterminado para que apunte al nombre DNS de Route53 que EKS genera al crear el clúster. Puede ver este valor en el ConfigMap del `kube-proxy`, en el espacio de nombres `kube-system`. El contenido de este ConfigMap es un `kubeconfig` que se inyecta en el pod de `kube-proxy`, así que debe buscar el campo `clusters0.cluster.server`. Este valor coincidirá con el campo `endpoint` de su clúster de EKS (al llamar a la API `DescribeCluster` de EKS).

```
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
```

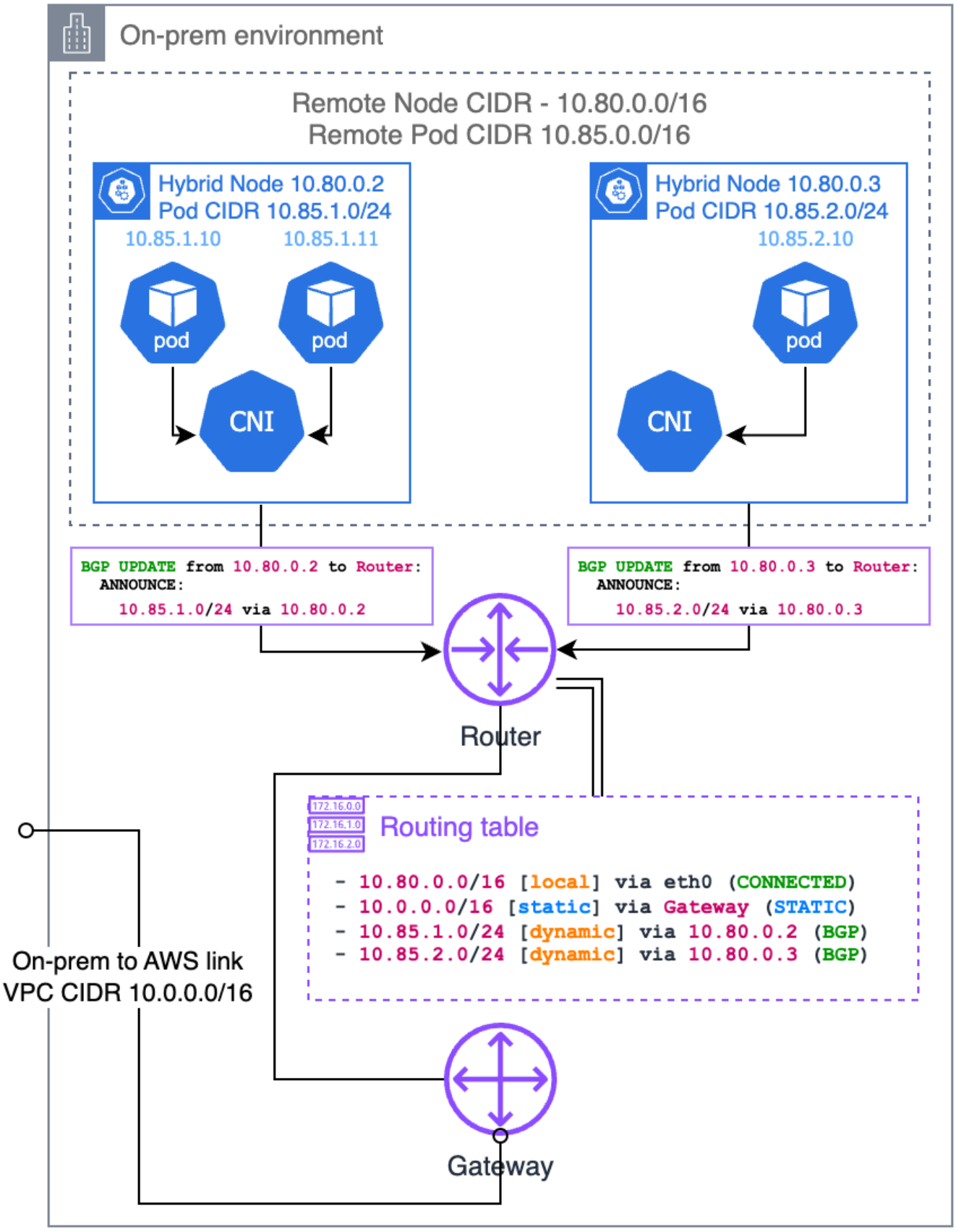
CIDR de pod remoto enrutable

En la página the section called “Conceptos relacionados con las redes”, se detallan los requisitos para ejecutar webhooks en nodos híbridos o para que los pods que se ejecutan en nodos de la nube se comuniquen con los pods que se ejecutan en nodos híbridos. El requisito clave es que el enrutador en las instalaciones debe saber qué nodo es responsable de la IP de un pod concreto. Existen varias formas de lograrlo, como el protocolo de puerta de enlace fronteriza (BGP), las rutas estáticas y el uso de proxies con el protocolo de resolución de direcciones (ARP). Estos pasos se detallan en las siguientes secciones.

Protocolo de puerta de enlace fronteriza (BGP)

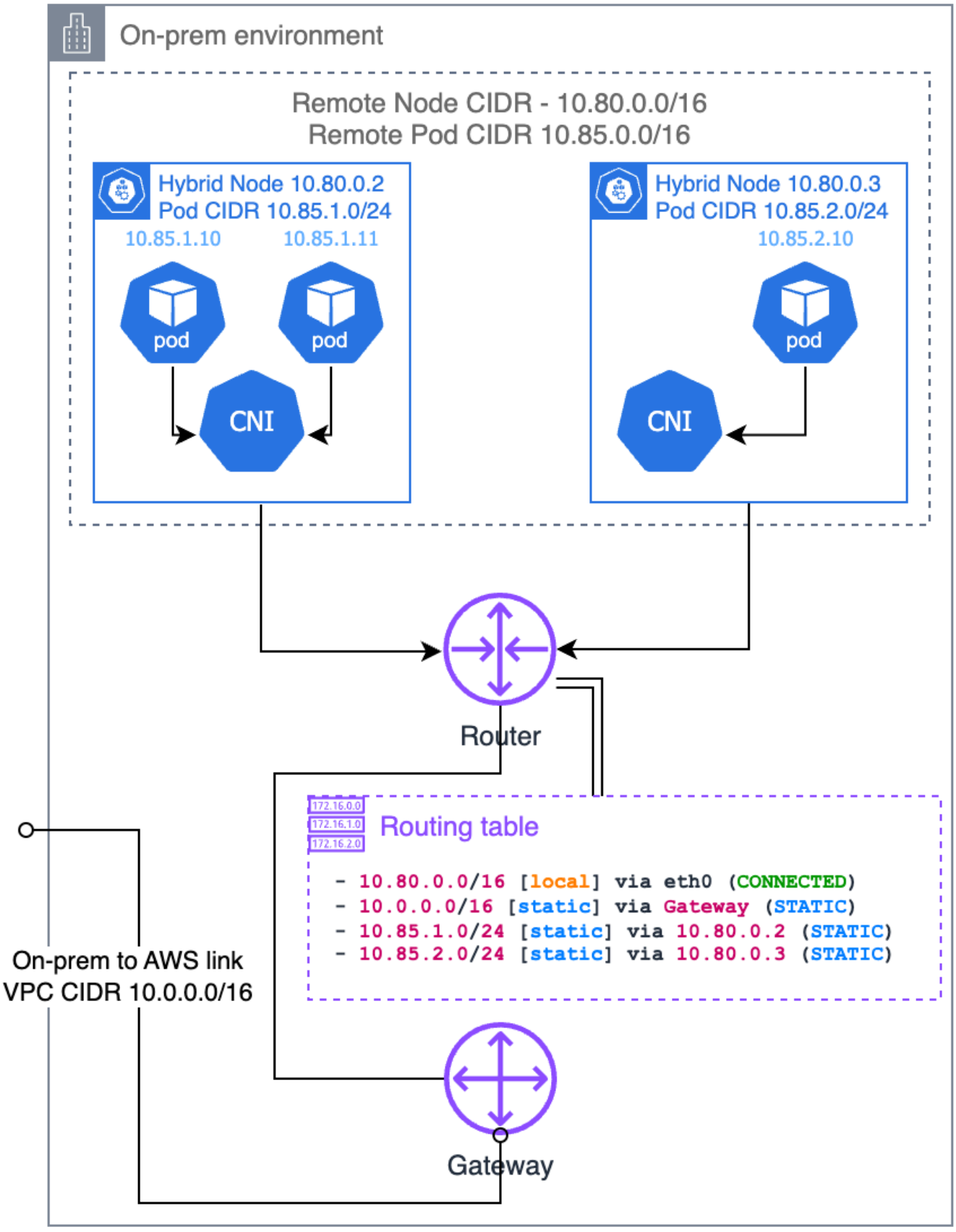
Si su CNI lo admite (por ejemplo, Cilium y Calico), puede utilizar el modo de BGP de su CNI para propagar las rutas a los CIDR de cada pod del nodo desde los nodos hasta el enrutador local. Al utilizar el modo de BGP de la CNI, esta actúa como un enrutador virtual, por lo que el enrutador local

cree que el CIDR del pod pertenece a una subred diferente y que el nodo es la puerta de enlace a esa subred.


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

Rutas estáticas

De otro modo, puede configurar rutas estáticas en el enrutador local. Esta es la forma más sencilla de enrutar el CIDR del pod en las instalaciones a su VPC, pero también es la más propensa a errores y la más difícil de mantener. Debe asegurarse de que las rutas estén siempre actualizadas con los nodos existentes y sus CIDR de pod asignados. Si su número de nodos es pequeño y la infraestructura es estática, esta es una opción viable y elimina la necesidad de admitir BGP en su enrutador. Si elige esta opción, le recomendamos que configure su CNI con el segmento CIDR del pod que desee asignar a cada nodo, en lugar de dejar que su IPAM decida.


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
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 [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

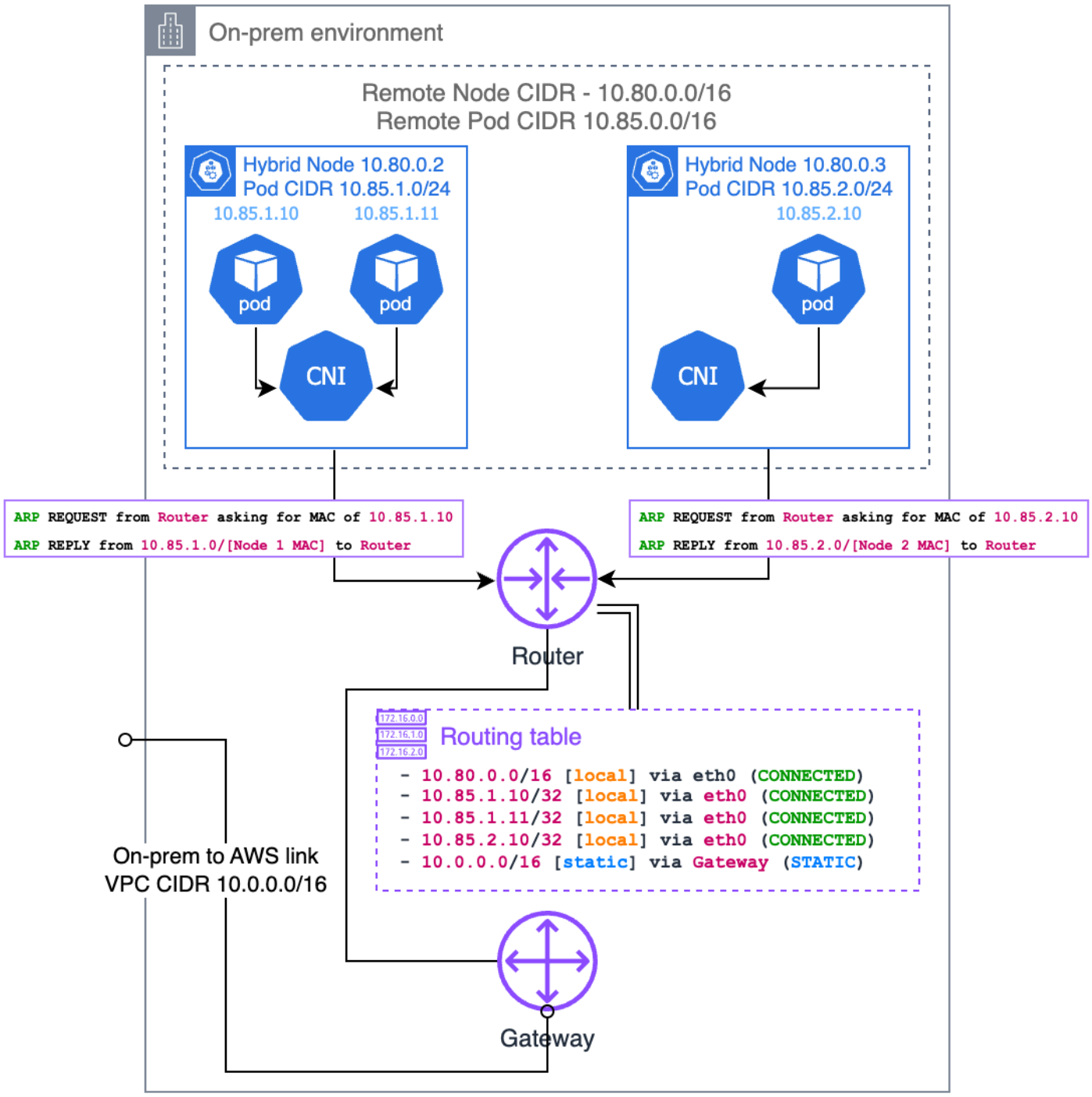
Uso de proxy del protocolo de resolución de direcciones (ARP)

El uso de proxy de ARP es otro enfoque para hacer que las IP de los pods en las instalaciones sean enrutables, lo que resulta especialmente útil cuando los nodos híbridos se encuentran en la misma red de capa 2 que el enrutador local. Con el proxy de ARP habilitado, un nodo responde a las solicitudes de ARP de las IP de los pods que aloja, aunque esas IP pertenezcan a una subred diferente.

Cuando un dispositivo de la red local intenta acceder a la IP de un pod, primero envía una solicitud de ARP en la que pregunta “¿Quién tiene esta IP?”. El nodo híbrido que aloja ese pod responderá con su propia dirección MAC y dirá: “Puedo gestionar el tráfico de esa IP”. Esto crea una ruta directa entre los dispositivos de la red local y los pods sin necesidad de configurar el enrutador.

Para que esto funcione, su CNI debe ser compatible con la funcionalidad de ARP del proxy. Cilium cuenta con compatibilidad integrada para el ARP del proxy que se puede activar mediante la configuración. La consideración clave es que el CIDR del pod no debe superponerse con ninguna otra red de su entorno, ya que esto podría provocar conflictos de enrutamiento.

Este enfoque tiene varias ventajas: * No es necesario configurar el enrutador con un BGP ni mantener rutas estáticas * Funciona bien en entornos en los que no se tiene control de la configuración del enrutador.



Encapsulación de pod a pod

En los entornos en las instalaciones, las CNI suelen utilizar protocolos de encapsulación para crear redes superpuestas que puedan funcionar sobre la red física sin necesidad de volver a configurarla. En esta sección, se explica cómo funciona esta encapsulación. Tenga en cuenta que algunos de los detalles pueden variar en función de la CNI que utilice.

La encapsulación envuelve los paquetes de red del pod originales dentro de otro paquete de red que se puede enrutar a través de la red física subyacente. Esto permite que los pods se comuniquen entre nodos que ejecutan la misma CNI sin necesidad de que la red física sepa cómo enrutar esos CIDR de los pods.

El protocolo de encapsulación más común que se utiliza con Kubernetes es la LAN virtual extensible (VXLAN), aunque también hay otros (por ejemplo, `Geneve`) disponibles en función de la CNI.

Encapsulación mediante VXLAN

VXLAN encapsula las tramas Ethernet de capa 2 dentro de los paquetes UDP. Cuando un pod envía tráfico a otro pod en un nodo diferente, la CNI realiza lo siguiente:

1. La CNI intercepta los paquetes del pod A.
2. La CNI envuelve el paquete original en un encabezado de VXLAN.
3. Este paquete envuelto se envía luego a través de la pila de red normal del nodo al nodo de destino.
4. La CNI del nodo de destino desenvuelve el paquete y lo entrega al pod B.

Esto es lo que ocurre con la estructura del paquete durante la encapsulación mediante VXLAN:

Paquete original de pod a pod:

```
+-----------------+---------------+-------------+-----------------+
| 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    |                 |
+-----------------+---------------+-------------+-----------------+
```

Después de la encapsulación mediante VXLAN:

```
+-----------------+-------------+--------------+------------
+--------------------------+
| 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)
 |
```

```
+-----------------+-------------+--------------+------------
+--------------------------+
```

El identificador de red de VXLAN (VNI) distingue entre diferentes redes superpuestas.

Escenarios de comunicación por pod

Pods en el mismo nodo híbrido

Cuando los pods del mismo nodo híbrido se comunican, normalmente no es necesaria la encapsulación. La CNI configura rutas locales que dirigen el tráfico entre los pods a través de las interfaces virtuales internas del nodo:

```
Pod A -> veth0 -> node's bridge/routing table -> veth1 -> Pod B
```

El paquete nunca sale del nodo y no requiere encapsulación.

Pods en diferentes nodos híbridos

La comunicación entre los pods de diferentes nodos híbridos requiere encapsulación:

```
Pod A -> CNI -> [VXLAN encapsulation] -> Node A network -> router or gateway -> Node B
 network -> [VXLAN decapsulation] -> CNI -> Pod B
```

Esto permite que el tráfico del pod atraviese la infraestructura de red física sin necesidad de que la red física comprenda el enrutamiento de la IP del pod.

Flujos de tráfico de red para nodos híbridos

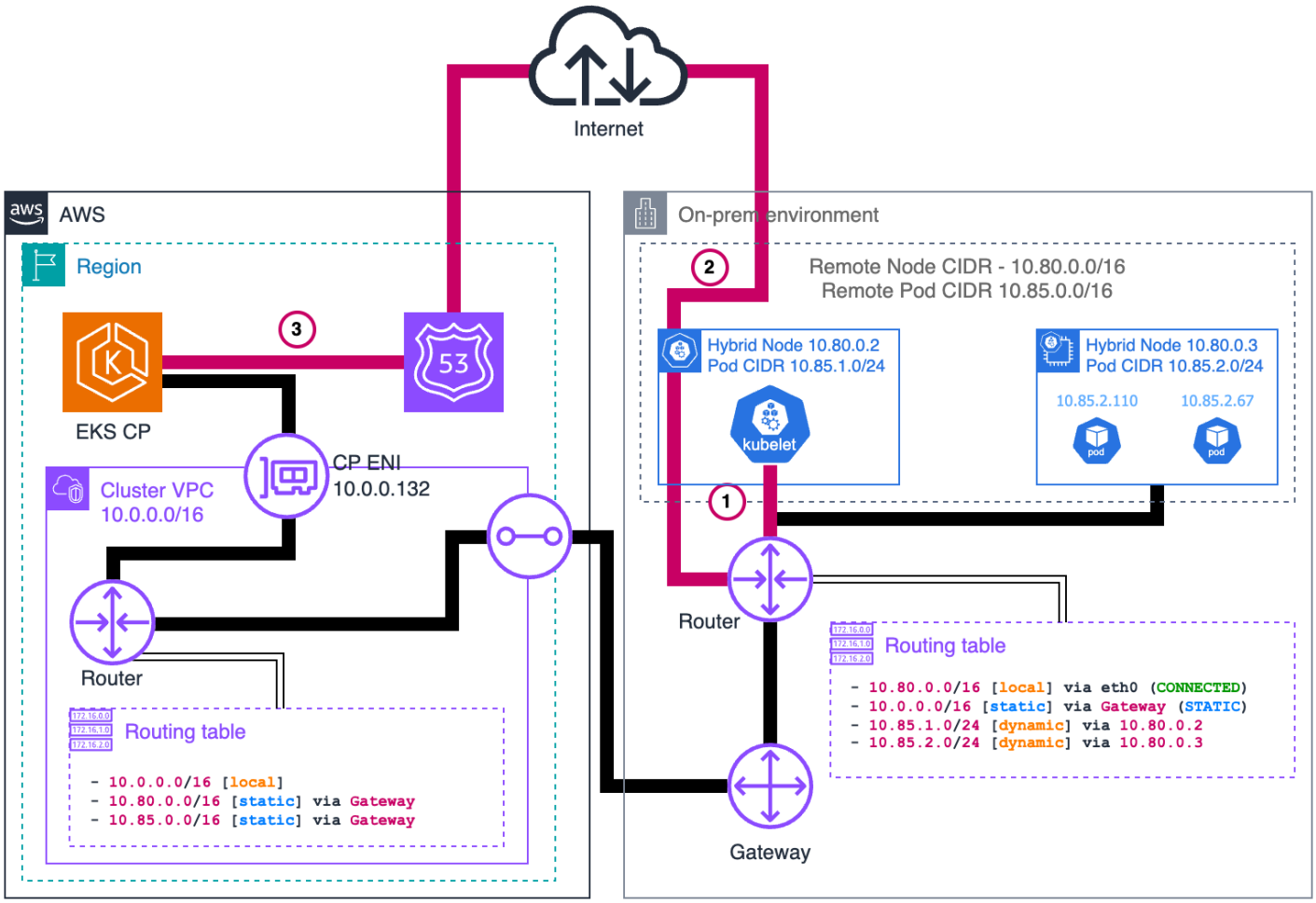
En esta página, se detallan los flujos de tráfico de red para los nodos híbridos de EKS con diagramas que muestran las rutas de red de extremo a extremo para los diferentes tipos de tráfico.

Se tratan los siguientes flujos de tráfico:

- the section called “Del `kubelet` del nodo híbrido al plano de control de EKS”
- the section called “Del plano de control de EKS al nodo híbrido (servidor de `kubelet`)”
- the section called “Desde pods que se ejecutan en nodos híbridos hasta el plano de control de EKS”
- the section called “Del plano de control de EKS a los pods que se ejecutan en un nodo híbrido (webhooks)”

- the section called “De pod a pod que se ejecuta en nodos híbridos”
- the section called “De pods de nodos en la nube a pods de nodos híbridos (tráfico de este a oeste)”

Del **`kubelet`** del nodo híbrido al plano de control de EKS



Solicitud

1**`kubelet`**. Inicia la solicitud

Cuando el `kubelet` de un nodo híbrido necesita comunicarse con el plano de control de EKS (por ejemplo, para informar del estado del nodo u obtener las especificaciones del pod), utiliza el archivo `kubeconfig` proporcionado durante el registro del nodo. Este `kubeconfig` tiene la URL del punto de conexión del servidor de la API (el nombre DNS de Route53) en lugar de direcciones IP directas.

El `kubelet` realiza una consulta DNS para el punto de conexión (por ejemplo, `https://xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx.gr7.us-west-2.eks.amazonaws.com`).

En un clúster con acceso público, esto se resuelve en una dirección IP pública (por ejemplo, `54.239.118.52`) que pertenece al servicio de EKS que se ejecuta en AWS. Luego, el `kubelet` crea una solicitud HTTPS segura para este punto de conexión. El paquete inicial tiene un aspecto similar al siguiente:

```
+-------------------+--------------------+-----------------+
| IP Header         | TCP Header         | Payload         |
| Src: 10.80.0.2    | Src: 52390 (random) |                 |
| Dst: 54.239.118.52 | Dst: 443           |                 |
+-------------------+--------------------+-----------------+
```

2. Enrutamiento del enrutador local

Como la IP de destino es una dirección IP pública y no forma parte de la red local, el `kubelet` envía este paquete a su puerta de enlace predeterminada (el enrutador en las instalaciones). El enrutador examina la IP de destino y determina que es una dirección IP pública.

En el caso del tráfico público, el enrutador normalmente reenvía el paquete a una puerta de enlace de Internet o a un enrutador fronterizo que gestiona el tráfico saliente a Internet. Esto se omite en el diagrama y dependerá de la configuración de la red en las instalaciones. El paquete atraviesa la infraestructura de red en las instalaciones y, finalmente, llega a la red de su proveedor de servicios de Internet.

3. Entrega al plano de control de EKS

El paquete atraviesa Internet y redes de tránsito públicas hasta llegar a la red de AWS, donde se dirige al punto de conexión del servicio de EKS en la región correspondiente. Cuando el paquete llega al servicio de EKS, se reenvía al plano de control de EKS real del clúster.

Este enrutamiento a través de la Internet pública es diferente de la ruta privada enrutada por la VPC que veremos en otros flujos de tráfico. La diferencia clave es que, cuando se utiliza el modo de acceso público, el tráfico desde el `kubelet` en las instalaciones (aunque no desde los pods) hasta el plano de control de EKS no pasa por la VPC, sino que utiliza la infraestructura global de Internet.

Respuesta

Una vez que el plano de control de EKS procesa la solicitud de `kubelet`, envía una respuesta:

3. El plano de control de EKS envía una respuesta

El plano de control de EKS genera un paquete de respuesta. Este paquete tiene como origen la IP pública y como destino la IP del nodo híbrido:

```
+--------------------+---------------------+-----------------+
| IP Header          | TCP Header          | Payload         |
| Src: 54.239.118.52 | Src: 443            |                 |
| Dst: 10.80.0.2     | Dst: 52390          |                 |
+--------------------+---------------------+-----------------+
```

2. Enrutamiento de Internet

El paquete de respuesta regresa a través de Internet, siguiendo la ruta de enrutamiento determinada por los proveedores de servicios de Internet, hasta llegar al enrutador periférico de la red en las instalaciones.

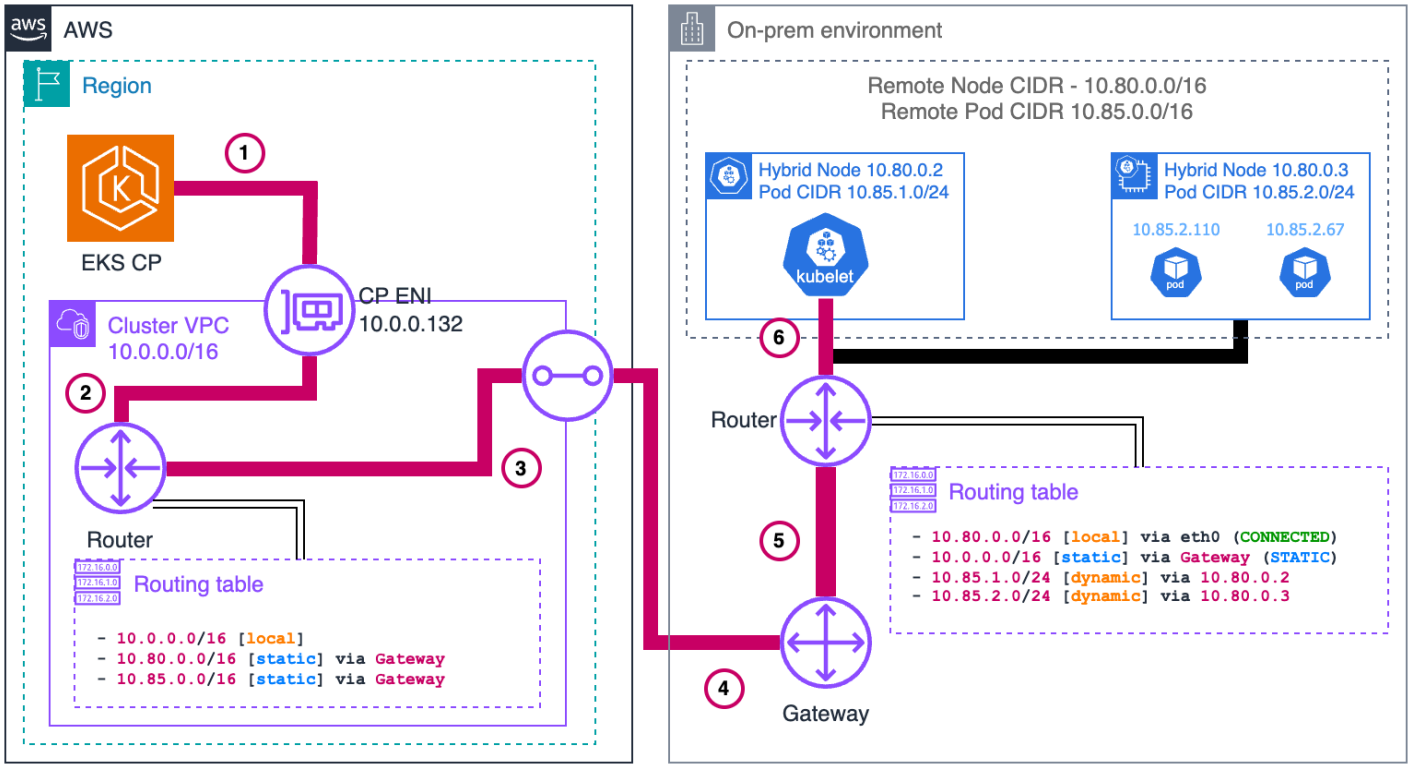
1. Entrega local

El enrutador en las instalaciones recibe el paquete y reconoce que la IP de destino (`10.80.0.2`) pertenece a la red local. Reenvía el paquete a través de la infraestructura de red local hasta que llega al nodo híbrido de destino, donde el `kubelet` recibe y procesa la respuesta.

Del **`kube-proxy`** del nodo híbrido al plano de control de EKS

Si habilita el acceso al punto de conexión público para el clúster, el tráfico de retorno utilizará Internet pública. Este tráfico se origina en el `kube-proxy` del nodo híbrido hacia el plano de control de EKS y sigue la misma ruta que el tráfico del `kubelet` hacia el plano de control de EKS.

Del plano de control de EKS al nodo híbrido (servidor de **`kubelet`**)



Solicitud

1. El servidor de la API de Kubernetes de EKS inicia la solicitud

El servidor de la API de Kubernetes en EKS obtiene la dirección IP del nodo (`10.80.0.2`) a partir del estado del objeto del nodo. Luego dirige esta solicitud a través de la ENI en la VPC, ya que la IP de destino pertenece al CIDR de nodo remoto configurado (`10.80.0.0/16`). El paquete inicial tiene un aspecto similar al siguiente:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.0.0.132 | Src: 67493 (random) |                 |
| Dst: 10.80.0.2  | Dst: 10250          |                 |
+-----------------+---------------------+-----------------+
```

2. Procesamiento de redes de la VPC

El paquete sale de la ENI y entra en la capa de red de la VPC, donde se dirige a la puerta de enlace de la subred para su posterior enrutamiento.

3. Búsqueda en la tabla de enrutamiento de la VPC

La tabla de enrutamiento de la VPC para la subred que contiene la ENI del plano de control de EKS tiene una ruta específica (la segunda del diagrama) para el CIDR del nodo remoto. Según esta regla de enrutamiento, el paquete se dirige a la puerta de enlace de VPC a la red en las instalaciones.

4. Tránsito transfronterizo

La puerta de enlace transfiere el paquete a través de los límites de la nube mediante la conexión establecida (como Direct Connect o VPN) a la red en las instalaciones.

5. Recepción de la red en las instalaciones

El paquete llega al enrutador en las instalaciones que gestiona el tráfico de la subred en la que se encuentran los nodos híbridos.

6. Entrega final

El enrutador local identifica que la dirección IP de destino (`10.80.0.2`) pertenece a su red directamente conectada y reenvía el paquete directamente al nodo híbrido de destino, donde el `kubelet` lo recibe y procesa la solicitud.

Respuesta

Una vez que el `kubelet` del nodo híbrido procesa la solicitud, envía una respuesta siguiendo la misma ruta a la inversa:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.80.0.2  | Src: 10250          |                 |
| Dst: 10.0.0.132 | Dst: 67493          |                 |
+-----------------+---------------------+-----------------+
```

6. **`kubelet`** Envía una respuesta

El `kubelet` en el nodo híbrido (`10.80.0.2`) crea un paquete de respuesta con la IP de origen original como destino. El destino no pertenece a la red local, por lo que se envía a la puerta de enlace predeterminada del host, que es el enrutador local.

5. Enrutamiento del enrutador local

El enrutador determina que la IP de destino (`10.0.0.132`) pertenece a `10.0.0.0/16`, que tiene una ruta que apunta a la puerta de enlace que conecta con AWS.

4. Retorno transfronterizo

El paquete regresa a través de la misma conexión en las instalaciones a la VPC (como Direct Connect o VPN) y cruza el límite de la nube en la dirección opuesta.

3. Enrutamiento de la VPC

Cuando el paquete llega a la VPC, las tablas de enrutamiento identifican que la IP de destino pertenece a un CIDR de la VPC. El paquete se enruta dentro de la VPC.

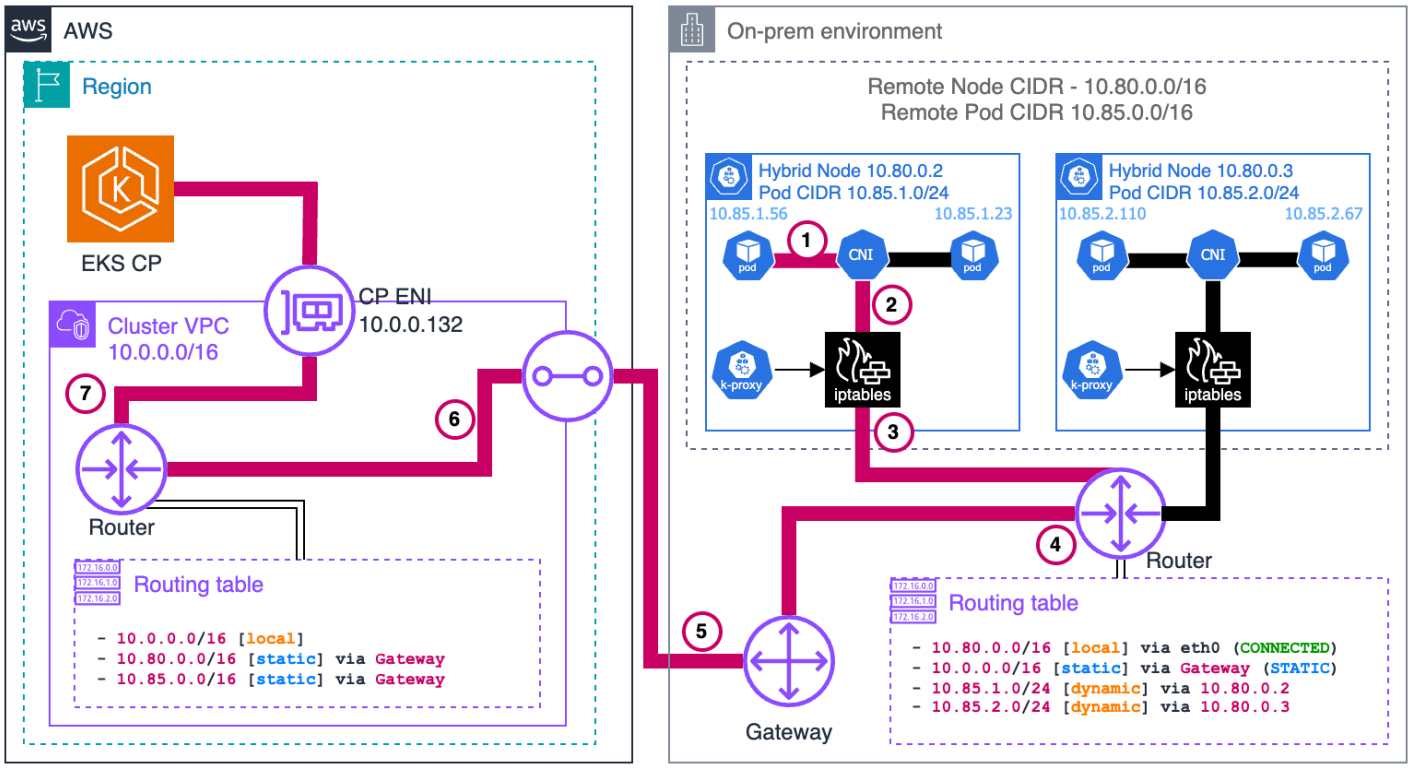
2. Entrega en la red de la VPC

La capa de red de la VPC reenvía el paquete a la subred con la ENI del plano de control de EKS (`10.0.0.132`).

1. Recepción de la ENI

El paquete llega a la ENI del plano de control de EKS conectado al servidor de la API de Kubernetes y completa el viaje de ida y vuelta.

Desde pods que se ejecutan en nodos híbridos hasta el plano de control de EKS



Sin NAT de la CNI

Solicitud

Por lo general, los pods se comunican con el servidor de la API de Kubernetes a través del servicio de `kubernetes`. La IP del servicio es la primera IP del CIDR del servicio del clúster. Esta convención permite que los pods que deben ejecutarse antes de que CoreDNS esté disponible lleguen al servidor de la API, por ejemplo, la CNI. Las solicitudes salen del pod con la IP del servicio como destino. Por ejemplo, si el CIDR del servicio es `172.16.0.0/16`, la IP del servicio será `172.16.0.1`.

1. El pod inicia la solicitud

El pod envía una solicitud a la IP del servicio de `kubernetes` (`172.16.0.1`) en el puerto del servidor de la API (443) desde un puerto de origen aleatorio. El paquete tiene un aspecto similar al siguiente:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.85.1.56 | Src: 67493 (random) |                 |
| Dst: 172.16.0.1 | Dst: 443            |                 |
+-----------------+---------------------+-----------------+
```

2. Procesamiento de la CNI

La CNI detecta que la IP de destino no pertenece a ningún CIDR de pod que administre. Como la NAT saliente está deshabilitada, la CNI pasa el paquete a la pila de la red del host sin modificarlo.

3. Procesamiento de la red del nodo

El paquete entra en la pila de red del nodo, donde los enlaces `netfilter` activan las reglas `iptables` establecidas por kube-proxy. Se aplican varias reglas en el siguiente orden:

1. El paquete llega primero a la cadena `KUBE-SERVICES`, que contiene reglas que coinciden con el ClusterIP y el puerto de cada servicio.
2. La regla coincidente salta a la cadena `KUBE-SVC-XXX` del servicio de `kubernetes` (paquetes destinados a `172.16.0.1:443`), que contiene reglas de equilibrio de carga.
3. La regla de equilibrio de carga selecciona aleatoriamente una de las cadenas `KUBE-SEP-XXX` para las IP (`10.0.0.132` o `10.0.1.23`) de la ENI del plano de control.

4. La cadena KUBE-SEP-XXX seleccionada tiene la regla en sí que cambia la IP de destino de la IP de servicio a la IP seleccionada. Esto se conoce como traducción de direcciones de red de destino (DNAT).

Una vez aplicadas estas reglas, suponiendo que la IP de la ENI del plano de control de EKS seleccionada sea 10.0.0.132, el paquete tendrá el siguiente aspecto:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.85.1.56 | Src: 67493 (random) |                 |
| Dst: 10.0.0.132 | Dst: 443            |                 |
+-----------------+---------------------+-----------------+
```

El nodo reenvía el paquete a su puerta de enlace predeterminada porque la IP de destino no está en la red local.

4. Enrutamiento del enrutador local

El enrutador local determina que la IP de destino (10.0.0.132) pertenece al CIDR de la VPC (10.0.0.0/16) y la reenvía a la puerta de enlace que conecta con AWS.

5. Tránsito transfronterizo

El paquete viaja mediante la conexión establecida (como Direct Connect o la VPN) a través del límite de la nube hasta la VPC.

6. Entrega en la red de la VPC

La capa de red de la VPC dirige el paquete a la subred correcta donde se encuentra la ENI (10.0.0.132) del plano de control de EKS.

7. Recepción de la ENI

El paquete llega a la ENI del plano de control de EKS conectado al servidor de la API de Kubernetes.

Respuesta

Una vez que el plano de control de EKS procesa la solicitud, envía una respuesta al pod:

7. El servidor de la API envía una respuesta

El servidor de la API de Kubernetes de EKS crea un paquete de respuesta con la IP de origen original como destino. El paquete tiene un aspecto similar al siguiente:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.0.0.132 | Src: 443            |                 |
| Dst: 10.85.1.56 | Dst: 67493          |                 |
+-----------------+---------------------+-----------------+
```

Como la IP de destino pertenece al CIDR del pod remoto configurado (`10.85.0.0/16`), la envía a través de su ENI en la VPC con el enrutador de la subred como siguiente salto.

6. Enrutamiento de la VPC

La tabla de enrutamiento de la VPC contiene una entrada para el CIDR del pod remoto (`10.85.0.0/16`), que dirige este tráfico a la puerta de enlace de la VPC a la red en las instalaciones.

5. Tránsito transfronterizo

La puerta de enlace transfiere el paquete a través de los límites de la nube mediante la conexión establecida (como Direct Connect o VPN) a la red en las instalaciones.

4. Recepción de la red en las instalaciones

El paquete llega a su enrutador en las instalaciones.

3. Entrega al nodo

La tabla del enrutador tiene una entrada para `10.85.1.0/24` con `10.80.0.2` como siguiente salto, que entrega el paquete a nuestro nodo.

2. Procesamiento de la red del nodo

A medida que el paquete es procesado por la pila de red del nodo, `conntrack` (que forma parte de `netfilter`) encuentra una coincidencia con la conexión que el pod estableció inicialmente. Como se aplicó DNAT originalmente, `kubernetes` revierte el DNAT. Para ello, reescribe la IP de origen y reemplaza la IP de la ENI del plano de control de EKS por la IP del servicio de `conntrack`.

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 172.16.0.1 | Src: 443            |                 |
```

```
| Dst: 10.85.1.56 | Dst: 67493          |                 |
+-----------------+---------------------+-----------------+
```

1. Procesamiento de la CNI

La CNI identifica que la IP de destino pertenece a un pod de su red y entrega el paquete al espacio de nombres de la red del pod correcto.

Este flujo muestra por qué los CIDR de pods remotos deben poder enrutarse correctamente desde la VPC hasta el nodo específico que aloja cada pod; toda la ruta de retorno depende del enrutamiento correcto de las IP de los pods en las redes en las instalaciones y en la nube.

Con NAT de la CNI

Este flujo es muy similar al que no tiene la NAT de la CNI, pero con una diferencia clave: la CNI aplica la NAT de origen (SNAT) al paquete antes de enviarlo a la pila de red del nodo. Esto cambia la IP de origen del paquete por la IP del nodo, lo que permite que el paquete se enrute de vuelta al nodo sin requerir una configuración de enrutamiento adicional.

Solicitud

1. El pod inicia la solicitud

El pod envía una solicitud a la IP del servicio de `kubernetes` (`172.16.0.1`) del puerto del servidor de la API de Kubernetes de EKS (443) desde un puerto de origen aleatorio. El paquete tiene un aspecto similar al siguiente:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.85.1.56 | Src: 67493 (random) |                 |
| Dst: 172.16.0.1 | Dst: 443            |                 |
+-----------------+---------------------+-----------------+
```

2. Procesamiento de la CNI

La CNI detecta que la IP de destino no pertenece a ningún CIDR de pod que administre. Como la NAT saliente está habilitada, la CNI aplica la SNAT al paquete y cambia la IP de origen por la IP del nodo antes de pasarla a la pila de redes del nodo:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.80.0.2  | Src: 67493 (random) |                 |
```

```
| Dst: 172.16.0.1 | Dst: 443            |                 |
+-----------------+---------------------+-----------------+
```

Nota: La CNI y `iptables` se muestran en el ejemplo como bloques separados para mayor claridad, pero en la práctica, es posible que algunas CNI utilicen `iptables` para aplicar la NAT.

3. Procesamiento de la red del nodo

En este caso, las reglas `iptables` establecidas por `kube-proxy` se comportan igual que en el ejemplo anterior, equilibrando la carga del paquete con una de las ENI del plano de control de EKS. El paquete ahora tiene un aspecto similar al siguiente:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.80.0.2  | Src: 67493 (random) |                 |
| Dst: 10.0.0.132 | Dst: 443            |                 |
+-----------------+---------------------+-----------------+
```

El nodo reenvía el paquete a su puerta de enlace predeterminada porque la IP de destino no está en la red local.

4. Enrutamiento del enrutador local

El enrutador local determina que la IP de destino (`10.0.0.132`) pertenece al CIDR de la VPC (`10.0.0.0/16`) y la reenvía a la puerta de enlace que conecta con AWS.

5. Tránsito transfronterizo

El paquete viaja mediante la conexión establecida (como Direct Connect o la VPN) a través del límite de la nube hasta la VPC.

6. Entrega en la red de la VPC

La capa de red de la VPC dirige el paquete a la subred correcta donde se encuentra la ENI (`10.0.0.132`) del plano de control de EKS.

7. Recepción de la ENI

El paquete llega a la ENI del plano de control de EKS conectado al servidor de la API de Kubernetes.

Respuesta

Una vez que el plano de control de EKS procesa la solicitud, envía una respuesta al pod:

7. El servidor de la API envía una respuesta

El servidor de la API de Kubernetes de EKS crea un paquete de respuesta con la IP de origen original como destino. El paquete tiene un aspecto similar al siguiente:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.0.0.132 | Src: 443            |                 |
| Dst: 10.80.0.2  | Dst: 67493          |                 |
+-----------------+---------------------+-----------------+
```

Como la IP de destino pertenece al CIDR de nodo remoto configurado (`10.80.0.0/16`), la envía por su ENI en la VPC y establece como siguiente salto el enrutador de la subred.

6. Enrutamiento de la VPC

La tabla de enrutamiento de la VPC contiene una entrada para el CIDR del nodo remoto (`10.80.0.0/16`), que dirige este tráfico a la puerta de enlace de la VPC a la red en las instalaciones.

5. Tránsito transfronterizo

La puerta de enlace transfiere el paquete a través de los límites de la nube mediante la conexión establecida (como Direct Connect o VPN) a la red en las instalaciones.

4. Recepción de la red en las instalaciones

El paquete llega a su enrutador en las instalaciones.

3. Entrega al nodo

El enrutador local identifica que la dirección IP (`10.80.0.2`) de destino pertenece a su red conectada directamente y reenvía el paquete directamente al nodo híbrido de destino.

2. Procesamiento de la red del nodo

A medida que la pila de red del nodo procesa el paquete, `conntrack` (una parte de `netfilter`) hace coincidir el paquete con la conexión que el pod estableció inicialmente y, dado que originalmente se utilizó la DNAT, invierte esta situación reescribiendo la IP de origen desde la IP de la ENI del plano de control de EKS a la IP del servicio de `kubernetes`:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
```

```
| Src: 172.16.0.1 | Src: 443           |                 |
| Dst: 10.80.0.2  | Dst: 67493         |                 |
+-----------------+--------------------+-----------------+
```

1. Procesamiento de la CNI

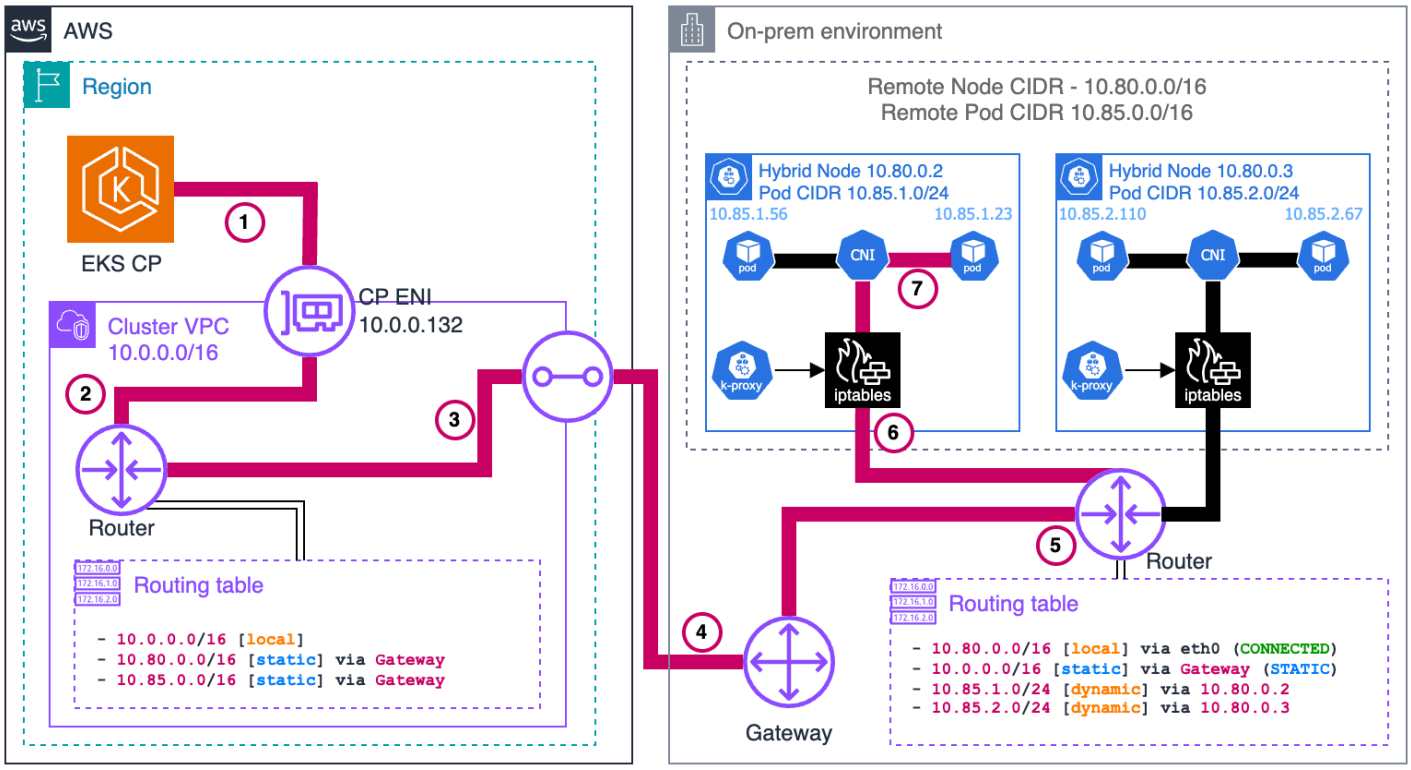
La CNI identifica que este paquete pertenece a una conexión en la que ha aplicado previamente la SNAT. Invierte la SNAT y vuelve a cambiar la IP de destino por la IP del pod:

```
+-----------------+--------------------+-----------------+
| IP Header       | TCP Header         | Payload         |
| Src: 172.16.0.1 | Src: 443           |                 |
| Dst: 10.85.1.56 | Dst: 67493         |                 |
+-----------------+--------------------+-----------------+
```

La CNI detecta que la IP de destino pertenece a un pod de su red y entrega el paquete al espacio de nombres de la red del pod correcto.

Este flujo muestra cómo la NAT de la CNI puede simplificar la configuración porque permite que los paquetes se enruten de vuelta al nodo sin requerir un enrutamiento adicional para los CIDR del pod.

Del plano de control de EKS a los pods que se ejecutan en un nodo híbrido (webhooks)



Este patrón de tráfico se observa con mayor frecuencia en los webhooks, donde el plano de control de EKS debe iniciar directamente las conexiones a los servidores de webhooks que se ejecutan en pods en nodos híbridos. Algunos ejemplos son la validación y la mutación de los webhooks de admisión, a los que el servidor de la API invoca durante los procesos de validación o mutación de recursos.

Solicitud

1. El servidor de la API de Kubernetes de EKS inicia la solicitud

Cuando se configura un webhook en el clúster y se activa una operación de API relevante, el servidor de la API de Kubernetes de EKS necesita establecer una conexión directa con el pod del servidor del webhook. El servidor de la API busca primero la dirección IP del pod en el recurso de servicio o punto de conexión asociado al webhook.

Si el pod del webhook se ejecuta en un nodo híbrido con la IP `10.85.1.23`, el servidor de la API de Kubernetes de EKS crea una solicitud HTTPS al punto de conexión del webhook. El paquete inicial se envía a través de la ENI del plano de control de EKS de la VPC porque la IP de destino `10.85.1.23` pertenece al CIDR del pod remoto configurado (`10.85.0.0/16`). El paquete tiene un aspecto similar al siguiente:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.0.0.132 | Src: 41892 (random) |                 |
| Dst: 10.85.1.23 | Dst: 8443           |                 |
+-----------------+---------------------+-----------------+
```

2. Procesamiento de redes de la VPC

El paquete sale de la ENI del plano de control de EKS y entra en la capa de red de la VPC con el enrutador de la subred como siguiente salto.

3. Búsqueda en la tabla de enrutamiento de la VPC

La tabla de enrutamiento de la VPC para la subred que contiene la ENI del plano de control de EKS tiene una ruta específica para el CIDR del pod remoto (`10.85.0.0/16`). Esta regla de enrutamiento dirige el paquete a la puerta de enlace de la VPC a local (por ejemplo, una puerta de enlace privada virtual para conexiones de Direct Connect o la VPN):

```
Destination     Target
10.0.0.0/16     local
```

```
10.85.0.0/16    vgw-id (VPC-to-onprem gateway)
```

4. Tránsito transfronterizo

La puerta de enlace transfiere el paquete a través de los límites de la nube mediante la conexión establecida (como Direct Connect o VPN) a la red en las instalaciones. El paquete mantiene sus direcciones IP de origen y destino originales a medida que atraviesa esta conexión.

5. Recepción de la red en las instalaciones

El paquete llega a su enrutador en las instalaciones. El enrutador consulta su tabla de enrutamiento para determinar cómo llegar a la dirección 10.85.1.23. Para que esto funcione, la red en las instalaciones debe tener rutas para los CIDR del pod que dirijan el tráfico al nodo híbrido correspondiente.

En este caso, la tabla de enrutamiento del enrutador contiene una entrada que indica que se puede acceder a la subred `10.85.1.0/24` a través del nodo híbrido con la IP `10.80.0.2`:

```
Destination     Next Hop
10.85.1.0/24    10.80.0.2
```

6. Entrega al nodo

Según la entrada de la tabla de enrutamiento, el enrutador reenvía el paquete al nodo híbrido (`10.80.0.2`). Cuando el paquete llega al nodo, tiene el mismo aspecto que cuando lo envió el servidor de la API de Kubernetes de EKS, pero la IP de destino sigue siendo la IP del pod.

7. Procesamiento de la CNI

La pila de red del nodo recibe el paquete y, al ver que la IP de destino no es la propia IP del nodo, lo pasa a la CNI para su procesamiento. La CNI identifica que la IP de destino pertenece a un pod que se ejecuta localmente en este nodo y reenvía el paquete al pod correcto a través de las interfaces virtuales adecuadas:

```
Original packet -> node routing -> CNI -> Pod's network namespace
```

El servidor del webhook del pod recibe la solicitud y la procesa.

Respuesta

Una vez que el pod del webhook procesa la solicitud, envía una respuesta siguiendo la misma ruta a la inversa:

7. El pod envía una respuesta

El pod del webhook crea un paquete de respuesta con su propia IP como origen y el solicitante original (la ENI del plano de control de EKS) como destino:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.85.1.23 | Src: 8443           |                 |
| Dst: 10.0.0.132 | Dst: 41892          |                 |
+-----------------+---------------------+-----------------+
```

El CNI identifica que este paquete se dirige a una red externa (no a un pod local) y lo entrega a la pila de red del nodo con la IP de origen original intacta.

6. Procesamiento de la red del nodo

El nodo determina que la IP de destino (`10.0.0.132`) no está en la red local y reenvía el paquete a su puerta de enlace predeterminada (el enrutador local).

5. Enrutamiento del enrutador local

El enrutador local consulta su tabla de enrutamiento y determina que la IP de destino (`10.0.0.132`) pertenece al CIDR de la VPC (`10.0.0.0/16`). Reenvía el paquete a la puerta de enlace que se conecta a AWS.

4. Tránsito transfronterizo

El paquete regresa a través de la misma conexión en las instalaciones a la VPC y cruza el límite de la nube en la dirección opuesta.

3. Enrutamiento de la VPC

Cuando el paquete llega a la VPC, las tablas de enrutamiento identifican que la IP de destino pertenece a una subred de la VPC. El paquete se enruta en consecuencia dentro de la VPC.

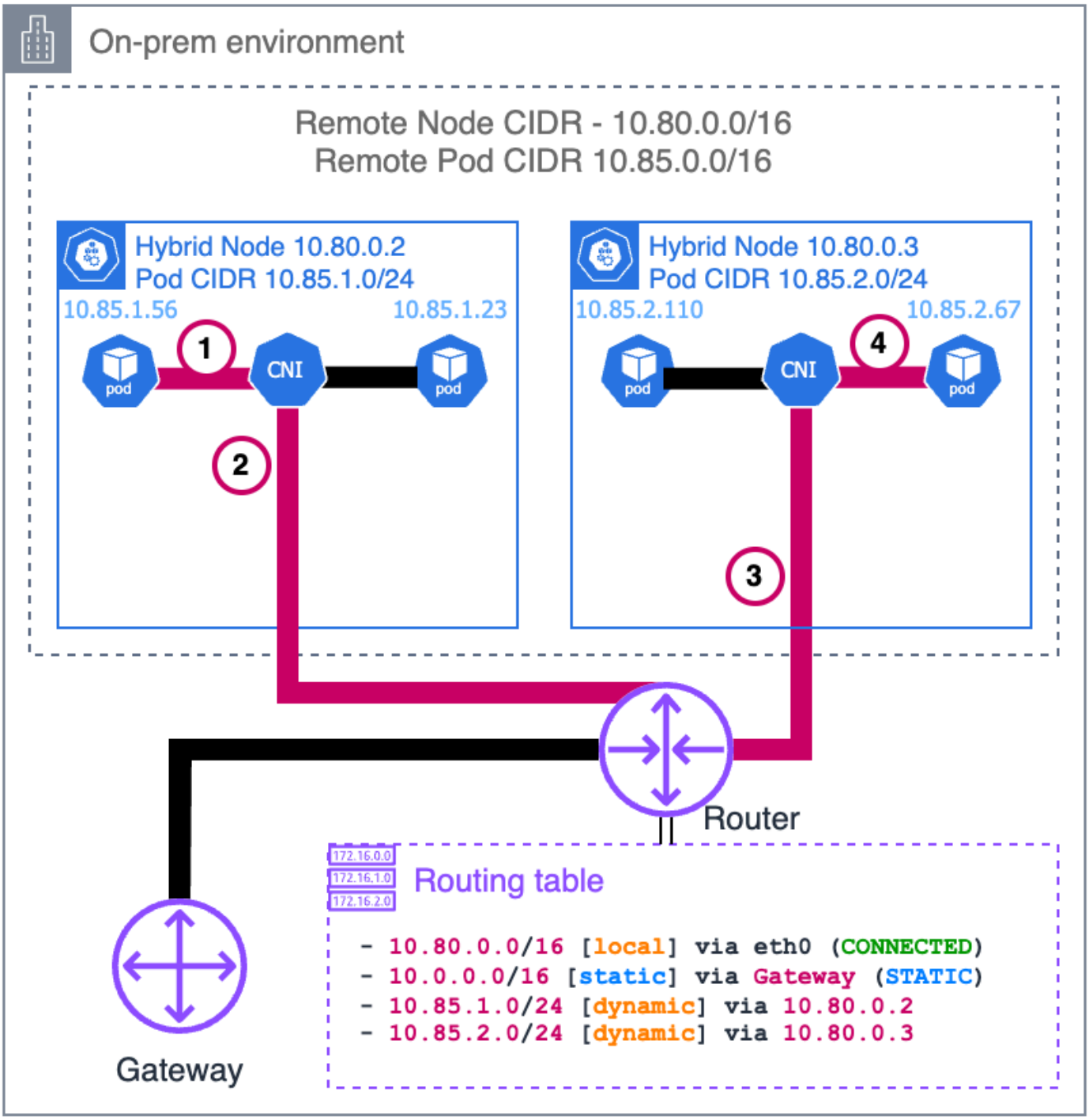
2. y 1. Recepción de la ENI del plano de control de EKS

El paquete llega a la ENI conectada al servidor de la API de Kubernetes de EKS y completa el viaje de ida y vuelta. El servidor de la API recibe la respuesta del webhook y continúa procesando la solicitud de la API original en función de esta respuesta.

Este flujo de tráfico demuestra por qué los CIDR de los pods remotos deben configurarse y enrutarse correctamente:

- La VPC debe tener rutas para los CIDR del pod remoto que apunten a la puerta de enlace en las instalaciones.
- La red en las instalaciones debe tener rutas para los CIDR de los pods que dirijan el tráfico a los nodos específicos que alojan esos pods.
- Sin esta configuración de enrutamiento, no se podría acceder a los webhooks y otros servicios similares que se ejecutan en pods en nodos híbridos desde el plano de control de EKS.

De pod a pod que se ejecuta en nodos híbridos



En esta sección, se explica cómo los pods que se ejecutan en diferentes nodos híbridos se comunican entre sí. En este ejemplo, se supone que su CNI utiliza VXLAN para la encapsulación, lo

que es habitual en CNI como Cilium o Calico. El proceso general es similar al de otros protocolos de encapsulación, como Geneve o IP-en-IP.

Solicitud

1. El pod A inicia la comunicación

El pod A (`10.85.1.56`) del nodo 1 quiere enviar tráfico al pod B (`10.85.2.67`) del nodo 2. El paquete inicial tiene un aspecto similar al siguiente:

```
+------------------+-----------------+-------------+-----------------+
| 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. La CNI intercepta y procesa el paquete

Cuando el paquete del pod A abandona el espacio de nombres de la red, la CNI lo intercepta. La CNI consulta su tabla de enrutamiento y determina: - La IP de destino (`10.85.2.67`) pertenece al CIDR del pod - Esta IP no está en el nodo local sino que pertenece al nodo 2 (`10.80.0.3`) - El paquete se debe encapsular con VXLAN.

La decisión de encapsular es fundamental porque la red física subyacente no sabe cómo enrutar los CIDR de los pods directamente, sino que solo sabe cómo enrutar el tráfico entre las IP de los nodos.

La CNI encapsula todo el paquete original dentro de una trama VXLAN. Esto crea efectivamente un “paquete dentro de un paquete” con nuevos encabezados:

```
+-----------------+---------------+-------------+-----------
+--------------------------+
| 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)
    |
+-----------------+---------------+-------------+-----------
+--------------------------+
```

Puntos clave sobre esta encapsulación: - El paquete externo se dirige del nodo 1 (`10.80.0.2`) al nodo 2 (`10.80.0.3`) - El puerto UDP `8472`es el puerto VXLAN que Cilium utiliza de forma

predeterminada - El identificador de red de VXLAN (VNI) identifica a qué red superpuesta pertenece este paquete - Todo el paquete original (con la IP del pod A como origen y la IP del pod B como destino) se conserva intacto en su interior

El paquete encapsulado entra ahora en la pila de red normal del nodo 1 y se procesa de la misma forma que cualquier otro paquete:

1. Procesamiento de red de nodos: la pila de red del nodo 1 enruta el paquete en función de su destino (`10.80.0.3`)
2. Entrega a través de la red local:
   - Si ambos nodos están en la misma red de capa 2, el paquete se envía directamente al nodo 2.
   - Si están en subredes diferentes, el paquete se reenvía primero al enrutador local.
3. Manejo del enrutador: el enrutador reenvía el paquete en función de su tabla de enrutamiento y lo entrega al nodo 2.

3. Procesamiento del nodo receptor

Cuando el paquete encapsulado llega al nodo 2 (`10.80.0.3`):

1. La pila de red del nodo lo recibe y lo identifica como un paquete VXLAN (puerto UDP 4789).
2. El paquete se pasa a la interfaz VXLAN de la CNI para su procesamiento.

4. Desencapsulación de VXLAN

La CNI del nodo 2 procesa el paquete VXLAN:

1. Elimina los encabezados externos (Ethernet, IP, UDP y VXLAN).
2. Extrae el paquete interno original.
3. El paquete ha vuelto a su forma original:

```
+------------------+-----------------+-------------+-----------------+
| 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   |                 |
+------------------+-----------------+-------------+-----------------+
```

La CNI del nodo 2 examina la IP de destino (`10.85.2.67`) y:

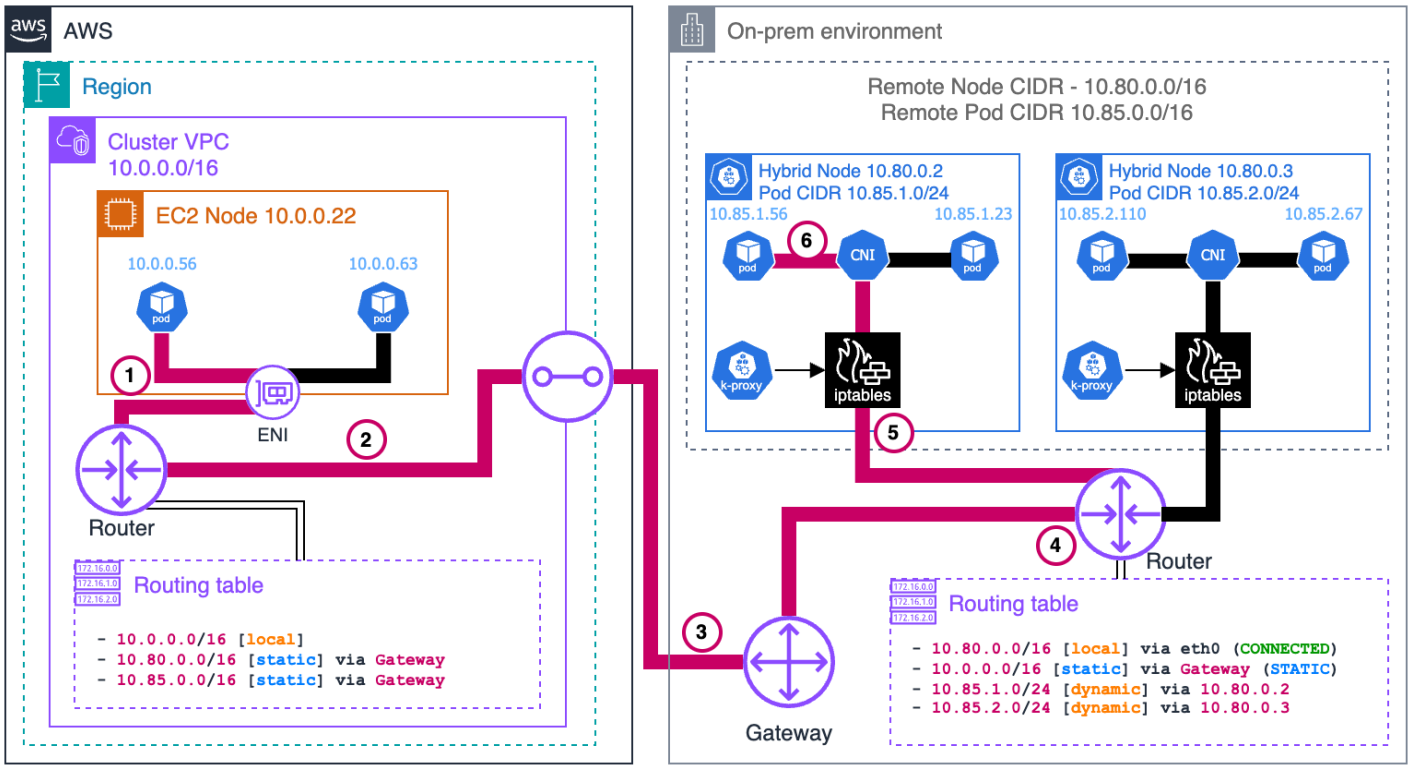
1. Identifica que esta IP pertenece a un pod local.
2. Enruta el paquete a través de las interfaces virtuales adecuadas.
3. Envía el paquete al espacio de nombres de red del Pod B.

Respuesta

Cuando el Pod B responde al Pod A, todo el proceso ocurre a la inversa:

4. El Pod B envía un paquete al Pod A (`10.85.1.56`)
5. La CNI del nodo 2 lo encapsula con VXLAN y establece el destino en el nodo 1 (`10.80.0.2`)
6. El paquete encapsulado se entrega al nodo 1.
7. La CNI del nodo 1 lo desencapsula y envía la respuesta original al pod A.

De pods de nodos en la nube a pods de nodos híbridos (tráfico de este a oeste)



Solicitud

1. El pod A inicia la comunicación

El pod A (`10.0.0.56`) del nodo EC2 quiere enviar tráfico al pod B (`10.85.1.56`) del nodo híbrido. El paquete inicial tiene un aspecto similar al siguiente:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.0.0.56  | Src: 52390 (random) |                 |
| Dst: 10.85.1.56 | Dst: 8080           |                 |
+-----------------+---------------------+-----------------+
```

Con la CNI de la VPC, el pod A tiene una IP del CIDR de la VPC y se conecta directamente a una ENI de la instancia EC2. El espacio de nombres de la red del pod está conectado a la red de la VPC, por lo que el paquete entra directamente en la infraestructura de enrutamiento de la VPC.

2. Enrutamiento de la VPC

La tabla de enrutamiento de la VPC contiene una ruta específica para el CIDR del pod remoto (`10.85.0.0/16`), que dirige este tráfico a la puerta de enlace de VPC a local:

```
Destination     Target
10.0.0.0/16     local
10.85.0.0/16    vgw-id (VPC-to-onprem gateway)
```

Según esta regla de enrutamiento, el paquete se dirige hacia la puerta de enlace que se conecta a la red en las instalaciones.

3. Tránsito transfronterizo

La puerta de enlace transfiere el paquete a través de los límites de la nube mediante la conexión establecida (como Direct Connect o VPN) a la red en las instalaciones. El paquete mantiene sus direcciones IP de origen y destino originales durante todo este tránsito.

4. Recepción de la red en las instalaciones

El paquete llega a su enrutador en las instalaciones. El enrutador consulta su tabla de enrutamiento para determinar el siguiente salto para llegar a la dirección 10.85.1.56. El enrutador en las instalaciones debe tener rutas para los CIDR del pod que dirijan el tráfico al nodo híbrido correspondiente.

La tabla del enrutador tiene una entrada que indica que la subred `10.85.1.0/24` es accesible a través del nodo híbrido con IP `10.80.0.2`:

```
Destination     Next Hop
10.85.1.0/24    10.80.0.2
```

5. Procesamiento de la red del nodo

El enrutador reenvía el paquete al nodo híbrido (`10.80.0.2`). Cuando el paquete llega al nodo, todavía tiene la IP del pod A como origen y la IP del pod B como destino.

6. Procesamiento de la CNI

La pila de red del nodo recibe el paquete y, al ver que la IP de destino no es la suya propia, lo pasa a la CNI para su procesamiento. La CNI identifica que la IP de destino pertenece a un pod que se ejecuta localmente en este nodo y reenvía el paquete al pod correcto a través de las interfaces virtuales adecuadas:

```
Original packet -> node routing -> CNI -> Pod B's network namespace
```

El pod B recibe el paquete y lo procesa según sea necesario.

Respuesta

6. El pod B envía una respuesta

El pod B crea un paquete de respuesta con su propia IP como origen y la IP del pod A como destino:

```
+-----------------+---------------------+-----------------+
| IP Header       | TCP Header          | Payload         |
| Src: 10.85.1.56 | Src: 8080           |                 |
| Dst: 10.0.0.56  | Dst: 52390          |                 |
+-----------------+---------------------+-----------------+
```

La CNI identifica que este paquete está destinado a una red externa y lo pasa a la pila de redes del nodo.

5. Procesamiento de la red del nodo

El nodo determina que la IP de destino (`10.0.0.56`) no pertenece a la red local y reenvía el paquete a su puerta de enlace predeterminada (el enrutador local).

4. Enrutamiento del enrutador local

El enrutador local consulta su tabla de enrutamiento y determina que la IP de destino (`10.0.0.56`) pertenece al CIDR de la VPC (`10.0.0.0/16`). Reenvía el paquete a la puerta de enlace que se conecta a AWS.

3. Tránsito transfronterizo

El paquete regresa a través de la misma conexión en las instalaciones a la VPC y cruza el límite de la nube en la dirección opuesta.

2. Enrutamiento de la VPC

Cuando el paquete llega a la VPC, el sistema de enrutamiento identifica que la IP de destino pertenece a una subred de la VPC. El paquete se enruta a través de la red de la VPC hacia la instancia EC2 que aloja el pod A.

1. El pod A recibe la respuesta

El paquete llega a la instancia EC2 y se entrega directamente al pod A a través de la ENI adjunta. Como la CNI de la VPC no utiliza redes superpuestas para los pods de la VPC, no es necesaria ninguna desencapsulación adicional: el paquete llega con sus encabezados originales intactos.

Este flujo de tráfico de este a oeste demuestra por qué los CIDR del pod remoto deben estar configurados correctamente y poder enrutarse desde ambas direcciones:

- La VPC debe tener rutas para los CIDR del pod remoto que apunten a la puerta de enlace en las instalaciones.
- La red en las instalaciones debe tener rutas para los CIDR del pod que dirijan el tráfico a los nodos específicos que alojan esos pods.

Referencia de **`nodeadm`** de nodos híbridos

La CLI (`nodeadm`) de los nodos híbridos de Amazon EKS simplifica la instalación, la configuración, el registro y la desinstalación de los componentes de los nodos híbridos. Puede incluir `nodeadm` en las imágenes de su sistema operativo para automatizar el arranque de los nodos híbridos; consulte the section called “Cómo preparar el sistema operativo” para obtener más información.

La versión `nodeadm` para los nodos híbridos es distinta de la versión de `nodeadm` utilizada para arrancar instancias de Amazon EC2 como nodos en clústeres de Amazon EKS. Siga la documentación y las referencias para obtener la versión de `nodeadm` adecuada. Esta página de documentación es para la versión `nodeadm` de los nodos híbridos.

El código fuente de `nodeadm` de los nodos híbridos se publica en el repositorio de GitHub https://github.com/aws/eks-hybrid.

> **Important**
>
> Debe ejecutar `nodeadm` con un usuario que tenga privilegios raíz/sudo.

Descarga de **`nodeadm`**

La versión de nodos híbridos de `nodeadm` está alojada en Amazon S3, con Amazon CloudFront como frontend. Para instalar `nodeadm` en cada host en las instalaciones, puede ejecutar el siguiente comando desde los hosts en las instalaciones.

Para los hosts x86_64

```
curl -OL 'https://hybrid-assets.eks.amazonaws.com/releases/latest/bin/linux/amd64/nodeadm'
```

Para los hosts ARM

```
curl -OL 'https://hybrid-assets.eks.amazonaws.com/releases/latest/bin/linux/arm64/nodeadm'
```

Agregue permiso de archivo ejecutable al binario descargado en cada host.

```
chmod +x nodeadm
```

`nodeadm install`

El comando `nodeadm install` se utiliza para instalar los artefactos y dependencias necesarios para ejecutar y unir nodos híbridos a un clúster de Amazon EKS. El comando `nodeadm install` se puede ejecutar de forma individual en cada nodo híbrido o se puede ejecutar durante las canalizaciones de creación de imágenes para preinstalar las dependencias de los nodos híbridos en las imágenes del sistema operativo.

Uso

```
nodeadm install [KUBERNETES_VERSION] [flags]
```

Argumentos de posición

(Obligatorio) `KUBERNETES_VERSION` La versión major.minor de Kubernetes de EKS que se va a instalar, por ejemplo `1.32`

Indicadores

Nombre	Obligatorio	Descripción
`-p`, `--credential-provider`	TRUE	Proveedor de credenciales para instalar. Los valores admitidos son `iam-ra` y `ssm`. Para obtener más información, consulte the section called "Cómo preparar las credenciales".
`-s`, `--containerd-source`	FALSO	Origen de `containerd` . `nodeadm` admite la instalación de `containerd` desde la distribución del sistema operativo, los paquetes de Docker y la omisión de `containerd` de la instalación. Valores `distro`: es el valor predeterminado. `nodeadm` instalará el paquete de `containerd` más reciente distribuido por el sistema operativo del nodo que sea compatible con la versión de Kubernetes de EKS. `distro` no es un valor compatible con los sistemas

Nombre	Obligatorio	Descripción
		operativos Red Hat Enterprise Linux (RHEL). `docker`: `nodeadm` instalará el paquete de `containerd` más reciente creado y distribuido por Docker que sea compatible con la versión de Kubernetes de EKS. `docker` no es un valor compatible con Amazon Linux 2023. `none`: `nodeadm` no instalará el paquete `containerd`. Debe instalar `containerd` manualmente antes de ejecutar `nodeadm init`.
`-r`, `--region`	FALSO	Especifica la región de AWS para descargar artefactos, como el agente SSM. El valor predeterminado es `us-west-2`.
`-t`, `--timeout`	FALSO	Duración máxima del comando de instalación. La entrada sigue el formato de duración. Por ejemplo: `1h23m`. El tiempo de espera de descarga predeterminado del domando de instalación está establecido en 20 minutos.

Nombre	Obligatorio	Descripción
`-h`, `--help`	FALSO	Muestra un mensaje de ayuda con los parámetros disponibles de indicadores, subcomandos y valores posicionales.

Ejemplos

Instale la versión `1.32` de Kubernetes con AWS Systems Manager (SSM) como proveedor de credenciales

```
nodeadm install 1.32 --credential-provider ssm
```

Instale la versión `1.32` de Kubernetes con AWS Systems Manager (SSM) como proveedor de credenciales y Docker como origen de containerd, con un tiempo de espera de descarga de 20 minutos.

```
nodeadm install 1.32 --credential-provider ssm --containerd-source docker --timeout 20m
```

Instale la versión `1.32` de Kubernetes con AWS IAM Roles Anywhere como proveedor de credenciales

```
nodeadm install 1.32 --credential-provider iam-ra
```

`nodeadm config check`

El comando `nodeadm config check` comprueba si hay errores en la configuración de nodo proporcionada. Este comando se puede utilizar para verificar un archivo de configuración de nodo híbrido y validar si es correcto.

Uso

```
nodeadm config check [flags]
```

Flags

Nombre	Obligatorio	Descripción
`-c`, `--config-source`	TRUE	Origen de la configuración de nodeadm. En el caso de los nodos híbridos, la entrada debe seguir un URI con un esquema de archivos.
`-h`, `--help`	FALSO	Muestra un mensaje de ayuda con los parámetros disponibl es de indicadores, subcomand os y valores posicionales.

Ejemplos

```
nodeadm config check -c file://nodeConfig.yaml
```

`nodeadm init`

El comando `nodeadm init` inicia y conecta el nodo híbrido con el clúster de Amazon EKS configurado. Consulte the section called “Configuración de nodo para activaciones híbridas de SSM” o the section called “Configuración de nodos para IAM Roles Anywhere” para obtener detalles sobre cómo configurar el archivo `nodeConfig.yaml`.

Uso

```
nodeadm init [flags]
```

Flags

Nombre	Obligatorio	Descripción
`-c`, `--config-source`	TRUE	Origen de la configuración de `nodeadm`. En el caso de los nodos híbridos, la entrada debe seguir un URI con un esquema de archivos.

Nombre	Obligatorio	Descripción
`-s`, `--skip`	FALSO	Las fases de `init` que se van a omitir. No se recomienda omitir ninguna de las fases a menos que ayude a solucionar un problema. Valores `install-validation` omite la verificación de si el comando de instalación anterior se ejecutó correctam ente. `cni-validation` omite la verificación de si los puertos VXLAN de las CNI Cilium o Calico están abiertos cuando el firewall está habilitado en el nodo. `node-ip-validation` omite la verificación de si la IP del nodo se encuentra dentro de un CIDR en las redes de nodos remotos
`-h`, `--help`	FALSO	Muestra un mensaje de ayuda con los parámetros disponibl es de indicadores, subcomand os y valores posicionales.

Ejemplos

```
nodeadm init -c file://nodeConfig.yaml
```

nodeadm upgrade

El comando `nodeadm upgrade` actualiza todos los artefactos instalados a la versión más reciente y arranca el nodo para configurar los artefactos actualizados y unirlos al clúster de EKS en AWS. La actualización es un comando que interrumpe las cargas de trabajo que se ejecutan en el nodo. Traslade las cargas de trabajo a otro nodo antes de ejecutar la actualización.

Uso

```
nodeadm upgrade [KUBERNETES_VERSION] [flags]
```

Argumentos de posición

(Obligatorio) `KUBERNETES_VERSION` La versión major.minor de Kubernetes de EKS que se va a instalar, por ejemplo `1.32`

Indicadores

Nombre	Obligatorio	Descripción
`-c`, `--config-source`	TRUE	Origen de la configuración de `nodeadm`. En el caso de los nodos híbridos, la entrada debe seguir un URI con un esquema de archivos.
`-t`, `--timeout`	FALSO	Tiempo de espera para descargar artefactos. La entrada sigue el formato de duración. Por ejemplo, 1h23m. El tiempo de espera de descarga predeterminado del domando de actualiza ción está establecido en 10 minutos.
`-s`, `--skip`	FALSO	Las fases de la actualización que se van a omitir. No se recomienda omitir ninguna de

Nombre	Obligatorio	Descripción
		las fases a menos que ayude a solucionar un problema. Valores `pod-validation` omite comprobar si todos los pods se ejecutan en el nodo, excepto los conjuntos de daemon y los pods estáticos. `node-validation` omite comprobar si el nodo ha sido acordonado. `init-validation` omite comprobar si el nodo se ha inicializado correctamente antes de ejecutar la actualización. `containerd-major-version-upgrade` impide las actualizaciones de las versiones principales de containerd durante la actualización del nodo.
`-h`, `--help`	FALSO	Muestra un mensaje de ayuda con los parámetros disponibles de indicadores, subcomandos y valores posicionales.

Ejemplos

```
nodeadm upgrade 1.32 -c file://nodeConfig.yaml
```

```
nodeadm upgrade 1.32 -c file://nodeConfig.yaml --timeout 20m
```

nodeadm uninstall

El comando `nodeadm uninstall` detiene y elimina los artefactos que `nodeadm` instala durante la `nodeadm install`, incluidos el kubelet y el containerd. Tenga en cuenta que el comando de desinstalación no vacía ni elimina los nodos híbridos del clúster. Debe ejecutar las operaciones de vaciado y eliminación por separado. Consulte the section called “Cómo eliminar nodos híbridos” para obtener más información. De forma predeterminada, `nodeadm uninstall` no procederá si quedan pods en el nodo. Del mismo modo, `nodeadm uninstall` no elimina las dependencias del CNI ni las dependencias de otros complementos de Kubernetes que ejecute en el clúster. Para eliminar por completo la instalación de CNI del host, consulte las instrucciones que aparecen en the section called “Cómo configurar una CNI”. Si utiliza activaciones híbridas de AWS SSM como proveedor de credenciales en las instalaciones, el comando `nodeadm uninstall` anula el registro de los hosts como instancias administradas por AWS SSM.

Uso

```
nodeadm uninstall [flags]
```

Flags

Nombre	Obligatorio	Descripción
`-s`, `--skip`	FALSO	Las fases de desinstalación que se van a omitir. No se recomienda omitir ninguna de las fases a menos que ayude a solucionar un problema. Valores `pod-validation` omite comprobar si todos los pods se ejecutan en el nodo, excepto los conjuntos de daemon y los pods estáticos.

Nombre	Obligatorio	Descripción
		`node-validation` omite comprobar si el nodo ha sido acordonado. `init-validation` omite comprobar si el nodo se ha inicializado correctamente antes de ejecutar la desinstal ación.
`-h`, `--help`	FALSO	Muestra un mensaje de ayuda con los parámetros disponibl es de indicadores, subcomand os y valores posicionales.

Nombre	Obligatorio	Descripción
`-f`, `--force`	FALSO	Fuerce la eliminación de directorios adicionales que puedan contener archivos restantes de los componentes de Kubernetes y CNI. ADVERTENCIA Esto eliminará todo el contenido de los directori os predeterminados de Kubernetes y CNI (`/var/lib/cni` , `/etc/cni/net.d` , etc.). No utilice esta marca si almacena sus propios datos en estas ubicaciones. A partir de nodeadm `v1.0.9`, el comando `./nodeadm uninstall --skip node-validation,pod-validation --force` ya no elimina el directorio `/var/lib/kubelet` . Esto se debe a que puede contener volúmenes de Pod y directori os de subrutas de volúmenes que, a veces, incluyen el sistema de archivos del nodo montado. Consejos para un manejo seguro

Nombre	Obligatorio	Descripción
		- La eliminación de rutas montadas puede llevar a la eliminación accidental del sistema de archivos del nodo montado real. Antes de eliminar manualmente el directorio `/var/lib/kubelet` , inspeccione cuidadosamente todos los montajes activos y desmonte los volúmenes de forma segura para evitar la pérdida de datos.

Ejemplos

```
nodeadm uninstall
```

```
nodeadm uninstall --skip node-validation,pod-validation
```

`nodeadm debug`

El comando `nodeadm debug` se puede utilizar para solucionar problemas de nodos híbridos mal configurados o en mal estado. Valida que se cumplan los siguientes requisitos.

- El nodo tiene acceso de red a las API de AWS necesarias para obtener las credenciales;
- El nodo puede obtener credenciales de AWS para el rol de IAM de nodos híbridos configurado;
- El nodo tiene acceso de red al punto de conexión de la API de Kubernetes de EKS y la validez del certificado de punto de conexión de la API de Kubernetes de EKS;
- El nodo se puede autenticar con el clúster de EKS; su identidad en el clúster es válida; y cuenta con acceso al clúster de EKS a través de la VPC configurada para el clúster de EKS.

Si se encuentran errores, el resultado del comando sugiere los pasos para solucionar el problema. Algunos pasos de validación muestran los procesos secundarios. Si fallan, el resultado aparece en una sección stderr debajo del error de validación.

Uso

```
nodeadm debug [flags]
```

Flags

Nombre	Obligatorio	Descripción
`-c`, `--config-source`	TRUE	Origen de la configuración de `nodeadm`. En el caso de los nodos híbridos, la entrada debe seguir un URI con un esquema de archivos.
`--no-color`	FALSO	Desactiva la salida en color. Útil para la automatización.
`-h`, `--help`	FALSO	Muestra un mensaje de ayuda con los parámetros disponibl es de indicadores, subcomand os y valores posicionales.

Ejemplos

```
nodeadm debug -c file://nodeConfig.yaml
```

Ubicaciones de archivos de nodeadm

instalación de nodeadm

Cuando se ejecuta `nodeadm install`, se configuran los siguientes archivos y ubicaciones de archivos.

Artefacto	Ruta
CLI de IAM Roles Anywhere	/usr/local/bin/aws_signing_helper
Kubelet binario	/usr/bin/kubelet
Kubectl binario	usr/local/bin/kubectl
Proveedor de credenciales de ECR	/etc/eks/image-credential-provider/ecr-credential-provider
Autenticador de AWS IAM	/usr/local/bin/aws-iam-authenticator
CLI de configuración de SSM	/opt/ssm/ssm-setup-cli
SSM Agent	En Ubuntu: /snap/amazon-ssm-agent/current/amazon-ssm-agent En RHEL y AL2023: /usr/bin/amazon-ssm-agent
Containerd	En Ubuntu y AL2023: /usr/bin/containerd En RHEL: /bin/containerd
Iptables	En Ubuntu y AL2023: /usr/sbin/iptables En RHEL: /sbin/iptables
Complementos de CNI	/opt/cni/bin
rastreador de artefactos instalados	/opt/nodeadm/tracker

nodeadm init

Cuando se ejecuta `nodeadm init`, se configuran los siguientes archivos y ubicaciones de archivos.

Nombre	Ruta
kubeconfig de Kubelet	/var/lib/kubelet/kubeconfig

Nombre	Ruta
Configuración de Kubelet	/etc/kubernetes/kubelet/config.json
Unidad systemd de Kubelet	/etc/systemd/system/kubelet.service
Configuración del proveedor de credenciales de imágenes	/etc/eks/image-credential-provider/config.json
Archivo env de Kubelet	/etc/eks/kubelet/environment
Certificados de Kubelet	/etc/kubernetes/pki/ca.crt
Configuración de containerd	/etc/containerd/config.toml
Configuración de los módulos del núcleo de containerd	/etc/modules-load.d/contianerd.conf
AWSArchivo de configuración de	/etc/aws/hybrid/config
Archivo de credenciales de AWS (si está habilitado el archivo de credenciales)	/eks-hybrid/.aws/credentials
Unidad de sistema auxiliar de firma de AWS	/etc/systemd/system/aws_signing_helper_update.service
Archivo conf de Sysctl	/etc/sysctl.d/99-nodeadm.conf
Ca-certificates	/etc/ssl/certs/ca-certificates.crt
Archivo de clave de Gpg	/etc/apt/keyrings/docker.asc
Archivo de origen del repositorio de Docker	/etc/apt/sources.list.d/docker.list

Configuración de nodo para activaciones híbridas de SSM

El siguiente es un `nodeConfig.yaml` de ejemplo al utilizar activaciones híbridas de AWS SSM para credenciales de nodos híbridos.

```
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
```

Configuración de nodos para IAM Roles Anywhere

El siguiente es un `nodeConfig.yaml` de ejemplo para nodos híbridos de AWS IAM Roles Anywhere.

Si utiliza AWS IAM Roles Anywhere como proveedor de credenciales en las instalaciones, el `nodeName` que utilice en la configuración de `nodeadm` debe coincidir con los permisos que haya definido para el rol de IAM de nodos híbridos. Por ejemplo, si los permisos para el rol de IAM de nodos híbridos solo permiten que AWS IAM Roles Anywhere asuma el rol cuando el nombre de la sesión del rol es igual al CN del certificado de host, entonces el `nodeName` de la configuración de `nodeadm` debe ser el mismo que el CN de los certificados. El `nodeName` que utilice no puede tener más de 64 caracteres. Para obtener más información, consulte the section called “Cómo preparar las credenciales”.

```
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
```

Configuración de nodos para personalizar kubelet (opcional)

Puede transmitir la configuración y los indicadores de kubelet en la configuración de `nodeadm`. Consulte el ejemplo que aparece a continuación sobre cómo agregar una etiqueta de nodo adicional `abc.amazonaws.com/test-label` y configurar para establecer `shutdownGracePeriod` en 30 segundos.

```
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
```

Configuración de nodos para personalizar containerd (opcional)

Puede transmitir la configuración de containerd personalizada en la configuración de `nodeadm`. La configuración de containerd para `nodeadm` acepta TOML en línea. Consulte el ejemplo que aparece a continuación para ver cómo configurar containerd para desactivar la eliminación de capas de imágenes desempaquetadas en el almacén de contenido de containerd.

```
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**
>
> Las versiones 1.x y 2.x de containerd utilizan diferentes formatos de configuración. containerd 1.x usa la versión de configuración 2, mientras que containerd 2.x usa la versión de configuración 3. Aunque containerd 2.x sigue siendo compatible con versiones anteriores de la versión de configuración 2, se recomienda la versión de configuración 3 para obtener un rendimiento óptimo. Compruebe la versión de containerd con `containerd --version` o consulte los registros de instalación de `nodeadm`. Para obtener más información sobre el control de versiones de configuración, consulte https://containerd.io/releases/

Además, puede utilizar la configuración de containerd para habilitar la compatibilidad con SELinux. Con SELinux habilitado en containerd, asegúrese de que los pods programados en el nodo tengan habilitados securityContext y seLinuxOptions adecuados. Puede encontrar más información sobre la configuración de un contexto de seguridad en la documentación de Kubernetes.

> **Note**
>
> Red Hat Enterprise Linux (RHEL) 8 y RHEL 9 tienen SELinux habilitado de forma predeterminada y establecido en el modo estricto en el host. De forma predeterminada, SELinux está habilitado y configurado en modo permisivo en Amazon Linux 2023. Cuando SELinux está configurado en modo permisivo en el host, al activarlo en containerd no se bloquearán las solicitudes, sino que se hará el registro de acuerdo con la configuración de SELinux en el 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
```

Solución de problemas relacionados con los nodos híbridos

En este tema se tratan algunos errores habituales que pueden aparecer al utilizar los Nodos híbridos de Amazon EKS y cómo solucionarlos. Para obtener información adicional sobre la solución de problemas, consulte Solución de problemas la Etiqueta del Centro de conocimientos para Amazon EKS en AWS re:Post. Si no puede resolver el problema, contacte a AWS Support.

Solución de problemas del nodo con **`nodeadm debug`**: puede ejecutar el comando `nodeadm debug` desde los nodos híbridos para comprobar que se cumplan los requisitos de red y credenciales. Para obtener más información acerca del comando `nodeadm debug`, consulte the section called “Nodeadm de nodos híbridos”.

Detección de problemas con los nodos híbridos mediante la información sobre clústeres: la información sobre clústeres de Amazon EKS incluye comprobaciones de información que detectan problemas comunes con la configuración de los nodos híbridos de EKS en el clúster. Puede ver los resultados de todas las comprobaciones de información desde la Consola de administración de AWS, la AWS CLI y los AWS SDK. Para obtener más detalles acerca de la información del clúster, consulte the section called “Información sobre clústeres”.

Solución de problemas de instalación de nodos híbridos

Los siguientes temas de solución de problemas están relacionados con la instalación de las dependencias de los nodos híbridos en los hosts mediante el comando `nodeadm install`.

`nodeadm` El comando falló **`must run as root`**

El comando `nodeadm install` se debe ejecutar con un usuario que tenga privilegios de raíz o `sudo` en el host. Si ejecuta `nodeadm install` con un usuario que no tiene privilegios de raíz o `sudo`, el siguiente error aparecerá en el resultado de `nodeadm`.

```
"msg":"Command failed","error":"must run as root"
```

No se puede conectar a las dependencias

El comando `nodeadm install` instala las dependencias necesarias para los nodos híbridos. Algunas de las dependencias de los nodos híbridos son `containerd`, `kubelet`, `kubectl` y AWS SSM o componentes de AWS IAM Roles Anywhere. Debe tener acceso desde el lugar donde ejecuta `nodeadm install` para descargar estas dependencias. Para obtener más información sobre la lista de ubicaciones a las que debe tener acceso, consulte the section called “Preparación de las redes”. Si no tiene acceso, se producirán errores similares a los siguientes en el resultado de `nodeadm install`.

```
"msg":"Command failed","error":"failed reading file from url: ...: max retries achieved
 for http request"
```

No se pudo actualizar el administrador de paquetes

El comando `nodeadm install` ejecuta `apt update`, `yum update` o `dnf update` antes de instalar las dependencias de los nodos híbridos. Si este paso no se realiza correctamente, es posible que se produzcan errores similares a los siguientes. Para solucionarlo, puede ejecutar `apt update`, `yum update` o `dnf update` antes de ejecutar `nodeadm install`. De otro modo, puede intentar volver a ejecutar `nodeadm install`.

```
failed to run update using package manager
```

Se ha superado el tiempo de espera o el plazo de contexto

Al ejecutar `nodeadm install`, si observa problemas en distintas etapas del proceso de instalación con errores que indiquen un tiempo de espera agotado o que se superó el plazo de contexto, es posible que tenga una conexión lenta que impide la instalación de las dependencias de los nodos híbridos antes de que se cumplan los tiempos de espera. Para solucionar estos problemas, puede intentar utilizar la marca `--timeout` en `nodeadm` para prolongar los tiempos de espera para descargar las dependencias.

```
nodeadm install K8S_VERSION --credential-provider CREDS_PROVIDER --timeout 20m0s
```

Solución de problemas de conexión de nodos híbridos

Los temas de solución de problemas de esta sección están relacionados con el proceso de conexión de nodos híbridos a clústeres de EKS mediante el comando `nodeadm init`.

Errores de operaciones o esquemas no compatibles

Si al ejecutar `nodeadm init` aparecen errores relacionados con `operation error` o `unsupported scheme`, compruebe `nodeConfig.yaml` para garantizar que tenga el formato correcto y que se ha transmitido a `nodeadm`. Para obtener más información sobre el formato y las opciones para `nodeConfig.yaml`, consulte the section called “Nodeadm de nodos híbridos”.

```
"msg":"Command failed","error":"operation error ec2imds: GetRegion, request canceled,
 context deadline exceeded"
```

El rol de IAM de los Nodos híbridos carece de permisos para la acción **`eks:DescribeCluster`**

Cuando se ejecuta `nodeadm init`, `nodeadm` intenta recopilar información sobre el clúster de EKS mediante una llamada a la acción `DescribeCluster` de EKS. Si el rol de IAM de Nodos híbridos no tiene permiso para la acción `eks:DescribeCluster`, deberá transmitir el punto de conexión de la API de Kubernetes, el paquete de CA del clúster y el CIDR IPv4 del servicio a la configuración de nodos que transmite a `nodeadm` al ejecutar `nodeadm init`. Para obtener más información sobre los permisos necesarios para el rol de IAM de los Nodos híbridos, consulte the section called “Cómo preparar las credenciales”.

```
"msg":"Command failed","error":"operation error EKS: DescribeCluster, https response
 error StatusCode: 403 ... AccessDeniedException"
```

El rol de IAM de los Nodos híbridos carece de permisos para la acción **`eks:ListAccessEntries`**

Cuando se ejecuta `nodeadm init`, `nodeadm` intenta validar si el clúster de EKS tiene una entrada de acceso del tipo `HYBRID_LINUX` asociada al rol de IAM de Nodos híbridos mediante una llamada a la acción `ListAccessEntries` de EKS. Si el rol de IAM de Nodos híbridos no tiene permiso para la acción `eks:ListAccessEntries`, debe transmitir el indicador `--skip cluster-access-validation` al ejecutar el comando `nodeadm init`. Para obtener más información sobre los permisos necesarios para el rol de IAM de los Nodos híbridos, consulte the section called “Cómo preparar las credenciales”.

```
"msg":"Command failed","error":"operation error EKS: ListAccessEntries, https response
 error StatusCode: 403 ... AccessDeniedException"
```

La IP del nodo no está dentro del CIDR de la red del nodo remoto

Al ejecutar `nodeadm init`, se podría producir un error si la dirección IP del nodo no se encuentra dentro de los CIDR de red de nodos remotos especificados. El error tendrá un aspecto similar al siguiente ejemplo:

```
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]
```

Este ejemplo muestra un nodo con la IP 10.18.0.1 que intenta unirse a un clúster con CIDR de red remota 10.0.0.0/16 y 192.168.0.0/16. El error se produce porque la IP 10.18.0.1 no se encuentra dentro de ninguno de los rangos.

Confirme que haya configurado correctamente las `RemoteNodeNetworks` para incluir todas las direcciones IP de los nodos. Para obtener más información sobre la configuración de la red, consulte the section called “Preparación de las redes”.

- Ejecute el siguiente comando en la región en la que se encuentra el clúster para comprobar las configuraciones de `RemoteNodeNetwork`. Verifique que los bloques de CIDR que aparecen en la salida incluyan el rango de IP del nodo y coincidan con los bloques de CIDR mencionados en el mensaje de error. Si no coinciden, confirme que el nombre del clúster y la región especificados en el archivo `nodeConfig.yaml` correspondan al clúster previsto.

```
aws eks describe-cluster --name CLUSTER_NAME --region REGION_NAME --query
 cluster.remoteNetworkConfig.remoteNodeNetworks
```

- Verifique que el nodo sea el previsto:
  - Para confirmar que se trata del nodo correcto, verifique que el nombre de host y la dirección IP coincidan con los que desea registrar en el clúster.
  - Confirme que este nodo se encuentra en la red en las instalaciones correcta (aquella cuyo rango de CIDR se registró como `RemoteNodeNetwork` durante la configuración del clúster).

Si la IP del nodo aún no es la esperada, verifique lo siguiente:

- Si utiliza IAM Roles Anywhere, `kubelet` realiza una consulta de DNS sobre el `nodeName` de IAM Roles Anywhere y usa una IP asociada con el nombre de nodo si está disponible. Si dispone de entradas de DNS para los nodos, confirme que dichas entradas apunten a direcciones IP que se encuentren dentro de los CIDR de la red de nodos remotos.
- Si el nodo tiene múltiples interfaces de red, `kubelet` podría elegir como predeterminada una interfaz cuya dirección IP no esté dentro de los CIDR de la red de nodos remotos. Para utilizar una interfaz diferente, especifique su dirección IP con la marca `--node-ip` `kubelet` en el archivo `nodeConfig.yaml`. Para obtener más información, consulte the section called “Nodeadm de

nodos híbridos". Para ver las interfaces de red del nodo y sus direcciones IP, ejecute el siguiente comando en el nodo:

```
ip addr show
```

Los nodos híbridos no aparecen en el clúster de EKS

Si ejecutó `nodeadm init` y se completó, pero los nodos híbridos no aparecen en el clúster, es posible que haya problemas con la conexión de red entre los nodos híbridos y el plano de control de EKS, que no tenga configurados los permisos de grupo de seguridad necesarios o que no tenga la asignación necesaria del rol de IAM de nodos híbridos al control de acceso basado en roles (RBAC) de Kubernetes. Para iniciar el proceso de depuración, puede utilizar los siguientes comandos para comprobar el estado de `kubelet` y los registros de `kubelet`. Ejecute los siguientes comandos desde los nodos híbridos que no se pudieron unir al clúster.

```
systemctl status kubelet
```

```
journalctl -u kubelet -f
```

No se puede establecer comunicación con el clúster

Si el nodo híbrido no se pudo comunicar con el plano de control del clúster, es posible que se produzcan registros similares a los siguientes.

```
"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
```

Si aparecen estos mensajes, compruebe lo siguiente para asegurarse de que cumple con los requisitos de los nodos híbridos que se indican en the section called "Preparación de las redes".

- Confirme que la VPC trasladada al clúster de EKS tenga rutas a la puerta de enlace de tránsito (TGW) o puerta de enlace privada virtual (VGW) para el nodo en las instalaciones y, opcionalmente, los CIDR del pod.

- Confirme que cuenta con un grupo de seguridad adicional para el clúster de EKS que tenga reglas de entrada para los CIDR de los nodos en las instalaciones y, opcionalmente, los CIDR de los pods.
- Confirme que el enrutador en las instalaciones esté configurado para permitir el tráfico hacia y desde el plano de control de EKS.

No autorizado

Si el nodo híbrido se pudo comunicar con el plano de control de EKS pero no se pudo registrar, es posible que vea registros similares a los siguientes. Tenga en cuenta que la diferencia clave en los mensajes de registro que aparecen a continuación es el error `Unauthorized`. Esto indica que el nodo no pudo realizar sus tareas porque no tiene los permisos necesarios.

```
"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
```

Si aparecen estos mensajes, compruebe lo siguiente para asegurarse de que cumple con los requisitos de los nodos híbridos que se indican en the section called “Cómo preparar las credenciales” y the section called “Cómo preparar el acceso al clúster”.

- Confirme que la identidad de los nodos híbridos coincide con el rol de IAM previsto para los nodos híbridos. Para ello, se puede ejecutar `sudo aws sts get-caller-identity` desde los nodos híbridos.
- Compruebe que el rol de IAM de los Nodos híbridos tiene los permisos necesarios.
- Confirme que en el clúster tiene una entrada de acceso a EKS para el rol de IAM de los Nodos híbridos o confirme que el ConfigMap de `aws-auth` tiene una entrada para el rol de IAM de los Nodos híbridos. Si utiliza entradas de acceso a EKS, confirme que la entrada de acceso para el rol de IAM de los Nodos híbridos sea del tipo de acceso `HYBRID_LINUX`. Si utiliza ConfigMap de `aws-auth`, confirme que la entrada para el rol de IAM de los Nodos híbridos cumpla con los requisitos y el formato que se detallan en the section called “Cómo preparar el acceso al clúster”.

Los nodos híbridos están registrados en el clúster de EKS, pero aparecen en estado **`Not Ready`**

Si los nodos híbridos se registraron correctamente en el clúster de EKS, pero los nodos híbridos aparecen en estado `Not Ready`, lo primero que hay que comprobar es el estado de la interfaz de red de contenedores (CNI). Si no ha instalado una CNI, lo normal es que los nodos híbridos se encuentren en estado `Not Ready`. Una vez que una CNI se instala y se ejecuta correctamente, los nodos se actualizan al estado `Ready`. Si ha intentado instalar una CNI, pero no se ejecuta correctamente, consulte the section called “Solución de problemas de la CNI de nodos híbridos” en esta página.

Las solicitudes de firma de certificados (CSR) están atascadas en estado Pendiente

Después de conectar nodos híbridos al clúster de EKS, si ve que hay CSR pendientes correspondientes a los nodos híbridos, significa que los nodos híbridos no cumplen los requisitos para la aprobación automática. Las CSR para nodos híbridos se aprueban y emiten automáticamente si las CSR para nodos híbridos fueron creadas por un nodo con etiqueta `eks.amazonaws.com/compute-type: hybrid`, y la CSR tiene los siguientes nombres alternativos de asunto (SAN): al menos un SAN de DNS igual al nombre del nodo y los SAN de IP pertenecen a los CIDR de la red de nodos remotos.

Ya existe un perfil híbrido

Si ha cambiado la configuración de `nodeadm` e intenta volver a registrar el nodo con la nueva configuración, es posible que aparezca un error que indica que el perfil híbrido ya existe, pero su contenido ha cambiado. En lugar de ejecutar `nodeadm init` entre los cambios de configuración, ejecute `nodeadm uninstall` seguido de `nodeadm install` y `nodeadm init`. Esto garantiza una limpieza adecuada con los cambios en la configuración.

```
"msg":"Command failed","error":"hybrid profile already exists at /etc/aws/hybrid/config
 but its contents do not align with the expected configuration"
```

El nodo híbrido no pudo resolver la API privada

Después de ejecutar `nodeadm init`, si aparece un error en los registros de `kubelet` que indique que no se ha establecido contacto con el servidor de la API de Kubernetes de EKS debido a que `no such host`, es posible que tenga que cambiar la entrada de DNS del punto de conexión de la API de Kubernetes de EKS en la red en las instalaciones o a nivel de host. Consulte Reenvío de consultas de DNS entrantes a la VPC en la documentación de AWS Route53.

```
Failed to contact API server when waiting for CSINode publishing: Get ... no such host
```

No se pueden ver los nodos híbridos en la consola de EKS

Si ha registrado los nodos híbridos, pero no puede verlos en la consola de EKS, compruebe los permisos de la entidad principal de IAM que utiliza para ver la consola. La entidad principal de IAM que utiliza debe tener permisos mínimos específicos de IAM y Kubernetes para ver los recursos de la consola. Para obtener más información, consulte the section called “Acceso a recursos del clúster”.

Solución de problemas de ejecución de nodos híbridos

Si los nodos híbridos registrados en el clúster de EKS se encontraban en estado `Ready` y, posteriormente, pasaron al estado `Not Ready`, existe una amplia gama de problemas que pueden haber contribuido al mal estado, como que el nodo carezca de recursos suficientes para la CPU, la memoria o el espacio en disco disponible, o que el nodo esté desconectado del plano de control de EKS. Puede seguir los pasos que se indican a continuación para solucionar los problemas de los nodos y, si no puede resolver el problema, contacte a AWS Support.

Ejecute `nodeadm debug` desde los nodos híbridos para comprobar que se cumplen los requisitos de red y credenciales. Para obtener más información acerca del comando `nodeadm debug`, consulte the section called “Nodeadm de nodos híbridos”.

Obtención del estado de los nodos

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

Comprobación de las condiciones y los eventos de nodos

```
kubectl describe node NODE_NAME
```

Obtención del estado de los pods

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

Comprobación de las condiciones y los eventos de pods

```
kubectl describe pod POD_NAME
```

Comprobación de los registros de pods

```
kubectl logs POD_NAME
```

Comprobar los registros de **`kubectl`**

```
systemctl status kubelet
```

```
journalctl -u kubelet -f
```

Las sondas de actividad del pod fallan o los webhooks no funcionan

Si las aplicaciones, complementos o webhooks que se ejecutan en los nodos híbridos no se inician correctamente, es posible que existan problemas de red que bloqueen la comunicación con los pods. Para que el plano de control de EKS se ponga en contacto con los webhooks que se ejecutan en nodos híbridos, debe configurar el clúster de EKS con una red de pods remotos y disponer de rutas para el CIDR de pods en las instalaciones en la tabla de enrutamiento de VPC con el destino como la puerta de enlace de tránsito (TGW), la puerta de enlace virtual privada (VPW) u otra puerta de enlace que utilice para conectar la VPC con la red en las instalaciones. Para obtener más información sobre los requisitos de red para los nodos híbridos, consulte the section called “Preparación de las redes”. Además, debe permitir este tráfico en el firewall en las instalaciones y asegurarse de que el enrutador puede dirigir correctamente el tráfico a los pods. Consulte the section called “Configuración de webhooks” para obtener más información sobre los requisitos para ejecutar webhooks en nodos híbridos.

A continuación aparece un mensaje de registro de pod común para este escenario, en el que ip-address es la IP del clúster del servicio de Kubernetes.

```
dial tcp <ip-address>:443: connect: no route to host
```

Los comandos **`kubectl logs`** o **`kubectl exec`** no funcionan (comandos de la API de **`kubelet`**)

Si los comandos `kubectl attach`, `kubectl cp`, `kubectl exec`, `kubectl logs` y `kubectl port-forward` agotan el tiempo de espera mientras otros comandos `kubectl` se ejecutan

correctamente, es probable que el problema esté relacionado con la configuración de la red remota. Estos comandos se conectan a través del clúster al punto de conexión de `kubelet` en el nodo. Para obtener más información consulte () the section called “Punto de conexión de `kubelet`”.

Verifique que las direcciones IP de los nodos y pods se encuentren dentro de los CIDR de la red de nodos remotos y de la red de pods remotos configurados para el clúster. Use los siguientes comandos para examinar las asignaciones de IP.

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

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

Compare estas direcciones IP con los CIDR de red remota configurados para asegurarse de que el enrutamiento sea correcto. Para consultar los requisitos de configuración de red, consulte the section called “Preparación de las redes”.

Solución de problemas de la CNI de nodos híbridos

Si tiene problemas al iniciar Cilium o Calico al principio con nodos híbridos, la mayoría de las veces se debe a problemas de red entre los nodos híbridos o los pods de la CNI que se ejecutan en los nodos híbridos y el plano de control de EKS. Asegúrese de que el entorno cumpla los requisitos que se indican en Cómo preparar las redes para los nodos híbridos. Resulta útil dividir el problema en partes.

Configuración del clúster de EKS

- ¿Son correctas las configuraciones de RemoteNodeNetwork y RemotePodNetwork?

Configuración de la VPC

- ¿Existen rutas para RemoteNodeNetwork y RemotePodNetwork en la tabla de enrutamiento de la VPC con el destino de la puerta de enlace de tránsito o la puerta de enlace virtual privada?

Configuración del grupo de seguridad

- ¿Existen reglas de entrada y salida para RemoteNodeNetwork y RemotePodNetwork?

Red on-premise

- ¿Existen rutas y acceso hacia y desde el plano de control de EKS, así como hacia y desde los nodos híbridos y los pods que se ejecutan en estos?

Configuración de la CNI

Si se utiliza una red superpuesta, ¿la configuración del grupo de IP de la CNI coincide la RemotePodNetwork configurada para el clúster de EKS si se utilizan webhooks?

El nodo híbrido se encuentra en estado **`Ready`** sin una CNI instalada

Si los nodos híbridos se encuentran en el estado `Ready`, pero no ha instalado una CNI en el clúster, es posible que haya artefactos de CNI antiguos en los nodos híbridos. De forma predeterminada, al desinstalar Cilium y Calico con herramientas como Helm, los recursos del disco no se eliminan de las máquinas físicas o virtuales. Además, es posible que las definiciones de recursos personalizados (CRD) de estas CNI aún estén presentes en el clúster a partir de una instalación anterior. Para obtener más información, consulte las secciones Cómo eliminar Cilio y Cómo eliminar Calico en the section called “Cómo configurar una CNI”.

Solución de problemas de Cilium

Si tiene problemas al ejecutar Cilium en nodos híbridos, consulte los pasos de solución de problemas en la documentación de Cilium. Las secciones siguientes tratan problemas que pueden ser particulares de la implementación de Cilium en nodos híbridos.

Cilium no se inicia

Si los agentes de Cilium que se ejecutan en cada nodo híbrido no se inician, compruebe si hay errores en los registros de los pods de agentes de Cilium. El agente de Cilium requiere conectividad con el punto de conexión de la API de Kubernetes de EKS para poder iniciarse. Se producirá un error al iniciar el agente de Cilium si la conectividad no está configurada correctamente. En este caso, aparecerán mensajes de registro similares a los siguientes en los registros del pod del agente de Cilium.

```
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"
```

El agente de Cilium se ejecuta en la red host. El clúster de EKS debe estar configurado con `RemoteNodeNetwork` para la conectividad con Cilium. Confirme que cuenta con un grupo de seguridad adicional para el clúster de EKS con una regla de entrada para la `RemoteNodeNetwork`, que dispone de rutas en la VPC para la `RemoteNodeNetwork` y que la red en las instalaciones está configurada correctamente para permitir la conectividad con el plano de control de EKS.

Si el operador de Cilium está en ejecución y algunos de los agentes de Cilium se ejecutan, pero no la totalidad, confirme que tiene direcciones IP de pod disponibles para asignarlas a todos los nodos del clúster. El tamaño de los CIDR de pods asignables se configura al usar la administración de IP del grupo del clúster con `clusterPoolIPv4PodCIDRList` en la configuración de Cilium. El tamaño del CIDR por nodo se configura con el ajuste `clusterPoolIPv4MaskSize` de la configuración de Cilium. Consulte Ampliación del grupo de clústeres en la documentación de Cilium para obtener más información.

El BGP de Cilium no funciona

Si utiliza el plano de control de BGP de Cilium para anunciar las direcciones de los pods o servicios en la red en las instalaciones, puede utilizar los siguientes comandos de la CLI de Cilium para comprobar si BGP anuncia las rutas a los recursos. Para conocer los pasos que se deben seguir para instalar la CLI de Cilium, consulte Instalación de la CLI de Cilium en la documentación de Cilium.

Si BGP funciona correctamente, debería ver los nodos híbridos con el estado de sesión `established` en el resultado. Es posible que necesite trabajar con el equipo de redes para identificar los valores correctos para el AS local, el AS del vecino y la dirección del vecino del entorno.

```
cilium bgp peers
```

```
cilium bgp routes
```

Si utiliza el BGP de Cilium para anunciar las IP de los servicios con el tipo `LoadBalancer`, debe tener la misma etiqueta tanto en el `CiliumLoadBalancerIPPool` como en el servicio, que se debe utilizar en el selector del `CiliumBGPAdvertisement`. A continuación se muestra un ejemplo. Tenga en cuenta que, si utiliza el BGP de Cilium para anunciar las IP de los servicios del tipo LoadBalancer, es posible que las rutas de BGP se interrumpan durante el reinicio del agente de Cilium. Para obtener más información, consulte los Escenarios de error en la documentación de Cilium.

Servicio de

```
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 ] }
```

Solución de problemas de Calico

Si tiene problemas al ejecutar Cilium en nodos híbridos, consulte los pasos de solución de problemas en la documentación de Calico. En las siguientes secciones se describen los problemas que pueden ser exclusivos de la implementación de Calico en nodos híbridos.

En la siguiente tabla se resumen los componentes de Calico y se indica si se ejecutan en la red de nodos o de pods de forma predeterminada. Si configuró Calico para usar NAT para el tráfico de pods saliente, la red en las instalaciones debe estar configurada para dirigir el tráfico al CIDR del nodo en las instalaciones. Además, las tablas de enrutamiento de la VPC deben estar configuradas con una ruta para el CIDR del nodo en las instalaciones con la puerta de enlace de tránsito (TGW) o la puerta de enlace privada virtual (VGW) como destino. Si no va a configurar Calico para usar NAT para el tráfico de pods saliente, la red en las instalaciones debe estar configurada para dirigir el tráfico al CIDR del pod en las instalaciones. Además, las tablas de enrutamiento de la VPC deben estar configuradas con una ruta para el CIDR del pod en las instalaciones con la puerta de enlace de tránsito (TGW) o la puerta de enlace privada virtual (VGW) como destino.

Componente	Network
Servidor de la API de Calico	Nodo
Controladores de Calicon para Kubernetes	Pod
Agende de nodos de Calico	Nodo
Calico `typha`	Nodo
Controlador de nodos de CSI de Calico	Pod
Operador de Calico	Nodo

Los recursos de Calico están programados o se ejecutan en nodos acordonados

Los recursos de Calico que no se ejecutan como un DaemonSet tienen tolerancias flexibles de forma predeterminada gracias a las cuales es posible programarlos en nodos acordonados que no están preparados para programar o ejecutar pods. Para ajustar las tolerancias para los recursos de

Calico que no son de DaemonSet, puede cambiar la instalación del operador de modo que incluya lo siguiente.

```
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
```

Solución de problemas de credenciales

Tanto para las activaciones híbridas de AWS SSM como para AWS IAM Roles Anywhere, puede validar que las credenciales del rol de IAM de los Nodos híbridos estén configuradas correctamente en los nodos híbridos. Para ello, ejecute el siguiente comando desde los nodos híbridos. Confirme que el nombre del nodo y el nombre del rol de IAM de los Nodos híbridos son los esperados.

```
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>"
}
```

AWS Solución problemas de Systems Manager (SSM

Si utiliza activaciones híbridas de AWS SSM para las credenciales de nodos híbridos, tenga en cuenta los siguientes directorios y artefactos de SSM que `nodeadm` instala en los nodos híbridos. Para obtener más información sobre el agente de SSM, consulte Uso del agente de SSM en la Guía del usuario de AWS Systems Manager.

Descripción	Ubicación
Agente de SSM	Ubuntu: `/snap/amazon-ssm-agent/current/amazon-ssm-agent` RHEL y AL2023: `/usr/bin/amazon-ssm-agent`
Registros de SSM Agent	`/var/log/amazon/ssm`
AWSCredenciales de	`/root/.aws/credentials`
CLI de configuración de SSM	`/opt/ssm/ssm-setup-cli`

Reinicio del agente de SSM

Para resolver algunos problemas, se puede reiniciar el agente de SSM. Puede utilizar los comandos que aparecen a continuación para reiniciarlo.

AL2023 y otros sistemas operativos

```
systemctl restart amazon-ssm-agent
```

Ubuntu

```
systemctl restart snap.amazon-ssm-agent.amazon-ssm-agent
```

Compruebe la conectividad con los puntos de conexión de SSM

Confirme que se puede conectar a los puntos de conexión de SSM desde los nodos híbridos. Para obtener una lista de los puntos de conexión de SSM, consulte Puntos de conexión y cuotas de AWS Systems Manager. Sustituya `us-west-2` en el comando que aparece a continuación por la región de AWS de activación híbrida de AWS SSM.

```
ping ssm.us-west-2.amazonaws.com
```

Visualización del estado de la conexión de las instancias de SSM registradas

Puede comprobar el estado de conexión de las instancias que están registradas en las activaciones híbridas de SSM con el siguiente comando de la AWS CLI. Sustituya el ID de máquina por el ID de máquina de la instancia.

```
aws ssm get-connection-status --target mi-012345678abcdefgh
```

Falta de coincidencia de la suma de comprobación de la CLI de configuración de SSM

Al ejecutar `nodeadm install`, si detecta un problema de falta de coincidencia de la suma de comprobación de `ssm-setup-cli`, deberá confirmar que no haya instalaciones de SSM más antiguas en el host. Si hay instalaciones de SSM antiguas en el host, elimínelas y vuelva a ejecutar `nodeadm install` para resolver el problema.

```
Failed to perform agent-installation/on-prem registration: error while verifying
 installed ssm-setup-cli checksum: checksum mismatch with latest ssm-setup-cli.
```

De SSM **`InvalidActivation`**

Si aparece un error al registrar la instancia en AWS SSM, confirme que la `region`, `activationCode` y `activationId` en el `nodeConfig.yaml` son correctos. La región de AWS del clúster de EKS debe coincidir con la región de activación híbrida de SSM. Si estos valores están mal configurados, es posible que aparezca un error similar al siguiente.

```
ERROR Registration failed due to error registering the instance with AWS SSM.
 InvalidActivation
```

`ExpiredTokenException` de SSM: el token de seguridad incluido en la solicitud ha vencido

Si el agente de SSM no puede actualizar las credenciales, es posible que aparezca una `ExpiredTokenException`. En este escenario, si se puede conectar a los puntos de conexión de SSM desde los nodos híbridos, es posible que tenga que reiniciar el agente de SSM para forzar la actualización de credenciales.

```
"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"
```

Error de SSM al ejecutar el comando de registro de la máquina

Si aparece un error al registrar la máquina en SSM, es posible que tenga que volver a ejecutar `nodeadm install` para asegurarse de que todas las dependencias de SSM están instaladas correctamente.

```
"error":"running register machine command: , error: fork/exec /opt/aws/ssm-setup-cli:
 no such file or directory"
```

De SSM **`ActivationExpired`**

Al ejecutar `nodeadm init`, si aparece un error al registrar la instancia en SSM debido a que la activación ha caducado, tendrá que crear una nueva activación híbrida de SSM, actualizar `nodeConfig.yaml` con el `activationCode` y el `activationId` de la nueva activación híbrida de SSM y volver a ejecutar `nodeadm init`.

```
"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 no pudo actualizar las credenciales almacenadas en caché

Si se presenta un error al actualizar las credenciales almacenadas en caché, es posible que el archivo `/root/.aws/credentials` se haya eliminado del host. En primer lugar, compruebe la activación híbrida de SSM y asegúrese de que esté activa y de que los nodos híbridos estén configurados correctamente para utilizar la activación. Compruebe los registros del agente de SSM en `/var/log/amazon/ssm` y vuelva a ejecutar el comando `nodeadm init` una vez que haya resuelto el problema del SSM.

```
"Command failed","error":"operation error SSM: DescribeInstanceInformation, get
 identity: get credentials: failed to refresh cached credentials"
```

Limpie SSM

Para eliminar el agente de SSM del host, puede ejecutar los siguientes comandos.

```
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
```

AWS Solución de problemas de IAM Roles Anywhere

Si utiliza AWS IAM Roles Anywhere para las credenciales de nodos híbridos, tenga en cuenta los siguientes directorios y artefactos que `nodeadm` instala en los nodos híbridos. Para obtener más información sobre la solución de problemas de IAM Roles Anywhere, consulte Solución de problemas de identidad y acceso de AWS IAM Roles Anywhere en la Guía del usuario de AWS IAM Roles Anywhere.

Descripción	Ubicación
CLI de IAM Roles Anywhere	`/usr/local/bin/aws_signing_helper`
Ubicación y nombre predeterminados del certificado	`/etc/iam/pki/server.pem`
Ubicación y nombre predeterminados de la clave	`/etc/iam/pki/server.key`

IAM Roles Anywhere no pudo volver a refrescar las credenciales almacenadas en caché

Si detecta un error al refrescar las credenciales almacenadas en caché, revise el contenido de `/etc/aws/hybrid/config` y confirme que IAM Roles Anywhere se ha configurado correctamente en la configuración de `nodeadm`. Confirme que `/etc/iam/pki` existe. Cada nodo debe tener un certificado y una clave únicos. De forma predeterminada, cuando se utiliza IAM Roles Anywhere como proveedor de credenciales, `nodeadm` utiliza `/etc/iam/pki/server.pem` para la ubicación y el nombre del certificado, y `/etc/iam/pki/server.key` para la clave privada. Es posible que tenga que crear los directorios antes de colocar los certificados y las claves en los directorios con `sudo mkdir -p /etc/iam/pki`. Puede comprobar el contenido del certificado con el siguiente comando.

```
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 no está autorizado para realizar **`sts:AssumeRole`**

Si se presenta un problema de acceso denegado en los registros de `kubelet` para la operación `sts:AssumeRole` al utilizar IAM Roles Anywhere, compruebe la política de confianza del rol de IAM de Nodos híbridos para confirmar que la entidad principal del servicio de IAM Roles Anywhere puede asumir el rol de IAM de Nodos híbridos. Además, confirme que el ARN del anclaje de veracidad esté configurado correctamente en la política de confianza del rol de IAM de Nodos híbridos y que el rol de IAM de Nodos híbridos se haya agregado al perfil de IAM Roles Anywhere.

```
could not get token: AccessDenied: User: ... is not authorized to perform:
 sts:AssumeRole on resource: ...
```

IAM Roles Anywhere no está autorizado para definir **`roleSessionName`**

Si se presenta un problema de acceso denegado en los registros de `kubelet` al configurar el `roleSessionName`, confirme que `acceptRoleSessionName` se ha establecido en verdadero para el perfil de IAM Roles Anywhere.

```
AccessDeniedException: Not authorized to set roleSessionName
```

Solución de problemas del sistema operativo

RHEL

Fallos en el registro del administrador de derechos o suscripciones

Si ejecuta `nodeadm install` y se produce un error al instalar las dependencias de los nodos híbridos debido a problemas de registro de derechos, asegúrese de haber configurado correctamente el nombre de usuario y la contraseña de Red Hat en el host.

```
This system is not registered with an entitlement server
```

Ubuntu

No se encontró GLIBC

Si utiliza Ubuntu para el sistema operativo e IAM Roles Anywhere para el proveedor de credenciales con nodos híbridos y se presenta el problema de que no se encontró GLIBC, puede instalar esa dependencia manualmente para resolver el problema.

```
GLIBC_2.32 not found (required by /usr/local/bin/aws_signing_helper)
```

Ejecute los siguientes comandos para instalar la dependencia.

```
ldd --version
sudo apt update && apt install libc6
sudo apt install glibc-source
```

Bottlerocket

Si tiene activado el contenedor de administración de Bottlerocket, puede acceder a este mediante SSH para realizar tareas avanzadas de depuración y resolución de problemas con privilegios elevados. Las siguientes secciones contienen comandos que se deben ejecutar en el contexto del host de Bottlerocket. Una vez que esté en el contenedor de administración, puede ejecutar `sheltie` para obtener un intérprete de comandos raíz completo en el host de Bottlerocket.

```
sheltie
```

Para ejecutar los comandos de las siguientes secciones desde el intérprete de comandos del contenedor de administración, también puede agregar el prefijo `sudo chroot /.bottlerocket/rootfs` a cada comando.

```
sudo chroot /.bottlerocket/rootfs <command>
```

Cómo utilizar logdog en la recopilación de registros

Bottlerocket proporciona la utilidad `logdog` para recopilar de manera eficiente los registros y la información del sistema con fines de resolución de problemas.

```
logdog
```

La utilidad `logdog` recopila registros de varias ubicaciones en un host de Bottlerocket y los combina en un archivo tarball. De forma predeterminada, el archivo tarball se creará en `/var/log/support/bottlerocket-logs.tar.gz`, y se podrá acceder a este desde los contenedores del host en `/.bottlerocket/support/bottlerocket-logs.tar.gz`.

Cómo acceder a los registros del sistema con journalctl

Puede utilizar los siguientes comandos para comprobar el estado de los distintos servicios del sistema, como `kubelet` o `containerd`, entre otros, así como para ver sus registros. La marca `-f` seguirá los registros en tiempo real.

Para comprobar el estado del servicio `kubelet` y recuperar los registros de `kubelet`, puede ejecutar:

```
systemctl status kubelet
journalctl -u kubelet -f
```

Para comprobar el estado del servicio `containerd` y recuperar los registros de la instancia orquestada de `containerd`, puede ejecutar:

```
systemctl status containerd
journalctl -u containerd -f
```

Para comprobar el estado del servicio `host-containerd` y recuperar los registros de la instancia host de `containerd`, puede ejecutar:

```
systemctl status host-containerd
journalctl -u host-containerd -f
```

Para recuperar los registros de los contenedores de arranque y los contenedores host, puede ejecutar:

```
journalctl _COMM=host-ctr -f
```

Uso del almacenamiento de datos de aplicaciones para un clúster

Puede utilizar una gama de servicios de almacenamiento de AWS con Amazon EKS para satisfacer las necesidades de almacenamiento de las aplicaciones. Gracias a una amplia gama de controladores de interfaz de almacenamiento de contenedores (CSI) compatibles con AWS, puede utilizar fácilmente Amazon EBS, Amazon S3, Amazon EFS, Amazon FSX y Amazon File Cache para satisfacer las necesidades de almacenamiento de las aplicaciones que se ejecutan en Amazon EKS. Para administrar las copias de seguridad del clúster de Amazon EKS, consulte Soporte de AWS Backup para Amazon EKS.

En este capítulo se tratan las opciones de almacenamiento para clústeres de Amazon EKS.

Temas

- Uso del almacenamiento de volúmenes de Kubernetes con Amazon EBS
- Uso del almacenamiento del sistema de archivos elástico con Amazon EFS
- Uso del almacenamiento de aplicaciones de alto rendimiento con Amazon FSx para Lustre
- Uso del almacenamiento de aplicaciones de alto rendimiento con FSx para NetApp ONTAP
- Uso del almacenamiento de datos con Amazon FSx para OpenZFS
- Minimización de la latencia con Amazon File Cache
- Acceso a los objetos de Amazon S3 mediante Mountpoint para Amazon S3 con el controlador CSI
- Habilitación de la funcionalidad de instantáneas para volúmenes de CSI

Uso del almacenamiento de volúmenes de Kubernetes con Amazon EBS

> **Note**
>
> Nuevo: el modo automático de Amazon EKS automatiza las tareas rutinarias para el almacenamiento en bloque. Información sobre cómo the section called “Implementación de una carga de trabajo con estado”.

El controlador de la interfaz de almacenamiento de contenedores (CSI) de Amazon Elastic Block Store (Amazon EBS) administra el ciclo de vida de los volúmenes de Amazon EBS como almacenamiento para los volúmenes de Kubernetes que se creen. El controlador CSI de Amazon EBS crea volúmenes de Amazon EBS para estos tipos de volúmenes de Kubernetes: volúmenes efímeros genéricos y volúmenes persistentes.

Consideraciones

- No necesita instalar el controlador de CSI de Amazon EBS en los clústeres de modo automático de EKS.
- No puede montar volúmenes de Amazon EBS en los pods de Fargate.
- Puede ejecutar el controlador CSI de Amazon EBS en nodos de Fargate, pero el nodo CSI de Amazon EBS `DaemonSet` solo se puede ejecutar en instancias de Amazon EC2.
- Los volúmenes de Amazon EBS y el controlador CSI de Amazon EBS no son compatibles con los nodos híbridos de Amazon EKS.
- Se ofrecerá soporte para la versión más reciente del complemento y para una versión anterior. Los errores o vulnerabilidades encontrados en la versión más reciente se trasladarán a la versión anterior en una nueva versión menor.
- El modo automático de EKS requiere que las clases de almacenamiento usen `ebs.csi.eks.amazonaws.com` como aprovisionador. El controlador de CSI estándar de Amazon EBS (`ebs.csi.aws.com`) administra sus propios volúmenes por separado. Para utilizar los volúmenes existentes con el modo automático de EKS, mígrelos mediante instantáneas de volumen a una clase de almacenamiento que utilice el aprovisionador de modo automático.

> **Important**
>
> Para utilizar la funcionalidad de instantáneas del controlador de CSI de Amazon EBS, primero deberá instalar el controlador de instantáneas de CSI. Para obtener más información, consulte the section called “Controlador de instantáneas CSI”.

Requisitos previos

- Un clúster existente. Para ver la versión de plataforma requerida, ejecute el siguiente comando.

```
aws eks describe-addon-versions --addon-name aws-ebs-csi-driver
```

- El controlador CSI de EBS necesita permisos de AWS IAM.
  - AWS sugiere usar Pod Identities de EKS. Para obtener más información, consulte the section called “Información general de configuración de las Pod Identities de EKS”.
  - Para obtener información sobre roles de IAM para cuentas de servicio, consulte the section called “Proveedor de OIDC de IAM”.

Paso 1: creación de un rol de IAM

El complemento CSI de Amazon EBS requiere permisos de IAM para realizar llamadas a las API de AWS en su nombre. Si no lleva a cabo estos pasos, aparecerá `failed to provision volume with StorageClass` con el error `could not create volume in EC2: UnauthorizedOperation` cuando intente instalar el complemento y ejecutar `kubectl describe pvc`. Para obtener más información, consulte Configurar el permiso del controlador en GitHub.

> **Note**
>
> Los pods tendrán acceso a los permisos asignados al rol de IAM, a menos que bloquee el acceso al IMDS. Para obtener más información, consulte the section called “Prácticas recomendadas”.

El siguiente procedimiento muestra cómo crear un rol de IAM y asociarle la política administrada de AWS. Para implementar este procedimiento, puede utilizar una de las siguientes herramientas:

- the section called “`eksctl`”
- the section called “Consola de administración de AWS”
- the section called “AWS CLI”

> **Note**
>
> Puede crear una política autoadministrada con permisos limitados. Consulte `AmazonEBSCSIDriverPolicy` y cree una política de IAM personalizada con permisos reducidos.

> **Note**
>
> Los pasos específicos de este procedimiento están diseñados para usar el controlador como complemento de Amazon EKS. Se necesitan diferentes pasos para usar el controlador como complemento autoadministrado. Para obtener más información, consulte Set up driver permissions en GitHub.

`eksctl`

1. Cree un rol de IAM y asóciele una política. AWS mantiene una política administrada de AWS, pero también puede crear su propia política personalizada. Puede crear un rol de IAM y asociar la política administrada de AWS con el siguiente comando. Reemplace *`my-cluster`* por el nombre de su clúster. El comando implementa una pila de AWS CloudFormation que crea un rol de IAM y le adjunta la política de IAM.

   ```
   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. Puede omitir este paso si no utiliza una clave KMS personalizada. Si utiliza una para el cifrado en los volúmenes de Amazon EBS, personalice el rol de IAM según sea necesario. Por ejemplo, haga lo siguiente:

   a. Copie y pegue el siguiente código en un nuevo archivo `kms-key-for-encryption-on-ebs.json`. Reemplace *`custom-key-arn`* por el ARN de clave de KMS personalizado.

   ```
   {
         "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. Creación de la política. Puede cambiar *KMS_Key_For_Encryption_On_EBS_Policy* por un nombre diferente. Sin embargo, si lo hace, asegúrese de hacerlo también en pasos posteriores.

```
aws iam create-policy \
      --policy-name KMS_Key_For_Encryption_On_EBS_Policy \
      --policy-document file://kms-key-for-encryption-on-ebs.json
```

c. Adjunte una política de IAM y adjunte la política de IAM al rol con el siguiente comando. Reemplace *111122223333* por el ID de su cuenta.

```
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
```

Consola de administración de AWS

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación izquierdo, elija Roles.
3. En la página Roles, elija Crear rol.
4. En la página Seleccionar entidad de confianza, haga lo siguiente:
   a. En la sección Tipo de entidad de confianza, elija Identidad web.
   b. Para Identity provider (Proveedor de identidades), elija OpenID Connect provider URL (URL del proveedor de OpenID Connect) para el clúster, como se muestra en Overview (Resumen) en Amazon EKS.
   c. En Audiencia, elija `sts.amazonaws.com`.
   d. Elija Siguiente.
5. En la página Agregar permisos, haga lo siguiente:
   a. En el cuadro Filtrar políticas, escriba `AmazonEBSCSIDriverPolicy`.
   b. Marque la casilla situada a la izquierda del nombre de la `AmazonEBSCSIDriverPolicy` que obtuvo en la búsqueda.
   c. Elija Siguiente.
6. En la página Nombrar, revisar y crear, haga lo siguiente:
   a. En Nombre del rol, ingrese un nombre único para su rol, por ejemplo, *`AmazonEKS_EBS_CSI_DriverRole`*.
   b. En Agregar etiquetas (Opcional), de manera opcional, agregue metadatos al rol asociando etiquetas como pares de clave-valor. Para obtener más información sobre el uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM en la Guía de usuario de IAM.
   c. Seleccione Crear rol.
7. Una vez creado el rol, seleccione el rol en la consola para abrirlo y editarlo.
8. Elija la pestaña Relaciones de confianza y, a continuación, Editar política de confianza.
9. Busque la línea que se parezca a la siguiente:

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud":
 "sts.amazonaws.com"
```

   Agregue una coma al final de la línea anterior y, luego, agregue la siguiente línea después de la anterior. Reemplace *`region-code`* por la región de AWS en la que se encuentra el clúster.

Reemplace *EXAMPLED539D4633E53DE1B71EXAMPLE* con el ID de proveedor de OIDC de su clúster.

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub":
 "system:serviceaccount:kube-system:ebs-csi-controller-sa"
```

10. Elija Actualizar política para terminar.
11. Si utiliza una clave de KMS personalizada para el cifrado en los volúmenes de Amazon EBS, personalice el rol de IAM según sea necesario. Por ejemplo, haga lo siguiente:
    a. En el panel de navegación izquierdo, elija Políticas.
    b. En la página Políticas, seleccione Crear una política.
    c. En la página Crear política, elija la pestaña JSON.
    d. Copie y pegue el siguiente código en el editor y reemplace *custom-key-arn* por el ARN de la clave de KMS personalizado.

```
{
      "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. Elija Siguiente: etiquetas.

f. En la página Agregar etiquetas (opcional), elija Siguiente: revisar.

g. En Nombre, introduzca un nombre exclusivo para la política (por ejemplo, *KMS_Key_For_Encryption_On_EBS_Policy*).

h. Seleccione Crear política.

i. En el panel de navegación izquierdo, elija Roles.

j. Elija ***AmazonEKS_EBS_CSI_DriverRole*** en la consola para abrirlo y editarlo.

k. En la lista desplegable Agregar permisos, seleccione Asociar políticas.

l. En el cuadro Políticas de filtrado, escriba *KMS_Key_For_Encryption_On_EBS_Policy*.

m. Seleccione la casilla situada a la izquierda del nombre de la *KMS_Key_For_Encryption_On_EBS_Policy* que obtuvo en la búsqueda.

n. Seleccione Asociar políticas.

AWS CLI

1. Visualización de la URL del proveedor de OIDC de su clúster. Sustituya *my-cluster* por el nombre del clúster. Si la salida del comando es None, revise los Requisitos previos.

```
aws eks describe-cluster --name my-cluster --query "cluster.identity.oidc.issuer" --
output text
```

Un ejemplo de salida sería el siguiente.

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

2. Creación del rol de IAM y otorgamiento de la acción `AssumeRoleWithWebIdentity`.

a. Copie el siguiente contenido en un archivo con el nombre `aws-ebs-csi-driver-trust-policy.json`. Reemplace *111122223333* por el ID de su cuenta. Reemplace

EXAMPLED539D4633E53DE1B71EXAMPLE y *region-code* por los valores que se devolvieron en el paso anterior.

```
{
      "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. Creación del rol. Puede cambiar *AmazonEKS_EBS_CSI_DriverRole* a un nombre diferente. Si lo cambia, asegúrese de hacerlo también en pasos posteriores.

```
aws iam create-role \
      --role-name AmazonEKS_EBS_CSI_DriverRole \
      --assume-role-policy-document file://"aws-ebs-csi-driver-trust-policy.json"
```

3. Asocie una política. AWS mantiene una política administrada de AWS, pero también puede crear su propia política personalizada. Asocie la política administrada de AWS al rol con el siguiente comando.

```
aws iam attach-role-policy \
      --policy-arn arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy \
      --role-name AmazonEKS_EBS_CSI_DriverRole
```

4. Si utiliza una clave de KMS personalizada para el cifrado en los volúmenes de Amazon EBS, personalice el rol de IAM según sea necesario. Por ejemplo, haga lo siguiente:

   a. Copie y pegue el siguiente código en un nuevo archivo `kms-key-for-encryption-on-ebs.json`. Reemplace *`custom-key-arn`* por el ARN de clave de KMS personalizado.

```
{
      "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. Creación de la política. Puede cambiar *`KMS_Key_For_Encryption_On_EBS_Policy`* por un nombre diferente. Sin embargo, si lo hace, asegúrese de hacerlo también en pasos posteriores.

```
aws iam create-policy \
```

```
      --policy-name KMS_Key_For_Encryption_On_EBS_Policy \
      --policy-document file://kms-key-for-encryption-on-ebs.json
```

c. Adjunte una política de IAM y adjunte la política de IAM al rol con el siguiente comando. Reemplace *111122223333* por el ID de su cuenta.

```
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
```

Ahora que ha creado el rol de IAM del controlador de CSI de Amazon EBS, puede continuar con la siguiente sección. Cuando implementa el complemento con este rol de IAM, se crea y se configura para utilizar una cuenta de servicio que se llama `ebs-csi-controller-sa`. La cuenta de servicio está vinculada a un `clusterrole` de Kubernetes al que se le asignan los permisos de Kubernetes necesarios.

Paso 2: obtención del controlador de CSI de Amazon EBS

Le recomendamos que instale el controlador de CSI de Amazon EBS a través del complemento de Amazon EKS para mejorar la seguridad y reducir la cantidad de trabajo. Para agregar un complemento de Amazon EKS al clúster, consulte the section called “Cómo crear un complemento”. Para obtener más información sobre los complementos, consulte the section called “Complementos de Amazon EKS”.

> **Important**
>
> Antes de añadir el controlador de Amazon EBS como complemento de Amazon EKS, confirme que no tiene una versión autoadministrada del controlador instalada en su clúster. Si es así, consulte Uninstalling a self-managed Amazon EBS CSI driver en GitHub.

> **Note**
>
> De forma predeterminada, el rol de RBAC que utiliza la CSI de EBS tiene permisos para mutar nodos a fin de admitir la característica de eliminación de taints. Debido a las limitaciones de RBAC de Kubernetes, también le permite mutar cualquier otro nodo del clúster. El gráfico de Helm tiene un parámetro

(`node.serviceAccount.disableMutation`) que desactiva los permisos de RBAC del nodo mutante para la cuenta de servicio ebs-csi-node. Cuando se activan, las características del controlador, como la eliminación de taints, no funcionarán.

Como alternativa, si desea una instalación autoadministrada del controlador CSI de Amazon EBS, consulte Installation en GitHub.

Paso 3: implementación de una aplicación de muestra

Puede implementar una variedad de aplicaciones de muestra y modificarlas según sea necesario. Para obtener más información, consulte Kubernetes Examples en GitHub.

Uso del almacenamiento del sistema de archivos elástico con Amazon EFS

Amazon Elastic File System (Amazon EFS) proporciona un almacenamiento de archivos totalmente elástico y sin servidor para que pueda compartir datos de archivos sin aprovisionar ni administrar la capacidad de almacenamiento ni el rendimiento. El controlador de Container Storage Interface (CSI) de Amazon EFS proporciona una interfaz CSI que permite a los clústeres de Kubernetes que se ejecutan en AWS administrar el ciclo de vida de los sistemas de archivos de Amazon EFS. En este tema se muestra cómo implementar el controlador CSI de Amazon EFS en su clúster de Amazon EKS.

Consideraciones

- El controlador CSI de Amazon EFS no es compatible con imágenes de contenedores basadas en Windows.
- No se puede utilizar el aprovisionamiento dinámico de volúmenes persistentes con los nodos de Fargate, pero sí se puede utilizar el aprovisonamiento estático.
- El aprovisionamiento dinámico requiere la versión 1.2 del controlador o posterior. Puede utilizar el aprovisionamiento estático para volúmenes persistentes con la versión `1.1` del controlador en cualquier versión de clúster de Amazon EKS compatible (consulte las versiones compatibles de Amazon EKS).
- La versión 1.3.2 y posteriores de este controlador son compatibles con la arquitectura Arm64, incluidas las instancias basadas en Graviton de Amazon EC2.

- La versión 1.4.2 o posterior de este controlador admite el uso de FIPS para montar sistemas de archivos.
- Tome nota de las cuotas de recursos de Amazon EFS. Por ejemplo, hay una cuota de 1000 puntos de acceso que se pueden crear para cada sistema de archivos de Amazon EFS. Para obtener más información, consulte Cuotas de recursos de Amazon EFS que no puede cambiar.
- A partir de la versión 2.0.0, este controlador pasó de utilizar `stunnel` a `efs-proxy` para las conexiones TLS. Cuando se utiliza `efs-proxy`, se abrirá una cantidad de subprocesos igual a uno más la cantidad de núcleos del nodo en el que se ejecuta.
- El controlador CSI de Amazon EFS no es compatible con los nodos híbridos de Amazon EKS.

Requisitos previos

- El controlador CSI de Amazon EFS necesita los permisos de AWS Identity and Access Management (IAM).
  - AWS sugiere usar Pod Identities de EKS. Para obtener más información, consulte the section called “Información general de configuración de las Pod Identities de EKS”.
  - Para obtener más información sobre los roles de IAM para las cuentas de servicio y la configuración de un proveedor de OpenID Connect (OIDC) de IAM para el clúster, consulte the section called “Proveedor de OIDC de IAM”.
- La versión `2.12.3` o posterior, o bien, la versión `1.27.160` o posterior de la AWS interfaz de la línea de comandos (AWS CLI) instalada y configurada en su dispositivo o AWS CloudShell. Para comprobar su versión actual, utilice `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente. Para actualizarla, consulte Instalación de la CLI de AWS en su directorio principal en la Guía del usuario de AWS CloudShell.
- La herramienta de línea de comandos de `kubectl` está instalada en su dispositivo o AWS CloudShell. La versión puede ser la misma o hasta una versión secundaria anterior o posterior a la versión de Kubernetes de su clúster. Por ejemplo, si la versión del clúster es `1.29`, puede usar la versión `1.28`, `1.29` o `1.30` de `kubectl` con él. Para instalar o actualizar `kubectl`, consulte the section called “Configure kubectl y eksctl”.

> **Note**
>
> Un pod que se ejecuta en Fargate monta automáticamente un sistema de archivos de Amazon EFS, sin necesidad de seguir pasos manuales para la instalación de controladores.

Paso 1: creación de un rol de IAM

El controlador CSI de Amazon EFS requiere permisos de IAM para interactuar con el sistema de archivos. Cree un rol de IAM y adjúntelo a la política administrada de AWS requerida. Para implementar este procedimiento, puede utilizar una de las siguientes herramientas:

- the section called “`eksctl`”
- the section called “Consola de administración de AWS”
- the section called “AWS CLI”

> **Note**
>
> Los pasos específicos de este procedimiento están diseñados para usar el controlador como complemento de Amazon EKS. Para obtener más información sobre las instalaciones autoadministradas, consulte Set up driver permission en GitHub.

`eksctl`

Si se utiliza Pod Identities

Ejecute los siguientes comandos para crear un rol de IAM y la asociación de Pod Identity con `eksctl`. Reemplace `my-cluster` por el nombre del clúster. También puede reemplazar `AmazonEKS_EFS_CSI_DriverRole` por un nombre diferente.

```
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
```

Si se utilizan roles de IAM para cuentas de servicio

Ejecute los siguientes comandos para crear el rol de IAM con `eksctl`. Reemplace `my-cluster` por el nombre del clúster. También puede reemplazar `AmazonEKS_EFS_CSI_DriverRole` por un nombre diferente.

```
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"
```

Consola de administración de AWS

Ejecute lo siguiente para crear un rol de IAM con la Consola de administración de AWS.

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación izquierdo, elija Roles.
3. En la página Roles, elija Crear rol.
4. En la página Seleccionar entidad de confianza, haga lo siguiente:
   a. Si se utiliza Pod Identities de EKS:
      i. En la sección Tipo de entidad de confianza, elija Servicio de AWS.
      ii. En el menú desplegable Servicio o caso de uso, seleccione EKS.
      iii. En la sección Caso de uso, seleccione EKS: Pod Identity.
      iv. Elija Siguiente.
   b. Si se utilizan roles de IAM para cuentas de servicio:

i. En la sección Tipo de entidad de confianza, elija Identidad web.

ii. Para Identity provider (Proveedor de identidades), elija OpenID Connect provider URL (URL del proveedor de OpenID Connect) para el clúster, como se muestra en Overview (Resumen) en Amazon EKS.

iii. En Audiencia, elija `sts.amazonaws.com`.

iv. Elija Siguiente.

5. En la página Agregar permisos, haga lo siguiente:

a. En el cuadro Filtrar políticas, escriba `AmazonEFSCSIDriverPolicy`.

b. Marque la casilla situada a la izquierda del nombre de la `AmazonEFSCSIDriverPolicy` que obtuvo en la búsqueda.

c. Elija Siguiente.

6. En la página Nombrar, revisar y crear, haga lo siguiente:

a. En Nombre del rol, ingrese un nombre único para su rol, por ejemplo, `AmazonEKS_EFS_CSI_DriverRole`.

b. En Agregar etiquetas (Opcional), de manera opcional, agregue metadatos al rol asociando etiquetas como pares de clave-valor. Para obtener más información sobre el uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM en la Guía de usuario de IAM.

c. Seleccione Crear rol.

7. Una vez creado el rol:

a. Si se utiliza Pod Identities de EKS:

i. Abra la consola de Amazon EKS.

ii. En el panel de navegación izquierdo, seleccione Clústeres y, a continuación, seleccione el nombre del clúster para el que desea configurar la asociación de Pod Identity de EKS.

iii. Elija la pestaña Acceso.

iv. En Asociaciones de Pod Identity, elija Crear.

v. Elija el menú desplegable Rol de IAM y seleccione el rol recién creado.

vi. Elija el campo del espacio de nombres de Kubernetes e introduzca `kube-system`.

vii. Elija el campo de la cuenta de servicio de Kubernetes e introduzca `efs-csi-controller-sa`.

viii. Seleccione Crear.

ix. Para obtener más información sobre la creación de asociaciones de Pod Identity, consulte the section called “Creación de una asociación de Pod Identity (consola de AWS)”.

b. Si se utilizan roles de IAM para cuentas de servicio:

i. Elija el rol para abrirlo y editarlo.

ii. Elija la pestaña Relaciones de confianza y, a continuación, Editar política de confianza.

iii. Busque la línea que se parezca a la siguiente:

```
"oidc.eks.region-code.amazonaws.com/id/<EXAMPLED539D4633E53DE1B71EXAMPLE>:aud":
 "sts.amazonaws.com"
```

Añada la siguiente línea por encima de la línea anterior. Reemplace `<region-code>` con la región de AWS en la que se encuentra el clúster. Reemplace `<EXAMPLED539D4633E53DE1B71EXAMPLE>` con el ID del proveedor OIDC del clúster.

```
"oidc.eks.<region-code>.amazonaws.com/id/
<EXAMPLED539D4633E53DE1B71EXAMPLE>:sub": "system:serviceaccount:kube-system:efs-
csi-*",
```

iv. Modifique el operador `Condition` de `"StringEquals"` a `"StringLike"`.

v. Elija Actualizar política para terminar.

AWS CLI

Ejecute los siguientes comandos para crear un rol de IAM con la CLI de AWS.

Si se utiliza Pod Identities

1. Cree el rol de IAM que otorga las acciones `AssumeRole` y `TagSession` al servicio `pods.eks.amazonaws.com`.

a. Copie el siguiente contenido en un archivo denominado `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. Cree el rol. Reemplace `my-cluster` por el nombre del clúster. También puede reemplazar `AmazonEKS_EFS_CSI_DriverRole` por un nombre diferente.

```
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. Asocie el comando requerido administrado por AWS al rol con el siguiente comando.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/service-role/AmazonEFSCSIDriverPolicy \
  --role-name $role_name
```

3. Ejecute el siguiente comando para crear la asociación de Pod Identity. Sustituya `arn:aws:iam::<111122223333>:role/my-role` por el rol creado en los pasos anteriores.

```
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. Para obtener más información sobre la creación de asociaciones de Pod Identity, consulte the section called “Creación de una asociación de Pod Identity (consola de AWS)”.

Si se utilizan roles de IAM para cuentas de servicio

1. Visualización de la URL del proveedor de OIDC de su clúster. Reemplace `my-cluster` por el nombre del clúster. También puede reemplazar `AmazonEKS_EFS_CSI_DriverRole` por un nombre diferente.

```
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
```

Un ejemplo de salida sería el siguiente.

```
https://oidc.eks.<region-code>.amazonaws.com/id/<EXAMPLED539D4633E53DE1B71EXAMPLE>
```

Si la salida del comando es `None`, revise los Requisitos previos.

2. Cree el rol de IAM que concede la acción `AssumeRoleWithWebIdentity`.
   a. Copie el siguiente contenido en un archivo denominado `aws-efs-csi-driver-trust-policy.json`. Reemplace <111122223333> por su ID de cuenta. Reemplace <EXAMPLED539D4633E53DE1B71EXAMPLE> y <region-code> por los valores que se devolvieron en el paso anterior.

```
{
  "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. Creación del rol.

```
aws iam create-role \
  --role-name $role_name \
  --assume-role-policy-document file://"aws-efs-csi-driver-trust-policy.json"
```

3. Asocie el comando requerido administrado por AWS al rol con el siguiente comando.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/service-role/AmazonEFSCSIDriverPolicy \
  --role-name $role_name
```

Paso 2: obtención del controlador de CSI de Amazon EFS

Le recomendamos que instale el controlador CSI de Amazon EFS a través del complemento Amazon EKS. Para agregar un complemento de Amazon EKS al clúster, consulte the section called “Cómo crear un complemento”. Para obtener más información sobre los complementos, consulte the section called “Complementos de Amazon EKS”. Si no puede utilizar el complemento de Amazon EKS, le recomendamos que envíe una pregunta sobre los motivos por los que no puede hacerlo al repositorio de GitHub de la hoja de ruta de contenedores.

> **Important**
>
> Antes de agregar el controlador de Amazon EBS como complemento de Amazon EKS, asegúrese de que no haya una versión autoadministrada del controlador instalada en el clúster. Si es así, consulte Desinstalar el controlador CSI de Amazon EFS en GitHub.

Como alternativa, si desea una instalación autoadministrada del controlador CSI de Amazon EFS, consulte Installation en GitHub.

Paso 3: creación de un sistema de archivos de Amazon EFS

Para crear un sistema de archivos de Amazon EFS, consulte Crear un sistema de archivos de Amazon EFS para Amazon EKS en GitHub.

Paso 4: implementación de una aplicación de muestra

Puede implementar una variedad de aplicaciones de muestra y modificarlas según sea necesario. Para obtener más información, consulte Ejemplos en GitHub.

Uso del almacenamiento de aplicaciones de alto rendimiento con Amazon FSx para Lustre

El controlador de Container Storage Interface (CSI) de Amazon FSx for Lustre proporciona una interfaz CSI que permite a los clústeres de Amazon EKS administrar el ciclo de vida de los sistemas de archivos de Amazon FSx for Lustre. Para obtener más información, consulte la Guía del usuario de Amazon FSx para Lustre.

Para obtener detalles sobre cómo implementar el controlador de CSI de Amazon FSx para Lustre en el clúster de Amazon EKS y verificar su funcionamiento, consulte the section called “Implementación del controlador”.

Implementación del controlador de FSx para Lustre

Este tema muestra cómo implementar el controlador CSI de FSx para Lustre en el clúster de Amazon EKS y verificar que funcione correctamente. Siempre recomendamos usar la versión más reciente del controlador. Para ver las versiones disponibles, consulte CSI Specification Compatibility Matrix (Matriz de compatibilidad de especificaciones de CSI) en GitHub.

> **Note**
>
> El controlador no es compatible en Fargate ni en los Nodos híbridos de Amazon EKS.

Para obtener descripciones detalladas de los parámetros disponibles y ejemplos completos que demuestran las características del controlador, consulte el proyecto FSx for Lustre Container Storage Interface (CSI) driver en GitHub.

Requisitos previos

- Un clúster existente.
- El complemento de EKS del controlador de CSI de Amazon FSx requiere el agente de EKS Pod Identity para la autenticación. Sin este componente, se producirá el error `Amazon EKS Pod Identity agent is not installed in the cluster` en el complemento, que impedirá las operaciones de volumen. Instale el agente de Pod Identity antes o después de implementar el complemento del controlador de CSI de FSx. Para obtener más información, consulte the section called “Configuración del agente”.

- La versión `2.12.3` o posterior, o bien, la versión `1.27.160` o posterior de la AWS interfaz de la línea de comandos (AWS CLI) instalada y configurada en su dispositivo o AWS CloudShell. Para comprobar su versión actual, utilice `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente. Para actualizarla, consulte Instalación de AWS CLI en su directorio de usuarios principal en la Guía del usuario de AWS CloudShell.
- La versión `0.215.0` o posterior de la herramienta de línea de comandos `eksctl` instalada en su dispositivo o AWS CloudShell. Para instalar o actualizar `eksctl`, consulte la sección de Instalación en la documentación de `eksctl`.
- La herramienta de línea de comandos de `kubectl` está instalada en su dispositivo o AWS CloudShell. La versión puede ser la misma o hasta una versión secundaria anterior o posterior a la versión de Kubernetes de su clúster. Por ejemplo, si la versión del clúster es `1.29`, puede usar la versión `1.28`, `1.29` o `1.30` de `kubectl` con él. Para instalar o actualizar `kubectl`, consulte the section called “Configure kubectl y eksctl”.

Paso 1: creación de un rol de IAM

El complemento de CSI de Amazon FSx requiere permisos de IAM para hacer llamadas a las API de AWS en su nombre.

> **Note**
>
> Los pods tendrán acceso a los permisos asignados al rol de IAM, a menos que bloquee el acceso al IMDS. Para obtener más información, consulte the section called “Prácticas recomendadas”.

El siguiente procedimiento muestra cómo crear un rol de IAM y asociarle la política administrada de AWS.

1. Cree un rol de IAM y adjunte la política administrada de AWS con el siguiente comando. Sustituya `my-cluster` por el nombre del clúster que desea utilizar. El comando implementa una pila de AWS CloudFormation que crea un rol de IAM y le adjunta la política de IAM.

```
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
```

Verá varias líneas de salida a medida que se crea la cuenta de servicio. Las últimas líneas de salida es similar a la siguiente línea de ejemplo.

```
[#]  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"
```

Apunte el nombre de la pila de AWS CloudFormation que se implementó. En la salida de ejemplo anterior, la pila se denomina `eksctl-my-cluster-addon-iamserviceaccount-kube-system-fsx-csi-controller-sa`.

Ahora que ha creado el rol de IAM del controlador de CSI de Amazon FSx, puede continuar con la siguiente sección. Cuando implementa el complemento con este rol de IAM, se crea y se configura para utilizar una cuenta de servicio que se llama `fsx-csi-controller-sa`. La cuenta de servicio está vinculada a un `clusterrole` de Kubernetes al que se le asignan los permisos de Kubernetes necesarios.

Paso 2: instalación del controlador de CSI de Amazon FSx

Le recomendamos que instale el controlador de CSI de Amazon FSx a través del complemento de Amazon EKS para mejorar la seguridad y reducir la cantidad de trabajo. Para agregar un

complemento de Amazon EKS al clúster, consulte the section called “Cómo crear un complemento”. Para obtener más información sobre los complementos, consulte the section called “Complementos de Amazon EKS”.

> **Important**
>
> Las instalaciones preexistentes del controlador de CSI de Amazon FSx en el clúster pueden provocar errores en la instalación de los complementos. Si intenta instalar la versión del complemento de Amazon EKS mientras existe un controlador de CSI de FSx que no es de EKS, se producirá un error en la instalación debido a conflictos de recursos. Utilice la marca `OVERWRITE` durante la instalación para resolver este problema.
>
> ```
> aws eks create-addon --addon-name aws-fsx-csi-driver --cluster-name my-cluster
>  --resolve-conflicts OVERWRITE
> ```

Como alternativa, si desea una instalación autoadministrada del controlador de CSI de Amazon FSx, consulte Installation en GitHub.

Paso 3: implementación de una clase de almacenamiento, una solicitud de volumen persistente y una aplicación de ejemplo

Este procedimiento utiliza el repositorio GitHub del controlador de Container Storage Interface (CSI) de FSx para Lustre para utilizar un volumen de FSx for Lustre aprovisionado dinámicamente.

1. Anote el grupo de seguridad del clúster. Puede verlo en la Consola de administración de AWS en la sección Redes o utilizando el siguiente comando de AWS CLI. Sustituya `my-cluster` por el nombre del clúster que desea utilizar.

   ```
   aws eks describe-cluster --name my-cluster --query
    cluster.resourcesVpcConfig.clusterSecurityGroupId
   ```

2. Cree un grupo de seguridad para su sistema de archivos Amazon FSx de acuerdo con los criterios que se muestran en Grupos de seguridad de Amazon VPC en la Guía del usuario de Amazon FSx para Lustre. Para la VPC, seleccione la VPC de su clúster tal como se muestra en la sección Networking (Redes). Para “los grupos de seguridad asociados a los clientes de Lustre”, utilice el grupo de seguridad de clúster. Puede dejar solo las reglas de salida para permitir All traffic (Todo el tráfico).

3. Descargue el manifiesto de con el siguiente comando.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/master/
examples/kubernetes/dynamic_provisioning/specs/storageclass.yaml
```

4. Edite la sección de parámetros del archivo `storageclass.yaml`. Reemplace todos los valores de ejemplo por sus propios valores.

```
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`** – el ID de subred en el que se debe crear el sistema de archivos de Amazon FSx para Lustre. Amazon FSx para Lustre no se admite en todas las zonas de disponibilidad. Abra la consola de Amazon FSx para Lustre en https://console.aws.amazon.com/fsx/ para confirmar que la subred que desea utilizar se encuentra en una zona de disponibilidad compatible. La subred puede incluir sus nodos o puede ser una subred o VPC diferente:
  - Puede comprobar si hay subredes de nodos en la Consola de administración de AWS seleccionando el grupo de nodos en la sección Compute (Informática).
  - Si la subred que especifica no es la misma en la que tiene los nodos, las VPC deben estar conectadas y debe asegurarse de que tiene abiertos los puertos necesarios en los grupos de seguridad.
- **`securityGroupIds`** – el ID del grupo de seguridad que ha creado para el sistema de archivos.
- **`deploymentType`** (opcional): el tipo de implementación del sistema de archivos. Los valores válidos son `SCRATCH_1`, `SCRATCH_2`, `PERSISTENT_1` y `PERSISTENT_2`. Para obtener más información sobre los tipos de implementación, consulte cómo crear su sistema de archivos de Amazon FSx para Lustre.
- otros parámetros (opcionales): para obtener información acerca del resto de parámetros, consulte Edit StorageClass (Editar StorageClass) en GitHub.

5. Cree el manifiesto de clase de almacenamiento.

```
kubectl apply -f storageclass.yaml
```

Un ejemplo de salida sería el siguiente.

```
storageclass.storage.k8s.io/fsx-sc created
```

6. Descargue el manifiesto de notificación de volumen persistente.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/master/
examples/kubernetes/dynamic_provisioning/specs/claim.yaml
```

7. (Opcional) Edite el archivo `claim.yaml`. Cambie `1200Gi` por uno de los valores de incremento que se indican a continuación, en función de los requisitos de almacenamiento y del `deploymentType` que seleccionó en los pasos anteriores.

```
storage: 1200Gi
```

- `SCRATCH_2` y `PERSISTENT`: `1.2 TiB`, `2.4 TiB` o incrementos de 2.4 TiB sobre 2.4 TiB.
- `SCRATCH_1` – `1.2 TiB`, `2.4 TiB`, `3.6 TiB`, o incrementos de 3.6 TiB sobre 3.6 TiB.

8. Cree la notificación de volumen persistente.

```
kubectl apply -f claim.yaml
```

Un ejemplo de salida sería el siguiente.

```
persistentvolumeclaim/fsx-claim created
```

9. Confirme que el sistema de archivos está aprovisionado.

```
kubectl describe pvc
```

Un ejemplo de salida sería el siguiente.

```
Name:          fsx-claim
Namespace:     default
StorageClass:  fsx-sc
Status:        Bound
```

```
[...]
```

> **Note**
>
> El `Status` puede aparecer como `Pending` durante 5-10 minutos, antes de cambiar a `Bound`. No continúe con el siguiente paso hasta que el `Status` sea `Bound`. Si el `Status` muestra `Pending` durante más de 10 minutos, utilice los mensajes de advertencia en los `Events` como referencia para abordar cualquier problema.

10. Implemente la aplicación de muestra.

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/master/examples/kubernetes/dynamic_provisioning/specs/pod.yaml
```

11. Verifique que la aplicación de muestra se está ejecutando.

```
kubectl get pods
```

Un ejemplo de salida sería el siguiente.

```
NAME      READY   STATUS              RESTARTS   AGE
fsx-app   1/1     Running             0          8s
```

12. Verifique que la aplicación haya montado correctamente el sistema de archivos.

```
kubectl exec -ti fsx-app -- df -h
```

Un ejemplo de salida sería el siguiente.

```
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. Verifique que la aplicación de muestra haya escrito los datos en el sistema de archivos de FSx para Lustre.

```
kubectl exec -it fsx-app -- ls /data
```

Un ejemplo de salida sería el siguiente.

```
out.txt
```

En este resultado de ejemplo se muestra que la aplicación de ejemplo escribió correctamente el archivo `out.txt` en el sistema de archivos.

> **Note**
>
> Antes de eliminar el clúster, asegúrese de eliminar el sistema de archivos de FSx para Lustre. Para obtener más información, consulte Limpiar recursos en la Guía de usuario de FSx para Lustre.

Ajuste del rendimiento para FSx para Lustre

Al utilizar FSx para Lustre con Amazon EKS, puede optimizar el rendimiento mediante la aplicación de ajustes de Lustre durante la inicialización del nodo. El enfoque recomendado consiste en utilizar los datos del usuario de la plantilla de lanzamiento para garantizar una configuración coherente en todos los nodos.

Estos ajustes incluyen lo siguiente:

- Optimizaciones de red y RPC
- Administración de módulos de Lustre
- Ajustes de LRU (unidad de recurso de bloqueo)
- Configuración de control de caché del cliente
- Controles de RPC para OST y MDC

Para obtener instrucciones detalladas sobre cómo implementar estos ajustes de rendimiento:

- Para optimizar el rendimiento de los nodos estándar (que no son EFA), consulte the section called “Optimización (sin EFA)” para obtener un script completo que puede agregarse a los datos de usuario de la plantilla de lanzamiento.
- Para optimizar el rendimiento de los nodos compatibles con EFA, consulte the section called “Optimización (EFA)”.

Optimización del rendimiento de Amazon FSx para Lustre en nodos (EFA)

Este tema describe cómo configurar el ajuste de Elastic Fabric Adapter (EFA) con Amazon EKS y Amazon FSx para Lustre.

> **Note**
>
> - Para obtener información sobre la creación e implementación del controlador de CSI de FSx para Lustre, consulte the section called “Implementación del controlador”.
> - Para optimizar los nodos estándar sin EFA, consulte the section called “Optimización (sin EFA)”.

Paso 1. Cómo crear un clúster de EKS

Cree un clúster con el archivo de configuración proporcionado:

```
# Create cluster using efa-cluster.yaml
eksctl create cluster -f efa-cluster.yaml
```

Ejemplo: `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)"
```

Paso 2. Crear un grupo de nodos

Cree un grupo de nodos habilitado para EFA:

```
# Create node group using efa-ng.yaml
eksctl create nodegroup -f efa-ng.yaml
```

> **Important**
>
> === Ajuste estos valores en función del entorno en la sección `# 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.
```

===

Ejemplo: `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."
```

(Opcional) Paso 3. Verificación de la configuración de EFA

Conexión al nodo mediante SSH:

```
# Get instance ID from EKS console or {aws} CLI
ssh -i /path/to/your-key.pem ec2-user@<node-internal-ip>
```

Verificación de la configuración de EFA:

```
sudo lnetctl net show
```

Comprobación de los registros de configuración:

```
sudo cat /var/log/cloud-init-output.log
```

A continuación se muestra un ejemplo del resultado esperado para `lnetctl net show`:

```
net:
    - net type: tcp
      ...
    - net type: efa
      local NI(s):
        - nid: xxx.xxx.xxx.xxx@efa
          status: up
```

Implementaciones de ejemplo

a. Creación de claim.yaml

```
#claim.yaml

apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: fsx-claim-efa
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: ""
  resources:
    requests:
      storage: 4800Gi
  volumeName: fsx-pv
```

Aplicación del archivo claim:

```
kubectl apply -f claim.yaml
```

b. Creación de pv.yaml

Actualización de `<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>
```

Aplicación del volumen persistente:

```
kubectl apply -f pv.yaml
```

c. Creación de pod.yaml

```
#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
```

Aplicación del pod:

```
kubectl apply -f pod.yaml
```

Comandos de verificación adicionales

Verificación de que el Pod monta y escribe en el sistema de archivos:

```
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
```

Conexión mediante SSH al nodo para verificar que el tráfico circula a través de EFA:

```
sudo lnetctl net show -v
```

La salida esperada mostrará las interfaces EFA junto con las estadísticas de tráfico.

Información relacionada

- the section called “Implementación del controlador”
- the section called “Optimización (sin EFA)”
- Amazon FSx for Lustre Performance
- Elastic Fabric Adapter

Optimización del rendimiento de Amazon FSx para Lustre en nodos (sin EFA)

Es posible optimizar el rendimiento de Amazon FSx para Lustre mediante la aplicación de parámetros de ajuste durante la inicialización de los nodos, con datos de usuario de la plantilla de lanzamiento.

> **Note**
>
> - Para obtener información sobre la creación e implementación del controlador de CSI de FSx para Lustre, consulte the section called “Implementación del controlador”. Para optimizar el rendimiento con nodos compatibles con EFA, consulte the section called “Optimización (EFA)”.

¿Por qué usar los datos de usuario de la plantilla de lanzamiento?

- Aplica ajustes de forma automática durante la inicialización del nodo.
- Garantiza una configuración coherente en todos los nodos.
- Elimina la necesidad de configurar los nodos de forma manual.

Información general del script de ejemplo

El script de ejemplo definido en este tema realiza las siguientes operaciones:

`# 1. Install Lustre client`

- Detecta automáticamente la versión del sistema operativo de Amazon Linux (AL).
- Instala el paquete de cliente de Lustre correspondiente.

`# 2. Apply network and RPC tunings`

- Configura `ptlrpcd_per_cpt_max=64` para el procesamiento paralelo de RPC.
- Configura `ksocklnd credits=2560` para optimizar los búferes de red.

`# 3. Load Lustre modules`

- Elimina de forma segura los módulos Lustre existentes, si están presentes.
- Se encarga del desmontaje de sistemas de archivos existentes.
- Carga módulos Lustre nuevos.

 # 4. Lustre Network Initialization

- Inicializa la configuración de red de Lustre.
- Configura los parámetros de red requeridos.

 # 5. Mount FSx filesystem

- Debe ajustar los valores de esta sección según el entorno.

 # 6. Apply tunings

- Ajustes de LRU (unidad de recurso de bloqueo):
  - `lru_max_age=600000`
  - `lru_size` calculado con base en la cantidad de CPU
- Control de caché del cliente: `max_cached_mb=64`
- Controles de RPC:
  - OST `max_rpcs_in_flight=32`
  - MDC `max_rpcs_in_flight=64`
  - MDC `max_mod_rpcs_in_flight=50`

 # 7. Verify tunings

- Verifica todos los ajustes aplicados.
- Informa si cada parámetro se aplicó correctamente o si generó una advertencia.

 # 8. Setup persistence

- También debe ajustar los valores de esta sección según el entorno.
- Detecta automáticamente la versión del sistema operativo (AL2023) para determinar qué servicio de `Systemd` aplicar.
- El sistema se inicia.
- `Systemd` inicia el servicio `lustre-tunings` (debido a `WantedBy=multi-user.target`).
- El servicio ejecuta `apply_lustre_tunings.sh`, que:
  - Verifica si el sistema de archivos está montado.

  - Monta el sistema de archivos si no está montado.
  - Espera a que se monte correctamente (hasta cinco minutos).
  - Aplica los parámetros de ajuste después de un montaje exitoso.
- Los ajustes permanecen activos hasta el reinicio.
- El servicio finaliza después de completar el script.
  - Systemd marca el servicio como “activo (finalizado)”.
- El proceso se repite en el próximo reinicio.

Crear una plantilla de lanzamiento

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. Seleccione Plantillas de lanzamiento.
3. Elija Crear plantilla de inicialización.
4. En Detalles avanzados, ubique la sección de Datos de usuario.
5. Pegue el siguiente script y actualice los valores según sea necesario.

> **Important**
>
> Ajuste estos valores según el entorno tanto en la sección `# 5. Mount FSx filesystem` como en la función `setup_persistence()` de `apply_lustre_tunings.sh` dentro de la sección `# 8. Setup persistence`:
>
> ```
> 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. Al crear grupos de nodos de Amazon EKS, seleccione esta plantilla de lanzamiento. Para obtener más información, consulte the section called “Creación”.

Información relacionada

- the section called “Implementación del controlador”
- the section called “Optimización (EFA)”
- Amazon FSx for Lustre Performance

Uso del almacenamiento de aplicaciones de alto rendimiento con FSx para NetApp ONTAP

NetApp Trident proporciona una orquestación dinámica del almacenamiento mediante un controlador compatible con la Interfaz de almacenamiento de contenedores (CSI). Esto permite a los clústeres de Amazon EKS administrar el ciclo de vida de los volúmenes persistentes (PV) respaldados por sistemas de archivos Amazon FSx para NetApp ONTAP. Tenga en cuenta que el controlador CSI de Amazon FSx para NetApp ONTAP no es compatible con los Nodos híbridos de Amazon EKS. Para comenzar, consulte Use Trident with Amazon FSx for NetApp ONTAP en la documentación de NetApp Trident.

Amazon FSx para ONTAP de NetApp es un servicio de almacenamiento que le permite lanzar y ejecutar sistemas de archivos ONTAP completamente administrados en la nube. ONTAP es la tecnología de sistema de archivos de NetApp que proporciona un conjunto ampliamente adoptado de capacidades de administración y de acceso a datos. FSx para ONTAP proporciona las características, el rendimiento y las API de los sistemas de archivos de NetApp en las instalaciones con la agilidad, escalabilidad y simplicidad de un servicio de completamente administrado de AWS. Para obtener más información, consulte la Guía del usuario de FSx para ONTAP.

Important

Si se utiliza Amazon FSx para NetApp ONTAP junto con el controlador CSI de Amazon EBS para aprovisionar volúmenes de EBS, es necesario especificar en el archivo `multipath.conf` que no se deben usar dispositivos EBS. Para conocer los métodos compatibles, consulte la Lista de exclusión del archivo de configuración. A continuación se muestra un ejemplo.

```
defaults {
       user_friendly_names yes
       find_multipaths no
     }
```

```
      blacklist {
        device {
          vendor "NVME"
          product "Amazon Elastic Block Store"
        }
      }
```

Uso del almacenamiento de datos con Amazon FSx para OpenZFS

Amazon FSx para OpenZFS es un servicio de almacenamiento de archivos completamente administrado que facilita el traslado de datos a AWS desde ZFS en las instalaciones u otros servidores de archivos basados en Linux. Puede hacerlo sin cambiar el código de la aplicación ni la forma en que administra los datos. Ofrece almacenamiento de archivos altamente confiable, escalable, eficiente y rico en características construido en el sistema de archivos OpenZFS de código abierto. Combina estas capacidades con la agilidad, la escalabilidad y la simplicidad de un servicio de AWS totalmente administrado. Para más información, consulte la Guía del usuario de Amazon FSx para OpenZFS.

El controlador de la interfaz de almacenamiento de contenedores (CSI) de FSx para OpenZFS proporciona una interfaz CSI que permite que los clústeres de Amazon EKS administren el ciclo de vida de los volúmenes de FSx para OpenZFS. Tenga en cuenta que el controlador de CSI de Amazon FSx para OpenZFS no es compatible con los Nodos híbridos de Amazon EKS. Para implementar el controlador CSI de FSx para OpenZFS en su clúster de Amazon EKS, consulte aws-fsx-openzfs-csi-driver en GitHub.

Minimización de la latencia con Amazon File Cache

Amazon File Cache es una memoria caché de alta velocidad totalmente administrada en AWS que se utiliza para procesar datos de archivos, independientemente de dónde estén almacenados los datos. Amazon File Cache carga automáticamente los datos en la memoria caché cuando se accede a ella por primera vez y publica los datos cuando no se utilizan. Para obtener más información, consulte la Guía del usuario de Amazon File Cache.

El controlador Container Storage Interface (CSI) de Amazon File Cache proporciona una interfaz CSI que permite a los clústeres de Amazon EKS administrar el ciclo de vida del almacenamiento caché de archivos de Amazon. Tenga en cuenta que el controlador CSI de Amazon File Cache no es

compatible con los Nodos híbridos de Amazon EKS. Para implementar el controlador CSI de Amazon File Cache en su clúster de Amazon EKS, consulte aws-file-cache-csi-driver en GitHub.

Acceso a los objetos de Amazon S3 mediante Mountpoint para Amazon S3 con el controlador CSI

Con el controlador de interfaz de almacenamiento de contenedores (CSI) de Mountpoint para Amazon S3, las aplicaciones de Kubernetes pueden acceder a los objetos de Amazon S3 a través de una interfaz de sistema de archivos, lo que permite lograr un rendimiento total alto sin cambiar códigos de aplicaciones. Basado en Mountpoint para Amazon S3, el controlador de CSI presenta un bucket de Amazon S3 como un volumen al que se puede acceder mediante contenedores de Amazon EKS y clústeres autoadministrados de Kubernetes.

Consideraciones

- El controlador de CSI de Mountpoint para Amazon S3 no es compatible con imágenes de contenedor basadas en Windows.
- El controlador de CSI de Mountpoint para Amazon S3 no es compatible actualmente con los Nodos híbridos de Amazon EKS.
- El controlador de CSI de Mountpoint para Amazon S3 no es compatible con AWS Fargate. Sin embargo, se admiten los contenedores que se ejecutan en Amazon EC2 (ya sea con Amazon EKS o con una instalación personalizada de Kubernetes).
- El controlador de CSI de Mountpoint para Amazon S3 solo admite el aprovisionamiento estático. No se admite el aprovisionamiento dinámico o la creación de nuevos buckets.

  > **Note**
  >
  > El aprovisionamiento estático se refiere al uso de un bucket de Amazon S3 existente que se especifica como `bucketName` en el `volumeAttributes` del objeto `PersistentVolume`. Para obtener más información, consulte Static Provisioning en GitHub.

- Los volúmenes montados con el controlador de CSI de Mountpoint para Amazon S3 no admiten todas las características del sistema de archivos POSIX. Para obtener más información sobre el comportamiento del sistema de archivos, consulte Mountpoint for Amazon S3 file system behavior en GitHub.

Para obtener información sobre la implementación del controlador, consulte the section called “Implementación del controlador”. Para obtener información sobre la eliminación del controlador, consulte the section called “Cómo eliminar el controlador”.

Implementación del controlador de Mountpoint para Amazon S3

Con el controlador de interfaz de almacenamiento de contenedores (CSI) de Mountpoint para Amazon S3, las aplicaciones de Kubernetes pueden acceder a los objetos de Amazon S3 a través de una interfaz de sistema de archivos, lo que permite lograr un rendimiento total alto sin cambiar códigos de aplicaciones.

Este procedimiento muestra cómo implementar el controlador de CSI de Mountpoint para Amazon S3 de Amazon EKS. Antes de continuar, revise las Consideraciones.

Requisitos previos

- Un proveedor existente de OpenID Connect (OIDC) de AWS Identity and Access Management (IAM) para su clúster. Para determinar si ya tiene un proveedor o para crear uno, consulte the section called “Proveedor de OIDC de IAM”.
- La versión 2.12.3 o posterior de AWS CLI instalada y configurada en su dispositivo o AWS CloudShell.
- La herramienta de línea de comandos de `kubectl` está instalada en su dispositivo o AWS CloudShell. La versión puede ser la misma o hasta una versión secundaria anterior o posterior a la versión de Kubernetes de su clúster. Por ejemplo, si la versión del clúster es `1.29`, puede usar la versión `1.28`, `1.29` o `1.30` de `kubectl` con él. Para instalar o actualizar `kubectl`, consulte the section called “Configure kubectl y eksctl”.

Paso 1: Crear una política de IAM

El controlador de CSI de Mountpoint para Amazon S3 requiere permisos de Amazon S3 para interactuar con el sistema de archivos. En esta sección se muestra cómo crear una política de IAM que conceda los permisos necesarios.

El siguiente ejemplo de política sigue las recomendaciones de permisos de IAM para Mountpoint. Como alternativa, puede utilizar la política administrada de AWS AmazonS3FullAccess, pero esta política administrada concede más permisos de los necesarios para Mountpoint.

Para obtener más información sobre los permisos recomendados por Mountpoint, consulte Mountpoint IAM permissions en GitHub.

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación izquierdo, elija Políticas.
3. En la página Políticas, seleccione Crear una política.
4. En el editor de políticas, seleccione JSON.
5. En el editor de políticas, copie y pegue lo siguiente:

> **Important**
>
> Reemplace `amzn-s3-demo-bucket1` por el nombre de su bucket de Amazon S3.

```
{
   "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/*"
            ]
        }
   ]
}
```

Los buckets de directorio, introducidos con la clase de almacenamiento Amazon S3 Express One Zone, utilizan un mecanismo de autenticación diferente al de los buckets de uso general. En lugar de utilizar acciones `s3:*`, debe utilizar la acción `s3express:CreateSession`. Para más información sobre los buckets de directorio, consulte Buckets de directorio en la Guía de usuario de Amazon S3.

A continuación, se muestra un ejemplo de política de privilegios mínimos que utilizaría para un bucket de directorios.

```
{
    "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. Elija Siguiente.
7. En la página Revisar y crear, asigne un nombre a la política. En este tutorial de ejemplo se utiliza el nombre `AmazonS3CSIDriverPolicy`.
8. Seleccione Crear política.

Paso 2: creación de un rol de IAM

El controlador de CSI de Mountpoint para Amazon S3 requiere permisos de Amazon S3 para interactuar con el sistema de archivos. En esta sección se muestra cómo crear un rol de IAM para delegar estos permisos. Para crear este rol, puede utilizar una de las siguientes herramientas:

- the section called “eksctl”
- the section called “Consola de administración de AWS”
- the section called “AWS CLI”

> **Note**
>
> La política de IAM `AmazonS3CSIDriverPolicy` se creó en la sección anterior.

eksctl

Creación del rol de IAM del controlador de CSI de Mountpoint para Amazon S3 con **`eksctl`**

Para crear el rol de IAM y la cuenta de servicio de Kubernetes, ejecute los siguientes comandos. Estos comandos también asocian la política de IAM `AmazonS3CSIDriverPolicy` al rol, anotan la cuenta de servicio de Kubernetes (`s3-csi-controller-sa`) con el Nombre de recurso de Amazon (ARN) del rol de IAM y agregan el nombre de la cuenta de servicio de Kubernetes a la política de confianza correspondiente al rol de IAM.

```
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
```

Consola de administración de AWS

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación izquierdo, elija Roles.
3. En la página Roles, elija Crear rol.
4. En la página Seleccionar entidad de confianza, haga lo siguiente:
   a. En la sección Tipo de entidad de confianza, elija Identidad web.
   b. Para Identity provider (Proveedor de identidades), elija OpenID Connect provider URL (URL del proveedor de OpenID Connect) para el clúster, como se muestra en Overview (Resumen) en Amazon EKS.

Si no se muestra ninguna URL, consulte la sección Requisitos previos.

c. En Audiencia, elija `sts.amazonaws.com`.

d. Elija Siguiente.

5. En la página Agregar permisos, haga lo siguiente:

a. En el cuadro Filtrar políticas, ingrese AmazonS3CSIDriverPolicy.

> **Note**
>
> Esta política se creó en la sección anterior.

b. Marque la casilla situada a la izquierda del resultado de `AmazonS3CSIDriverPolicy` que obtuvo en la búsqueda.

c. Elija Siguiente.

6. En la página Nombrar, revisar y crear, haga lo siguiente:

a. En Nombre del rol, ingrese un nombre único para su rol, por ejemplo, AmazonEKS_S3_CSI_DriverRole.

b. En Agregar etiquetas (Opcional), de manera opcional, agregue metadatos al rol asociando etiquetas como pares de clave-valor. Para obtener más información sobre el uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM en la Guía de usuario de IAM.

c. Seleccione Crear rol.

7. Una vez creado el rol, seleccione el rol en la consola para abrirlo y editarlo.

8. Elija la pestaña Relaciones de confianza y, a continuación, Editar política de confianza.

9. Busque la línea que se parezca a la siguiente:

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud":
 "sts.amazonaws.com"
```

Agregue una coma al final de la línea anterior y, luego, agregue la siguiente línea después de esta. Reemplace *region-code* por la región de AWS en la que se encuentra el clúster. Reemplace *EXAMPLED539D4633E53DE1B71EXAMPLE* con el ID de proveedor de OIDC de su clúster.

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub":
 "system:serviceaccount:kube-system:s3-csi-driver-sa"
```

10 Asegúrese de que el operador `Condition` esté configurado en `"StringEquals"`.

11 Elija Actualizar política para terminar.

AWS CLI

1. Vea la URL del proveedor de OIDC para su clúster. Reemplace *`my-cluster`* por el nombre de su clúster. Si la salida del comando es None, revise los Requisitos previos.

```
aws eks describe-cluster --name my-cluster --query "cluster.identity.oidc.issuer" --
output text
```

   Un ejemplo de salida sería el siguiente.

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

2. Cree el rol de IAM otorgándole a la cuenta de servicio de Kubernetes la acción `AssumeRoleWithWebIdentity`.

   a. Copie los siguientes contenidos en un archivo denominado `aws-s3-csi-driver-trust-policy.json`. Reemplace *`111122223333`* por el ID de su cuenta. Reemplace *`EXAMPLED539D4633E53DE1B71EXAMPLE`* y *`region-code`* por los valores que se devolvieron en el paso anterior.

```
{
  "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. Creación del rol. Puede cambiar el nombre de *AmazonEKS_S3_CSI_DriverRole*, pero si lo hace, asegúrese de cambiarlo también en los pasos posteriores.

```
aws iam create-role \
  --role-name AmazonEKS_S3_CSI_DriverRole \
  --assume-role-policy-document file://"aws-s3-csi-driver-trust-policy.json"
```

3. Cree un rol de IAM y adjunte la política de IAM creada previamente al rol con el siguiente comando.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonS3CSIDriverPolicy \
  --role-name AmazonEKS_S3_CSI_DriverRole
```

> **Note**
>
> La política de IAM `AmazonS3CSIDriverPolicy` se creó en la sección anterior.

4. Omita este paso si va a instalar el controlador como complemento de Amazon EKS. Para las instalaciones autoadministradas del controlador, cree cuentas de servicio de Kubernetes que estén anotadas con el ARN del rol de IAM que creó.

a. Guarde los siguientes contenidos en un archivo llamado `mountpoint-s3-service-account.yaml`. Reemplace *111122223333* por el ID de su cuenta.

```
---
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. Cree la cuenta de servicio de Kubernetes en el clúster. La cuenta de servicio de Kubernetes (`mountpoint-s3-csi-controller-sa`) está anotada con el rol de IAM que creó con el nombre *AmazonEKS_S3_CSI_DriverRole*.

```
kubectl apply -f mountpoint-s3-service-account.yaml
```

> **Note**
>
> Cuando implementa el complemento en este procedimiento, crea una cuenta de servicio que se llama `s3-csi-driver-sa` y se configura para utilizarla.

Paso 3: instalación del controlador de CSI de Mountpoint para Amazon S3

Puede instalar el controlador de CSI de Mountpoint para Amazon S3 a través del complemento de Amazon EKS. Puede usar las herramientas a continuación para agregar el complemento a su clúster:

- the section called “eksctl”
- the section called “Consola de administración de AWS”
- the section called “AWS CLI”

Como alternativa, puede instalar el controlador de CSI de Mountpoint para Amazon S3 como instalación autoadministrada. Para obtener instrucciones sobre cómo efectuar una instalación autoadministrada, consulte Installation en GitHub.

A partir de la versión `v1.8.0`, es posible configurar taints para tolerar los pods del controlador de CSI. Para ello, especifique un conjunto personalizado de taints para tolerar con `node.tolerations` o tolere todas las taints con `node.tolerateAllTaints`. Para obtener más información, consulte Taints y toleraciones en la documentación de Kubernetes.

eksctl

Agregar el complemento CSI de Amazon S3 con **`eksctl`**

Ejecute el siguiente comando. Reemplace *my-cluster* por el nombre del clúster, *111122223333* por el ID de cuenta y *AmazonEKS_S3_CSI_DriverRole* por el nombre del rol de IAM creado anteriormente.

```
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
```

Si quita la opción *--force* y cualquiera de las configuraciones del complemento de Amazon EKS entran en conflicto con la configuración existente, se produce un error al actualizar el complemento de Amazon EKS y recibe un mensaje de error para ayudarlo a resolver el conflicto. Antes de especificar esta opción, asegúrese de que el complemento de Amazon EKS no administra la configuración que necesita administrar, ya que dicha configuración se sobrescribe con esta opción. Para obtener más información acerca de otras opciones para este ajuste, consulte Addons (Complementos) en la documentación de `eksctl`. Para obtener más información sobre el campo de administración de Kubernetes de Amazon EKS, consulte the section called “Campos que puede personalizar”.

Puede personalizar `eksctl` mediante archivos de configuración. Para obtener más información, consulte Trabajar con valores de configuración en la documentación de `eksctl`. En el siguiente ejemplo se demuestra cómo tolerar todas las taints.

```
# config.yaml
...

addons:
- name: aws-mountpoint-s3-csi-driver
  serviceAccountRoleARN: arn:aws:iam::111122223333:role/AmazonEKS_S3_CSI_DriverRole
  configurationValues: |-
    node:
      tolerateAllTaints: true
```

Consola de administración de AWS

1. Abra la consola de Amazon EKS.
2. En el panel de navegación izquierdo, elija Clusters (Clústeres).
3. Seleccione el nombre del clúster para el cual desea configurar el complemento de CSI de Mountpoint para Amazon S3.
4. Elija la pestaña Complementos.
5. Escoja Obtener más complementos.
6. En la página Seleccionar complementos, haga lo siguiente:

a. En la sección Complementos de Amazon EKS, seleccione la casilla de verificación Controlador de CSI de Mountpoint para Amazon S3.

b. Elija Siguiente.

7. En la página Configurar las opciones de complementos seleccionados, haga lo siguiente:

a. Seleccione la Versión que desea utilizar.

b. En Seleccionar rol de IAM, seleccione el nombre de un rol de IAM al que le haya adjuntado la política de IAM del controlador de CSI de Mountpoint para Amazon S3.

c. (Opcional) Actualice el método de resolución de conflictos después de ampliar los ajustes de configuración opcionales. Si selecciona Anular, es posible que una o varias configuraciones del complemento existente se sobrescriban con la configuración del complemento de Amazon EKS. Si no habilita esta opción y hay un conflicto con la configuración existente, la operación falla. Puede utilizar el mensaje de error resultante para solucionar el conflicto. Antes de seleccionar esta opción, asegúrese de que el complemento de Amazon EKS no administra las configuraciones que se necesitan autoadministrar.

d. (Opcional) Configure las tolerancias en el campo Valores de configuración después de ampliar los ajustes de configuración opcionales.

e. Elija Siguiente.

8. En la página Revisar y añadir, elija Crear. Una vez finalizada la instalación del complemento, verá el complemento instalado.

AWS CLI

Adición del complemento de CSI de Mountpoint para Amazon S3 mediante la AWS CLI

Ejecute el siguiente comando. Reemplace *my-cluster* por el nombre del clúster, *111122223333* por el ID de cuenta y *AmazonEKS_S3_CSI_DriverRole* por el nombre del rol creado anteriormente.

```
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
```

Puede personalizar el comando con la marca `--configuration-values`. El siguiente ejemplo alternativo muestra cómo tolerar todas las taints.

```
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}}'
```

Paso 4: configuración de Mountpoint para Amazon S3

En la mayoría de los casos, puede configurar Mountpoint para Amazon S3 solo con un nombre de bucket. Para obtener instrucciones sobre la configuración de Mountpoint para Amazon S3, consulte Configuring Mountpoint for Amazon S3 en GitHub.

Paso 5: implementación de una aplicación de muestra

Puede implementar el aprovisionamiento estático en el controlador de un bucket de Amazon S3 existente. Para obtener más información, consulte Static provisioning en GitHub.

Cómo eliminar el complemento de Amazon EKS Mountpoint para Amazon S3

Tiene dos opciones para eliminar el controlador de CSI de Mountpoint para Amazon S3.

- Conservar el software del complemento en el clúster: esta opción elimina la administración de Amazon EKS de cualquier configuración. También elimina la capacidad de Amazon EKS de notificarle las actualizaciones y actualizar de forma automática el complemento de Amazon EKS después de iniciar una actualización. Sin embargo, conserva el software del complemento en el clúster. Esta opción hace que la instalación sea autoadministrada, en lugar de un complemento de Amazon EKS. Con esta opción, no hay tiempo de inactividad para el complemento. Los comandos de este procedimiento utilizan esta opción.
- Eliminar por completo el software del complemento del clúster: recomendamos que elimine el complemento de Amazon EKS del clúster solo si no hay recursos en el clúster que dependan de él. Para hacer esta opción, elimine `--preserve` del comando que utiliza en este procedimiento.

Si el complemento tiene una cuenta de IAM asociada, esta no se elimina.

Puede usar las herramientas a continuación para eliminar el complemento Amazon S3 CSI.

- the section called “eksctl”

- the section called “Consola de administración de AWS”
- the section called “AWS CLI”

eksctl

Eliminar el complemento Amazon S3 CSI con **`eksctl`**

Reemplace *`my-cluster`* por el nombre de su clúster y ejecute el siguiente comando.

```
eksctl delete addon --cluster my-cluster --name aws-mountpoint-s3-csi-driver --preserve
```

Consola de administración de AWS

1. Abra la consola de Amazon EKS.
2. En el panel de navegación izquierdo, elija Clusters (Clústeres).
3. Elija el nombre del clúster para el que desea eliminar el complemento CSI de Amazon EBS.
4. Elija la pestaña Complementos.
5. Elija Controlador de CSI de Mountpoint para Amazon S3.
6. Elija Eliminar.
7. En el cuadro de diálogo de confirmación Eliminar: aws-mountpoint-s3-csi-driver, haga lo siguiente:
   a. Si desea que Amazon EKS deje de administrar la configuración del complemento, seleccione Conservar en clúster. Haga esto si desea retener el software del complemento en el clúster. Esto es para que pueda administrar todas las configuraciones del complemento por su cuenta.
   b. Escriba `aws-mountpoint-s3-csi-driver`.
   c. Seleccione Eliminar.

AWS CLI

Eliminar el complemento Amazon S3 CSI con la AWS CLI

Reemplace *`my-cluster`* por el nombre de su clúster y ejecute el siguiente comando.

```
aws eks delete-addon --cluster-name my-cluster --addon-name aws-mountpoint-s3-csi-
driver --preserve
```

Habilitación de la funcionalidad de instantáneas para volúmenes de CSI

La funcionalidad de instantáneas permite hacer copias de un momento dado de los datos. Para que esta capacidad funcione en Kubernetes, necesita un controlador de CSI compatible con instantáneas (como el controlador de CSI de Amazon EBS) y un controlador de instantáneas de CSI. El controlador de instantáneas está disponible como complemento administrado por Amazon EKS o como instalación autoadministrada.

Aquí se incluyen algunos aspectos que debe tener en cuenta cuando se utiliza el controlador de instantáneas CSI.

- El controlador de instantáneas debe instalarse junto con un controlador de CSI con funcionalidad de captura de instantáneas. Para obtener instrucciones acerca de cómo instalar el controlador de CSI de Amazon EBS, consulte the section called “Amazon EBS”.
- Kubernetes no admite instantáneas de volúmenes que se entreguen mediante la migración de CSI, como los volúmenes de Amazon EBS que utilizan `StorageClass` con un aprovisionador `kubernetes.io/aws-ebs`. Los volúmenes deben crearse con un `StorageClass` que haga referencia al aprovisionador de controladores CSI `ebs.csi.aws.com`.
- El modo automático de Amazon EKS no incluye el controlador de instantáneas. La capacidad de almacenamiento del modo automático de EKS es compatible con el controlador de instantáneas.

Le recomendamos que instale el controlador de instantáneas CSI a través del complemento administrado de Amazon EKS. Este complemento incluye las definiciones de recursos personalizadas (CRD) necesarias para crear y administrar instantáneas en Amazon EKS. Para agregar un complemento de Amazon EKS al clúster, consulte the section called “Cómo crear un complemento”. Para obtener más información sobre los complementos, consulte the section called “Complementos de Amazon EKS”.

Como alternativa, si prefiere una instalación autoadministrada del controlador de instantáneas CSI de Amazon EBS, consulte Uso en la versión anterior de `external-snapshotter` de Kubernetes en GitHub.

Configuración de redes para clústeres de Amazon EKS

El clúster de Amazon EKS se crea en una VPC. El complemento de interfaz de red de contenedores (CNI) de Amazon VPC proporciona la red de pods para los nodos que se ejecutan en la infraestructura de AWS. Si ejecuta nodos en una infraestructura propia, consulte the section called "Cómo configurar una CNI". En este capítulo, se detallan los siguientes temas para obtener más información sobre redes para el clúster.

Temas

- Cómo agregar una subred de VPC existente a un clúster de Amazon EKS desde la consola de administración
- Requisitos de red de Amazon EKS para VPC y subredes
- Creación de una Amazon VPC para su clúster de Amazon EKS
- Revisión de requisitos de grupos de seguridad de Amazon EKS para clústeres
- Administración de complementos de red para clústeres de Amazon EKS

Cómo agregar una subred de VPC existente a un clúster de Amazon EKS desde la consola de administración

1. Vaya al clúster de EKS en la consola de administración
2. En la pestaña Redes, seleccione Administrar recursos de VPC
3. En el menú desplegable Subredes, seleccione subredes adicionales de la VPC del clúster.

Para crear una nueva subred de VPC:

- Revise los requisitos de subred de EKS
- Consulte Crear una subred en la Guía del usuario de Amazon Virtual Private Cloud.

Requisitos de red de Amazon EKS para VPC y subredes

Al crear un clúster, especifica una VPC y al menos dos subredes que se encuentran en diferentes zonas de disponibilidad. En este tema se proporciona una descripción general de los requisitos y consideraciones específicos de Amazon EKS para la VPC y las subredes que utiliza con el clúster. Si

no tiene una VPC que usar con Amazon EKS, consulte the section called “Creación de una VPC”. Si está creando un clúster local o extendido en AWS Outposts, consulte the section called “Creación de una VPC y de subredes” en lugar de este tema. El contenido de este tema se aplica a los clústeres de Amazon EKS con nodos híbridos. Para conocer los requisitos de red adicionales de los nodos híbridos, consulte the section called “Preparación de las redes”.

Requisitos y consideraciones de la VPC

Al crear un clúster, la VPC que especifique debe cumplir los siguientes requisitos y consideraciones:

- La VPC debe tener un número suficiente de direcciones IP disponibles para el clúster, los nodos y otros recursos de Kubernetes que desee crear. Si la VPC que desea utilizar no tiene un número suficiente de direcciones IP, intente aumentar el número de direcciones IP disponibles.

  Para ello, puede actualizar la configuración del clúster para cambiar las subredes y los grupos de seguridad que utiliza el clúster. Puede actualizar desde la Consola de administración de AWS, la última versión de la AWS CLI, AWS CloudFormation, y `eksctl` versión `v0.164.0-rc.0` o posterior. Es posible que tenga que hacer esto para proporcionar a las subredes más direcciones IP disponibles con las que poder actualizar correctamente la versión de un clúster.

  > **Important**
  >
  > Todas las subredes que agregue deben estar en el mismo conjunto de zonas de disponibilidad proporcionadas originalmente cuando creó el clúster. Las nuevas subredes deben cumplir todos los demás requisitos; por ejemplo, deben tener suficientes direcciones IP.
  >
  > Por ejemplo, suponga que creó un clúster y especificó cuatro subredes. En el orden en que las especificó, la primera subred está en la zona de disponibilidad `us-west-2a`, la segunda y tercera subredes están en la zona de disponibilidad `us-west-2b`, y la cuarta subred está en la zona de disponibilidad `us-west-2c`. Si desea cambiar las subredes, debe proporcionar al menos una subred en cada una de las tres zonas de disponibilidad, y las subredes deben estar en la misma VPC que las subredes originales.

  Si necesita más direcciones IP de las que tienen los bloques de CIDR de la VPC, puede agregar bloques de CIDR adicionales asociando bloques de enrutamiento entre dominios sin clases (CIDR) adicionales a la VPC. Puede asociar bloques de CIDR privados (RFC 1918) y públicos (no RFC 1918) a su VPC antes o después de crear el clúster.

Puede agregar nodos que utilicen el nuevo bloque de CIDR inmediatamente después de hacerlo. Sin embargo, dado que el plano de control reconoce el nuevo bloque de CIDR solo después de completar la conciliación, un clúster puede tardar hasta una hora en reconocer un bloque de CIDR que ha asociado a una VPC. A continuación, puede ejecutar los comandos `kubectl attach`, `kubectl cp`, `kubectl exec`, `kubectl logs` y `kubectl port-forward` (estos comandos utilizan la `kubelet API`) para los nodos y los pods del nuevo bloque de CIDR. Además, si tiene pods que funcionan como backend de webhook, debe esperar a que se complete la conciliación del plano de control.

- Evite la superposición de rangos de direcciones IP al conectar su clúster de EKS a otras VPC mediante la puerta de enlace de tránsito, la interconexión de VPC u otras configuraciones de red. Los conflictos de CIDR se producen cuando el CIDR del servicio del clúster de EKS se superpone con el CIDR de una VPC conectada. En estos escenarios, las direcciones IP del servicio tienen prioridad sobre los recursos de las VPC conectadas con la misma dirección IP, aunque el enrutamiento del tráfico puede resultar impredecible y es posible que las aplicaciones no puedan conectarse a los recursos previstos.

  Para evitar conflictos de CIDR, asegúrese de que el CIDR del servicio de EKS no se superponga con ningún CIDR de VPC conectado y mantenga un registro centralizado de todas las asignaciones de CIDR. Si encuentra superposiciones de CIDR, puede usar una puerta de enlace de tránsito con una VPC de servicios compartidos. Para obtener más información, consulte VPC aisladas con servicios compartidos y Patrones de conservación de direcciones IP enrutables de VPC de Amazon EKS en una red híbrida. Consulte también la sección Communication across VPCs de la página VPC and Subnet Considerations en la Guía de prácticas recomendadas de EKS.

- Si quiere que Kubernetes asigne direcciones `IPv6` a pods y servicios, asocie un bloque de CIDR `IPv6` con su VPC. Para obtener más información, consulte Asociar un bloque de CIDR IPv6 a su VPC en la Guía del usuario de Amazon VPC. No puede usar direcciones `IPv6` con pods y servicios que se ejecuten en nodos híbridos y no puede usar nodos híbridos con clústeres configurados con la familia de direcciones IP `IPv6`.
- La VPC debe tener un nombre de host `DNS` y admitir la resolución `DNS`. De lo contrario, los nodos no podrán registrarse en su clúster. A fin de obtener más información, consulte Atributos de DNS para su VPC en la Guía del usuario de Amazon VPC.
- Es posible que la VPC requiera puntos de conexión de VPC mediante AWS PrivateLink. Para obtener más información, consulte the section called “Requisitos y consideraciones de la subred”.

Si ha creado un clúster con Kubernetes `1.14` o anterior, Amazon EKS agregó la siguiente etiqueta a la VPC:

Clave	Valor
`kubernetes.io/cluster/` *`my-cluster`*	`owned`

Esta etiqueta solo la utilizó Amazon EKS. Puede quitar la etiqueta sin afectar a sus servicios. No se utiliza con clústeres que son versión `1.15` o posterior.

Requisitos y consideraciones de la subred

Al crear un clúster, Amazon EKS crea de 2 a 4 interfaces de red elásticas en las subredes que especifique. Estas interfaces de red permiten la comunicación entre el clúster y la VPC. Estas interfaces de red también activan características de Kubernetes que utilizan la `kubelet API`. Las conexiones a la API `kubelet` se utilizan en los comandos `kubectl attach`, `kubectl cp`, `kubectl exec`, `kubectl logs` y `kubectl port-forward`. Cada interfaz de red creada con Amazon EKS cuenta con el texto `Amazon EKS` *`cluster-name`* en su descripción.

Amazon EKS puede crear sus interfaces de red en cualquier subred que especifique al crear un clúster. Puede cambiar en qué subredes Amazon EKS crea sus interfaces de red después de crear el clúster. Al actualizar la versión de Kubernetes de un clúster, Amazon EKS elimina las interfaces de red originales que creó y crea nuevas interfaces de red. Estas interfaces de red se pueden crear en las mismas subredes que las interfaces de red originales o en subredes distintas de las interfaces de red originales. Para controlar en qué subredes se crean las interfaces de red, puede limitar el número de subredes especificadas a solo dos al crear un clúster, o bien actualizar las subredes después de crear el clúster.

Requisitos de la subred para los clústeres

Las subredes que especifique al crear o actualizar un clúster deben cumplir los siguientes requisitos:

- Las subredes deben tener al menos seis direcciones IP para que Amazon EKS las utilice. Sin embargo, recomendamos al menos 16 direcciones IP.
- Las subredes deben estar en al menos dos zonas de disponibilidad diferentes.
- Las subredes no pueden residir en AWS Outposts ni AWS Wavelength. Sin embargo, si los tiene en la VPC, puede implementar nodos autoadministrados y recursos de Kubernetes para este tipo

de subredes. Para obtener más información sobre los nodos autoadministrados, consulte the section called “Nodos autoadministrados”.

- Las subredes pueden ser públicas o privadas. Sin embargo, le recomendamos que especifique subredes privadas, si es posible. Una subred pública es una subred con una tabla de enrutamiento que incluye una ruta a una puerta de enlace de Internet, mientras que una subred privada es una subred con tabla de enrutamiento que no incluye ruta a la puerta de enlace de internet.
- Las subredes no pueden residir en las siguientes zonas de disponibilidad:

Región de AWS	Nombre de la región	ID de zona de disponibilidad que no están permitidos
us-east-1	Este de EE. UU. (Norte de Virginia)	use1-az3
us-west-1	Oeste de EE. UU. (Norte de California)	usw1-az2
ca-central-1	Canadá (centro)	cac1-az3

Uso de familias de direcciones IP por componente

La siguiente tabla contiene la familia de direcciones IP que utiliza cada componente de Amazon EKS. Puede utilizar una traducción de direcciones de red (NAT) u otro sistema de compatibilidad para conectarse a estos componentes desde las direcciones IP de origen en familias con el valor “No” para una entrada de la tabla.

La funcionalidad puede variar según la configuración de la familia de IP (`ipFamily`) del clúster. Esta configuración cambia el tipo de direcciones IP utilizadas para el bloque de CIDR que Kubernetes asigna a los servicios. Un clúster con el valor de configuración IPv4 se denomina clúster IPv4, mientras que un clúster con el valor de configuración IPv6 se denomina clúster IPv6.

Componente	Direcciones IPv4	Direcciones IPv6	Direcciones de pila doble
Punto de conexión público de la API de EKS	Sí[1,3]	Sí[1,3]	Sí[1,3]
Punto de conexión de VPC de la API de EKS	Sí	No	No
Punto de conexión público de la API de autenticación de EKS (Pod Identity de EKS)	Sí[1]	Sí[1]	Sí[1]
Punto de conexión de VPC de la API de autenticación de EKS (Pod Identity de EKS)	Sí[1]	Sí[1]	Sí[1]
`IPv4` Punto de conexión público de clúster de Kubernetes[2]	Sí	No	No
`IPv4` Punto de conexión privado de clúster de Kubernetes[2]	Sí	No	No
`IPv6` Punto de conexión público de clúster de Kubernetes[2]	Sí[1,4]	Sí[1,4]	Sí[4]
`IPv6` Punto de conexión privado de	Sí[1,4]	Sí[1,4]	Sí[4]

Componente	Direcciones IPv4	Direcciones IPv6	Direcciones de pila doble
clúster de Kubernetes[2]			
Subredes del clúster de Kubernetes	Sí[2]	No	Sí[2]
Direcciones IP principales del nodo	Sí[2]	No	Sí[2]
Rango de CIDR de clúster para direcciones IP de servicio	Sí[2]	Sí[2]	No
Direcciones IP del pod de CNI de VPC	Sí[2]	Sí[2]	No
URL de los emisores de OIDC para IRSA	Sí[1,3]	Sí[1,3]	Sí[1,3]

Note

[1] El punto de conexión es de pila doble con las direcciones `IPv4` y `IPv6`. Las aplicaciones fuera de AWS, los nodos del clúster y los pods dentro del clúster pueden alcanzar este punto de conexión mediante `IPv4` o `IPv6`.

[2] Cuando se crea un clúster, se elige entre un clúster `IPv4` y un clúster `IPv6` en la configuración de familia de IP (`ipFamily`) del clúster, y esto no se puede cambiar. En su lugar, es necesario elegir una configuración diferente al momento de crear otro clúster y migrar las cargas de trabajo.

[3] El punto de conexión de doble pila se introdujo en agosto de 2024. Para usar los puntos de conexión de doble pila con la AWS CLI, consulte la configuración de los puntos de conexión de doble pila y FIPS en la Guía de referencia de SDK y herramientas de AWS. A continuación se enumeran los nuevos puntos de conexión:

Punto de conexión público de la API de EKS

```
eks.region.api.aws
```

URL de los emisores de OIDC para IRSA

```
oidc-eks.region.api.aws
```

[4] El punto de conexión de clúster de doble pila se introdujo en octubre de 2024. EKS crea el siguiente punto de conexión para los nuevos clústeres que se creen después de esta fecha y que seleccionen `IPv6` en la configuración de familia IP (ipFamily) del clúster:

Punto de conexión público o privado del clúster de EKS

```
eks-cluster.region.api.aws
```

Requisitos de la subred para los nodos

Puede implementar nodos y recursos de Kubernetes en las mismas subredes que especifique al crear el clúster. Sin embargo, esto no es necesario. Esto se debe a que también puede implementar nodos y recursos de Kubernetes en subredes que no especificó al crear el clúster. Si implementa nodos en subredes diferentes, Amazon EKS no crea interfaces de red de clúster en esas subredes. Cualquier subred en la que implemente nodos y recursos de Kubernetes debe cumplir los siguientes requisitos:

- Las subredes deben tener suficientes direcciones IP disponibles para implementar todos sus nodos y recursos de Kubernetes.
- Si quiere que Kubernetes asigne direcciones `IPv6` a pods y servicios, debe tener un bloque de CIDR `IPv6` y un bloque de CIDR `IPv4` asociado a su subred. Para obtener más información, consulte Asociar un bloque de CIDR IPv6 a su subred en la Guía del usuario de Amazon VPC. Las tablas de enrutamiento asociadas a las subredes deben incluir rutas a direcciones `IPv4` y `IPv6`. Para obtener más información, consulte Rutas en la Guía del usuario de Amazon VPC. A los pods solo se les asigna una dirección `IPv6`. Sin embargo, a las interfaces de red que Amazon EKS crea para el clúster y los nodos se les asigna una dirección `IPv4` y `IPv6`.
- Si necesita acceso entrante desde Internet a sus pods, asegúrese de tener al menos una subred pública con suficientes direcciones IP disponibles para implementar equilibradores de carga e ingresar a ellos. Puede implementar un equilibrador de carga en una subred pública. Los

equilibradores de carga pueden equilibrar la carga de los pods en subredes privadas o públicas. Recomendamos implementar los nodos en subredes privadas, si es posible.

- Si planea implementar nodos en una subred pública, la subred debe asignar automáticamente direcciones `IPv4` públicas o direcciones `IPv6`. Si implementa nodos en una subred privada que tiene un bloque de CIDR `IPv6` asociado, la subred privada también debe asignarse automáticamente a direcciones `IPv6`. Si utilizó una plantilla de AWS CloudFormation proporcionada por Amazon EKS para implementar la VPC después del 26 de marzo de 2020, esta configuración está habilitada. Si ha utilizado las plantillas para implementar la VPC antes de esta fecha o utiliza su propia VPC, debe habilitar esta configuración manualmente. Para ver la plantilla, consulte the section called “Creación de una VPC”. Para obtener más información, consulte Modificación del atributo de direcciones IPv4 públicas de la subred y Modificación del atributo de direcciones IPv6 de la subred en la Guía de usuario de Amazon VPC.
- Si la subred en la que implementa un nodo es una subred privada y su tabla de enrutamiento no incluye ninguna ruta a un dispositivo de traducción de direcciones de red (NAT) (`IPv4`) o una puerta de enlace solo de salida (`IPv6`), agregue puntos de conexión de la VPC mediante AWS PrivateLink a su VPC. Se necesitan puntos de conexión de VPC para todos los servicios de AWS con los que sus nodos y pods necesitan comunicarse. Algunos ejemplos incluyen Amazon ECR, equilibradores de carga elástica, Amazon CloudWatch, AWS Security Token Service y Amazon Simple Storage Service (Amazon S3). El punto de conexión debe incluir la subred en la que se encuentran los nodos. No todos los servicios de AWS admiten los puntos de conexión de VPC. Para obtener más información, consulte ¿Qué es AWS PrivateLink? y Servicios de AWS que se integran con AWSPrivateLink. Para obtener una lista de más requisitos de Amazon EKS, consulte the section called “Clústeres privados”.
- Si desea implementar equilibradores de carga en una subred, la subred debe tener la siguiente etiqueta:
  - Subredes privadas

Clave	Valor
`kubernetes.io/role/internal-elb`	`1`

  - Subredes públicas

Clave	Valor
kubernetes.io/role/elb	1

Cuando se crea un clúster de Kubernetes con una versión 1.18 o anterior, Amazon EKS agrega la siguiente etiqueta a todas las subredes especificadas.

Clave	Valor
kubernetes.io/cluster/ *my-cluster*	shared

Cuando crea un nuevo clúster de Kubernetes ahora, Amazon EKS no agrega la etiqueta a sus subredes. Si la etiqueta estaba en subredes usadas por un clúster que anteriormente tenía una versión anterior a la 1.19, la etiqueta no se eliminó automáticamente de las subredes cuando el clúster se actualizó a una versión más reciente. La versión 2.1.1 o anterior del controlador de equilibrador de carga de AWS requiere esta etiqueta. Si está usando una versión más reciente del controlador de equilibrador de carga, puede eliminar la etiqueta sin interrumpir sus servicios. Para obtener más información sobre el controlador, consulte the section called “ AWS Controlador del equilibrador de carga de ”.

Si ha implementado una VPC mediante eksctl o con alguna de las plantillas de VPC de AWS CloudFormation de Amazon EKS, aplica lo siguiente:

- A partir del 26 de marzo de 2020: las subredes públicas asignan de manera automática direcciones IPv4 públicas a los nodos nuevos implementados en subredes públicas.
- Antes del 26 de marzo de 2020: las subredes públicas no asignan de forma automática las direcciones IPv4 públicas a los nodos nuevos implementados en subredes públicas.

Este cambio afecta a los nuevos grupos de nodos implementados en subredes públicas de las siguientes formas:

- Grupos de nodos administrados: si el grupo de nodos se implementa en una subred pública el 22 de abril de 2020 o después, la asignación automática de direcciones IP públicas debe estar habilitada para la subred pública. Para obtener más información, consulte Modificación del atributo de direcciones IPv4 públicas de su subred.

- Grupos de nodos autoadministrados de Linux, Windows o Arm: si el grupo de nodos se implementa en una subred pública a partir del 26 de marzo de 2020, la asignación automática de direcciones IP públicas deben estar habilitadas para la subred pública. De lo contrario, los nodos deben iniciarse con una dirección IP pública en su lugar. Para obtener más información, consulte Modificación del atributo de direcciones IPv4 públicas de su subred o Asignación de una dirección IPv4 pública durante el lanzamiento de la instancia.

Requisitos y consideraciones de la subred compartida

Puede usar Uso compartido de VPC para compartir subredes con otras cuentas de AWS dentro de la misma AWS Organizations. Puede crear clústeres de Amazon EKS en subredes compartidas, teniendo en cuenta las siguientes consideraciones:

- El propietario de la subred de VPC debe compartir una subred con una cuenta participante antes de que esa cuenta pueda crear un clúster de Amazon EKS en ella.
- No puede lanzar recursos mediante el grupo de seguridad predeterminado de la VPC porque pertenece al propietario. Además, los participantes no pueden lanzar recursos mediante grupos de seguridad que sean propiedad de otros participantes o del propietario.
- En una subred compartida, el participante y el propietario controlan por separado los grupos de seguridad de cada cuenta respectiva. El propietario de la subred puede ver estos grupos de seguridad creados por los participantes, pero no puede realizar ninguna acción en ellos. Si el propietario de la subred quiere eliminar o modificar estos grupos de seguridad, el participante que ha creado el grupo de seguridad debe realizar la acción.
- Si un participante crea un clúster, se deben tener en cuenta las siguientes consideraciones:
  - El rol de IAM de clúster y los roles de IAM de nodo deben crearse en esa cuenta. Para obtener más información, consulte the section called “Rol de IAM de clúster” y the section called “Rol de IAM de nodo”.
  - Todos los nodos debe crearlos el mismo participante, incluidos los grupos de nodos administrados.
- El propietario de la VPC compartida no puede ver, actualizar ni eliminar un clúster que un participante cree en la subred compartida. Esto se suma a los recursos de VPC a los que cada cuenta tiene un acceso diferente. Para obtener más información, consulte Responsabilidades y permisos de los propietarios y los participantes en la Guía del usuario de Amazon VPC.
- Si usa la característica de redes personalizadas del complemento CNI de Amazon VPC para Kubernetes, debe utilizar las asignaciones de ID de zona de disponibilidad que figuran en la cuenta

del propietario para crear cada `ENIConfig`. Para obtener más información, consulte the section called “Redes personalizadas”.

Para obtener más información sobre el uso compartido de la subred de VPC, consulte Compartir su VPC con otras cuentas en la Guía del usuario de Amazon VPC.

Creación de una Amazon VPC para su clúster de Amazon EKS

Puede utilizar Amazon Virtual Private Cloud (Amazon VPC) para lanzar recursos de AWS en una red virtual que haya definido. Esta red virtual es prácticamente idéntica a una red tradicional que podría operar en su propio centro de datos. Sin embargo, incluye los beneficios que supone utilizar la infraestructura escalable de Amazon Web Services. Le recomendamos que conozca a fondo el servicio Amazon VPC antes de implementar clústeres de Amazon EKS en producción. Para obtener más información, consulte la Guía del usuario de Amazon VPC.

Un clúster de Amazon EKS, nodos y recursos de Kubernetes se implementan en una VPC. Si desea utilizar una VPC existente con Amazon EKS, dicha VPC debe cumplir los requisitos que se describen en the section called “Requisitos de VPC y subred”. En este tema, se describe cómo puede crear una VPC que cumpla los requisitos de Amazon EKS mediante una plantilla de AWS CloudFormation proporcionada por Amazon EKS. Una vez que haya implementado una plantilla, podrá ver los recursos creados por la plantilla para saber exactamente qué recursos creó y la configuración de esos recursos. Si utiliza nodos híbridos, la VPC debe tener rutas en la tabla de enrutamiento correspondientes a la red en las instalaciones. Para obtener más información sobre los requisitos de red de los nodos híbridos, consulte the section called “Preparación de las redes”.

Requisitos previos

Para crear una VPC para Amazon EKS, debe tener los permisos de IAM necesarios para crear recursos de Amazon VPC. Estos recursos son VPC, subredes, grupos de seguridad, tablas de enrutamiento y rutas y puertas de enlace de Internet y NAT. Para obtener más información, consulte Crear una VPC con una política de ejemplo de subred pública en la Guía del usuario de Amazon VPC y en la lista completa de Acciones en la Referencia de autorizaciones de servicio.

Puede crear una VPC con subredes públicas y privadas, solo subredes públicas o solo subredes privadas.

Subredes públicas y privadas

Esta VPC tiene dos subredespúblicas y dos privadas. Una subred pública es una subred asociada a la tabla de enrutamiento con ruta a la puerta de enlace de internet. Sin embargo, la tabla de enrutamiento de una subred privada no tiene ninguna ruta a una puerta de enlace de Internet. Una subred pública y una subred privada se implementan en la misma zona de disponibilidad. Las otras subredes públicas y privadas se implementan en una segunda zona de disponibilidad en la misma región de AWS. Recomendamos esta opción para la mayoría de las implementaciones.

Con esta opción, puede implementar los nodos en subredes privadas. Esta opción permite a Kubernetes implementar los equilibradores de carga en las subredes públicas que pueden equilibrar la carga de tráfico a pods que se ejecutan en los nodos en las subredes privadas. Las direcciones `IPv4` públicas se asignan de forma automática a nodos implementados en subredes públicas, pero las direcciones `IPv4` públicas no se asignan a nodos implementados en subredes privadas.

También puede asignar direcciones `IPv6` a nodos en subredes públicas y privadas. Los nodos de las subredes privadas pueden comunicarse con el clúster y otros servicios de AWS. Los pods pueden comunicarse por Internet a través de una puerta de enlace de NAT mediante direcciones `IPv4` o una puerta de enlace de Internet de solo salida con direcciones `IPv6` que se implementa en cada zona de disponibilidad. Se implementa un grupo de seguridad que dispone de reglas que deniegan todo el tráfico entrante de fuentes distintas del clúster o los nodos, pero permite todo el tráfico saliente. Las subredes están etiquetadas para que Kubernetes pueda implementar balanceadores de carga en ellas.

a. Abra la Consola de AWS CloudFormation.

b. En la barra de navegación, seleccione una región de AWS compatible con Amazon EKS.

c. Elija Create stack (Crear pila), With new resources (standard) (Con nuevos recursos [estándar]).

d. En Prerequisite - Prepare template (Requisito previo - Preparar plantilla), asegúrese de que esté seleccionada la opción Template is ready (La plantilla está lista) y, a continuación, en Specify template (Especificar plantilla), seleccione Amazon S3 URL (URL de Amazon S3).

e. Puede crear una VPC que admita únicamente `IPv4` o una VPC que admita `IPv4` y `IPv6`. Pegue una de las siguientes URL en el área de texto de Amazon S3 URL (URL de Amazon S3) y elija Next (Siguiente):

   - `IPv4`

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml
```

- IPv4 y IPv6

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-ipv6-vpc-public-private-subnets.yaml
```

a. En la página Especificar detalles de pila, modifique los parámetros y, a continuación, elija Siguiente.

   - Nombre de pila: elija un nombre para su pila de AWS CloudFormation. Por ejemplo, puede usar el nombre de la plantilla que usó en el paso anterior. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster.
   - VpcBlock: elija un rango de CIDR IPv4 para su VPC. A cada nodo, pod y equilibrador de carga que implemente se le asigna una dirección IPv4 de este bloque. Los valores predeterminados de IPv4 proporcionan suficientes direcciones IP para la mayoría de las implementaciones, pero, si no lo hacen, puede cambiarlos. Para obtener más información, consulte Tamaño de la subred y VPC en la Guía del usuario de Amazon VPC. También puede agregar bloques de CIDR adicionales a la VPC una vez que la cree. Si crea una VPC IPv6, los rangos de CIDR IPv6 se asignan de forma automática desde el espacio de direcciones de unidifusión global de Amazon.
   - PublicSubnet01Block: especifique un bloque de CIDR IPv4 para la subred pública 1. El valor predeterminado proporciona suficientes direcciones IP para la mayoría de las implementaciones, pero si no lo hace, puede cambiarlo. Si crea una VPC IPv6, este bloque se especifica en su nombre dentro de la plantilla.
   - PublicSubnet02Block: especifique un bloque de CIDR IPv4 para la subred pública 2. El valor predeterminado proporciona suficientes direcciones IP para la mayoría de las implementaciones, pero si no lo hace, puede cambiarlo. Si crea una VPC IPv6, este bloque se especifica en su nombre dentro de la plantilla.
   - PrivateSubnet01Block: especifique un bloque de CIDR IPv4 para la subred privada 1. El valor predeterminado proporciona suficientes direcciones IP para la mayoría de las implementaciones, pero si no lo hace, puede cambiarlo. Si crea una VPC IPv6, este bloque se especifica en su nombre dentro de la plantilla.

- PrivateSubnet02Block: especifique un bloque de CIDR `IPv4` para la subred privada 2. El valor predeterminado proporciona suficientes direcciones IP para la mayoría de las implementaciones, pero si no lo hace, puede cambiarlo. Si crea una VPC `IPv6`, este bloque se especifica en su nombre dentro de la plantilla.

b. (Opcional) En la página Configurar las opciones de la pila, etiquete los recursos de la pila y, luego, elija Siguiente.

c. En la página Review (Revisar), elija Create stack (Crear pila).

d. Una vez creada la pila, selecciónela en la consola y elija Outputs (Salidas).

e. Registre el VpcId de la VPC que ha creado. Necesita esto cuando al crear su clúster y nodos.

f. Registre los SubnetIds de las subredes que se crearon y si las creó como subredes públicas o privadas. Necesita al menos dos de ellos al crear el clúster y los nodos.

g. Si has creado un VPC `IPv4`, omita este paso. Si ha creado una VPC `IPv6`, debe habilitar la opción de asignación automática de dirección `IPv6` para las subredes públicas que creó la plantilla. Esta configuración ya está habilitada para las subredes privadas. Complete los siguientes pasos para habilitar la configuración:

   i. Abra la consola de Amazon VPC en https://console.aws.amazon.com/vpc/.

   ii. En el panel de navegación izquierdo, elija Subnets (Subredes).

   iii. Seleccione una de las subredes públicas (***stack-name***/SubnetPublic01 o ***stack-name***/SubnetPublic02, que contienen la palabra público), elija Acciones y, luego, Editar la configuración de la subred.

   iv. Elija la casilla de verificación Enable auto-assign IPv6 address (Habilitar asignación automática de direcciones IPv6) y, luego, elija Save (Guardar).

   v. Complete los pasos anteriores de nuevo para la otra subred pública.

Solo subredes públicas

Esta VPC tiene tres subredes públicas que se implementan en zonas de disponibilidad diferentes en una región de AWS. A todos los nodos se les asignan de forma automática direcciones `IPv4` públicas y pueden enviar y recibir tráfico de Internet a través de una puerta de enlace de Internet. Se implementa un grupo de seguridad que deniega todo el tráfico entrante y permite todo el tráfico saliente. Las subredes están etiquetadas para que Kubernetes pueda implementar balanceadores de carga en ellas.

a. Abra la Consola de AWS CloudFormation.

b. En la barra de navegación, seleccione una región de AWS compatible con Amazon EKS.

c. Elija Create stack (Crear pila), With new resources (standard) (Con nuevos recursos [estándar]).

d. En Prepare template (Preparar la plantilla), asegúrese de que se seleccione Template is ready (La plantilla está lista) y, a continuación, en Template source (Origen de la plantilla), seleccione Amazon S3 URL (URL de Amazon S3).

e. Pegue la siguiente URL en el área de texto de URL de Amazon S3 y elija Siguiente:

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-vpc-sample.yaml
```

a. En la página Especificar detalles, rellene los parámetros y, luego, elija Siguiente.

- Nombre de pila: elija un nombre para su pila de AWS CloudFormation. Por ejemplo, puede llamarla *`amazon-eks-vpc-sample`*. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster.
- VpcBlock: elija un bloque de CIDR para la VPC. A cada nodo, pod y equilibrador de carga que implemente se le asigna una dirección `IPv4` de este bloque. Los valores predeterminados de `IPv4` proporcionan suficientes direcciones IP para la mayoría de las implementaciones, pero, si no lo hacen, puede cambiarlos. Para obtener más información, consulte Tamaño de la subred y VPC en la Guía del usuario de Amazon VPC. También puede agregar bloques de CIDR adicionales a la VPC una vez creada.
- Subnet01Block: especifique un bloque de CIDR para la subred 1. El valor predeterminado proporciona suficientes direcciones IP para la mayoría de las implementaciones, pero si no lo hace, puede cambiarlo.
- Subnet02Block: especifique un bloque de CIDR para la subred 2. El valor predeterminado proporciona suficientes direcciones IP para la mayoría de las implementaciones, pero si no lo hace, puede cambiarlo.
- Subnet03Block: especifique un bloque de CIDR para la subred 3. El valor predeterminado proporciona suficientes direcciones IP para la mayoría de las implementaciones, pero si no lo hace, puede cambiarlo.

b. (Opcional) En la página Options (Opciones), marque los recursos de la pila. Elija Siguiente.

c. En la página Review (Revisar), elija Create (Crear).

d. Una vez creada la pila, selecciónela en la consola y elija Outputs (Salidas).

e. Registre el VpcId de la VPC que ha creado. Necesita esto cuando al crear su clúster y nodos.

f. Anote el valor de SubnetIds de las subredes que se han creado. Necesita al menos dos de ellos al crear el clúster y los nodos.

g. (Opcional) Cualquier clúster que implemente en esta VPC puede asignar direcciones `IPv4` privadas a sus pods y servicios. Si desea implementar clústeres en esta VPC para asignar direcciones `IPv6` privadas a sus pods y servicios, debe actualizar la VPC, la subred, las tablas de enrutamiento y los grupos de seguridad. Para obtener más información, consulte Migrar VPC existentes de IPv4 a IPv6 en la Guía del usuario de Amazon VPC. Amazon EKS requiere que sus subredes tengan la opción de direcciones `Auto-assign IPv6` habilitada. De forma predeterminada, está deshabilitada.

Solo subredes privadas

Esta VPC tiene tres subredes privadas que se implementan en zonas de disponibilidad diferentes en una región de AWS. Los recursos implementados en las subredes no tienen acceso a Internet, ni se puede acceder a ellos desde Internet. La plantilla crea Puntos de conexión de VPC con AWS PrivateLink para varios servicios de AWS a los que normalmente deben acceder los nodos. Si los nodos necesitan acceso a Internet saliente, puede agregar una puerta de enlace NATpública en la zona de disponibilidad de cada subred después de crear la VPC. Se crea un grupo de seguridad que niega todo el tráfico entrante, excepto los recursos desplegados en las subredes. Un grupo de seguridad también permite todo el tráfico saliente. Las subredes están etiquetadas para que Kubernetes pueda implementar balanceadores de carga internos en ellas. Si va a crear una VPC con esta configuración, consulte the section called “Clústeres privados” para obtener requisitos y consideraciones adicionales.

a. Abra la Consola de AWS CloudFormation.

b. En la barra de navegación, seleccione una región de AWS compatible con Amazon EKS.

c. Elija Create stack (Crear pila), With new resources (standard) (Con nuevos recursos [estándar]).

d. En Prepare template (Preparar la plantilla), asegúrese de que se seleccione Template is ready (La plantilla está lista) y, a continuación, en Template source (Origen de la plantilla), seleccione Amazon S3 URL (URL de Amazon S3).

e. Pegue la siguiente URL en el área de texto de URL de Amazon S3 y elija Siguiente:

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-fully-private-vpc.yaml
```

a. En la página Especificar detalles, rellene los parámetros y, luego, elija Siguiente.
   - Nombre de pila: elija un nombre para su pila de AWS CloudFormation. Por ejemplo, puede llamarla *amazon-eks-fully-private-vpc*. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster.
   - VpcBlock: elija un bloque de CIDR para la VPC. A cada nodo, pod y equilibrador de carga que implemente se le asigna una dirección `IPv4` de este bloque. Los valores predeterminados de `IPv4` proporcionan suficientes direcciones IP para la mayoría de las implementaciones, pero, si no lo hacen, puede cambiarlos. Para obtener más información, consulte Tamaño de la subred y VPC en la Guía del usuario de Amazon VPC. También puede agregar bloques de CIDR adicionales a la VPC una vez creada.
   - PrivateSubnet01Block: especifique un bloque de CIDR para la subred 1. El valor predeterminado proporciona suficientes direcciones IP para la mayoría de las implementaciones, pero si no lo hace, puede cambiarlo.
   - PrivateSubnet02Block: especifique un bloque de CIDR para la subred 2. El valor predeterminado proporciona suficientes direcciones IP para la mayoría de las implementaciones, pero si no lo hace, puede cambiarlo.
   - PrivateSubnet03Block: especifique un bloque de CIDR para la subred 3. El valor predeterminado proporciona suficientes direcciones IP para la mayoría de las implementaciones, pero si no lo hace, puede cambiarlo.
b. (Opcional) En la página Options (Opciones), marque los recursos de la pila. Elija Siguiente.
c. En la página Review (Revisar), elija Create (Crear).
d. Una vez creada la pila, selecciónela en la consola y elija Outputs (Salidas).
e. Registre el VpcId de la VPC que ha creado. Necesita esto cuando al crear su clúster y nodos.
f. Anote el valor de SubnetIds de las subredes que se han creado. Necesita al menos dos de ellos al crear el clúster y los nodos.
g. (Opcional) Cualquier clúster que implemente en esta VPC puede asignar direcciones `IPv4` privadas a sus pods y servicios. Si desea implementar clústeres en esta VPC para asignar direcciones `IPv6` privadas a sus pods y servicios, debe actualizar la VPC, la subred, las tablas de enrutamiento y los grupos de seguridad. Para obtener más información, consulte Migrar VPC

existentes de IPv4 a IPv6 en la Guía del usuario de Amazon VPC. Amazon EKS requiere que sus subredes tengan la opción de direcciones `Auto-assign IPv6` habilitada (está desactivada de forma predeterminada).

Revisión de requisitos de grupos de seguridad de Amazon EKS para clústeres

En este tema se describen los requisitos del grupo de seguridad de un clúster de Amazon EKS.

Grupo de seguridad de clúster predeterminado

Al crear un clúster, Amazon EKS crea un grupo de seguridad denominado `eks-cluster-sg-`*`my-cluster-uniqueID`* . Este grupo de seguridad tiene las siguientes reglas de forma predeterminada:

Tipo de regla	Protocolo	Puertos	Origen	Destino
Entrada	Todos	Todos	Auto	
Salida	Todos	Todos		0.0.0.0/0(`IPv4`) or ::/0 (`IPv6`)
Salida	Todos	Todos		Propio (para tráfico de EFA)

El grupo de seguridad predeterminado incluye una regla de salida que permite el tráfico de Elastic Fabric Adapter (EFA) con el mismo grupo de seguridad como destino. Esto permite el tráfico de EFA dentro del clúster, lo que resulta beneficioso para las cargas de trabajo de IA y ML y la computación de alto rendimiento (HPC). Para obtener más información, consulte Elastic Fabric Adapter para cargas de trabajo de HPC e IA o ML en Amazon EC2 en la Guía del usuario de Amazon Elastic Compute Cloud.

> **Important**
>
> Si su clúster no necesita la regla de salida, puede eliminarla. Si la elimina, debe seguir teniendo las reglas mínimas enumeradas en Restricción del tráfico del clúster. Si elimina la regla de entrada, Amazon EKS la vuelve a crear cada vez que se actualice el clúster.

Amazon EKS agrega las siguientes etiquetas al grupo de seguridad. Si elimina las etiquetas, Amazon EKS las vuelve a agregar al grupo de seguridad cada vez que se actualice el clúster.

Clave	Valor
kubernetes.io/cluster/ *my-cluster*	owned
aws:eks:cluster-name	*my-cluster*
Name	eks-cluster-sg- *my-cluste r* -*uniqueid*

Amazon EKS asocia automáticamente este grupo de seguridad a los siguientes recursos que también crea:

- De 2 a 4 interfaces de red elásticas (denominadas para el resto de este documento como interfaz de red) que se crean al crear el clúster.
- Interfaces de redes de los nodos de cualquier grupo de nodos administrado que cree.

Las reglas predeterminadas permiten que todo el tráfico fluya libremente entre el clúster y los nodos, y permite que todo el tráfico saliente llegue a cualquier destino. Al crear un clúster, puede especificar (opcionalmente) sus propios grupos de seguridad. Si lo hace, Amazon EKS también asocia los grupos de seguridad especificados a las interfaces de red que crea para su clúster. Sin embargo, no los asocia a ningún grupo de nodos que cree.

Puede determinar el ID del grupo de seguridad de clúster en la Consola de administración de AWS bajo la sección Redes del clúster. O, puede hacerlo ejecutando el siguiente comando AWS CLI.

```
aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId
```

Restringir el tráfico del clúster

Si debe limitar los puertos abiertos entre el clúster y los nodos, puede eliminar la regla de salida predeterminada y agregar las siguientes reglas mínimas requeridas para el clúster. Si elimina la regla de entrada predeterminada, Amazon EKS la vuelve a crear cada vez que se actualice el clúster.

Tipo de regla	Protocolo	Puerto	Destino
Salida	TCP	443	Grupo de seguridad de clúster
Salida	TCP	10250	Grupo de seguridad de clúster
Saliente (DNS)	TCP y UDP	53	Grupo de seguridad de clúster

También debe agregar reglas para el siguiente tráfico:

- Cualquier protocolo que los puertos que espera que sus nodos usen para la comunicación entre nodos
- Acceso de salida de Internet para que los nodos puedan acceder a las API de Amazon EKS a fin de realizar la introspección del clúster y el registro de nodos en el momento del lanzamiento. Si los nodos no tienen acceso a Internet, revise Implementación de clústeres privados con acceso limitado a Internet para obtener consideraciones adicionales.
- Acceso a los nodos para extraer imágenes de contenedor de Amazon ECR u otras API de registros de contenedor del que necesiten extraer imágenes, como DockerHub. Para obtener más información, consulte Rangos de direcciones IP de AWS en la Referencia general de AWS.
- Acceso de nodo a Amazon S3.
- Se requieren reglas separadas para las direcciones `IPv4` y `IPv6`.
- Si utiliza nodos híbridos, debe agregar un grupo de seguridad adicional al clúster para permitir la comunicación con los nodos y pods en las instalaciones. Para obtener más información, consulte the section called “Preparación de las redes”.

Si se está planteando limitar las reglas, le recomendamos que pruebe detenidamente todos los pods antes de aplicar las reglas modificadas a un clúster de producción.

Si implementó originalmente un clúster con Kubernetes `1.14` y una versión de plataforma de `eks.3` o anterior, tenga en cuenta lo siguiente:

- Puede que también tenga grupos de seguridad de nodo y plano de control. Cuando se crearon estos grupos, incluyeron las reglas restringidas enumeradas en la tabla anterior. Estos grupos de seguridad ya no son necesarios y se pueden quitar. Sin embargo, debe asegurarse de que el grupo de seguridad del clúster contiene las reglas que contienen esos grupos.
- Si ha implementado el clúster utilizando la API directamente o ha utilizado una herramienta como AWS CLI o AWS CloudFormation para crear el clúster y no especificó un grupo de seguridad al crear el clúster, el grupo de seguridad predeterminado para la VPC se aplicó a las interfaces de red del clúster que creó Amazon EKS.

Grupos de seguridad compartidos

Amazon EKS admite grupos de seguridad compartidos.

- Las asociaciones de VPC de grupos de seguridad asocian grupos de seguridad a varias VPC de la misma cuenta y región.
  - Obtenga información sobre cómo Asociar grupos de seguridad a varias VPC en la Guía del usuario de Amazon VPC.
- Los grupos de seguridad compartidos permiten compartir grupos de seguridad con otras cuentas de AWS. Las cuentas deben estar en la misma organización de AWS.
  - Obtenga información sobre cómo Compartir grupos de seguridad con organizaciones en la Guía del usuario de Amazon VPC.
- Los grupos de seguridad se limitan siempre a una sola región de AWS.

Consideraciones para Amazon EKS

- EKS tiene los mismos requisitos de grupos de seguridad compartidos o de múltiples VPC que los grupos de seguridad estándar.

Administración de complementos de red para clústeres de Amazon EKS

Varios complementos de red están disponibles para el clúster de Amazon EKS.

Complementos incorporados

> **Note**
>
> Cuando crea un clúster de EKS:
>
> - Uso de la consola de AWS: los complementos integrados (como CoreDNS, kube-proxy, etc.) se instalan automáticamente como complementos de Amazon EKS. Pueden configurarse y actualizarse fácilmente a través de la consola de AWS, la CLI o los SDK.
> - Uso de otros métodos (CLI, SDK, etc.): los mismos complementos integrados se instalan como versiones autoadministradas que se ejecutan como implementaciones normales de Kubernetes. Requieren configuración y actualizaciones manuales, ya que no se pueden administrar mediante las herramientas de AWS.
>
> Recomendamos utilizar complementos de Amazon EKS en lugar de versiones autoadministradas para simplificar la administración de los complementos y permitir la configuración y las actualizaciones centralizadas a través de los servicios de AWS.

Complemento CNI de Amazon VPC para Kubernetes

Este CNI crea interfaces de red elástica y las adjunta a los nodos de Amazon EC2. El complemento también asigna una dirección `IPv4` o `IPv6` privada de la VPC a cada pod y servicio. De forma predeterminada, este complemento se instala en el clúster. Para obtener más información, consulte the section called “CNI de Amazon VPC”. Si utiliza nodos híbridos, el CNI de la VPC se mantiene instalado de forma predeterminada, pero se impide su ejecución en los nodos híbridos mediante una regla antiafinidad. Para obtener más información sobre las opciones de CNI para nodos híbridos, consulte the section called “Cómo configurar una CNI”.

CoreDNS

CoreDNS es un servidor de DNS flexible y extensible que puede servir como el DNS del clúster de Kubernetes. CoreDNS proporciona la resolución de nombres para todos los pods del clúster. De forma predeterminada, este complemento se instala en el clúster. Para obtener más información, consulte the section called “CoreDNS”.

`kube-proxy`

Este complemento mantiene las reglas de red en los nodos de Amazon EC2 y permite la comunicación de red con los pods. De forma predeterminada, este complemento se instala en el clúster. Para obtener más información, consulte the section called “kube-proxy”.

Complementos de red de AWS opcionales

AWS Controlador del equilibrador de carga de

Al implementar objetos de servicio de Kubernetes del tipo `loadbalancer`, el controlador crea equilibradores de carga de red de AWS. Al crear objetos de entrada de Kubernetes, el controlador crea equilibradores de carga de aplicaciones de AWS. Recomendamos usar este controlador para aprovisionar equilibradores de carga de red, en lugar de usar el controlador Proveedor de nube heredado integrado en Kubernetes. Para obtener más información, consulte la documentación del Controlador del equilibrador de carga de AWS.

AWS Controlador de API de la puerta de enlace de

Este controlador le permite conectar servicios en varios clústeres de Kubernetes con la API de la puerta de enlace de Kubernetes. El controlador conecta los servicios de Kubernetes que se ejecutan en instancias, contenedores y funciones sin servidor de Amazon EC2 con el servicio de Amazon VPC Lattice. Para obtener más información, consulte la documentación del controlador AWS de la API de puerta de enlace.

Para obtener más información sobre los complementos, consulte the section called “Complementos de Amazon EKS”.

Asignación de direcciones IP a pods con CNI de Amazon VPC

> **Tip**
>
> Con el modo automático de Amazon EKS, no necesita instalar ni actualizar los complementos de red. El modo automático ofrece capacidades para la conexión en red de pods y el equilibrio de carga.
> Para obtener más información, consulte Modo automático de EKS.

El complemento CNI de Amazon VPC para Kubernetes se implementa en cada nodo de Amazon EC2 de su clúster de Amazon EKS. El complemento crea interfaces de red elástica y las adjunta a los nodos de Amazon EC2. El complemento también asigna una dirección `IPv4` o `IPv6` privada de la VPC a cada pod.

Se implementa una versión del complemento con cada nodo de Fargate del clúster, pero no se actualiza en los nodos de Fargate. Hay otros complementos de CNI compatibles disponibles para su uso en los clústeres de Amazon EKS, pero este es el único complemento de CNI compatible con Amazon EKS para nodos que se ejecutan en la infraestructura de AWS. Para obtener más información acerca de otros complementos de CNI compatibles, consulte the section called "Complementos de CNI alternativos". No se admite el uso de la CNI de la VPC con nodos híbridos. Para obtener más información sobre las opciones de CNI para nodos híbridos, consulte the section called "Cómo configurar una CNI".

En la siguiente tabla se muestra la versión más reciente disponible del tipo de complemento de Amazon EKS para cada versión de Kubernetes.

Versiones de CNI de Amazon VPC

Versión de Kubernetes	Tipo de versión CNI de VPC de 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

Si administra este complemento, es posible que las versiones de la tabla no sean las mismas que las versiones autoadministradas disponibles. Para obtener más información acerca de la

actualización de complementos autoadministrados, consulte the section called “Actualización (autoadministrado)”.

> **Important**
>
> Para actualizar a VPC CNI v1.12.0 o superior, primero debe actualizar a VPC CNI v1.7.0. Le recomendamos que actualice una versión secundaria a la vez.

Consideraciones

A continuación, se detallan consideraciones que se deben tener a la hora de utilizar esta característica.

- Las versiones se especifican como `major-version.minor-version.patch-version-eksbuild.build-number`.
- Comprobar la compatibilidad de versiones para cada característica. Algunas características de cada versión del complemento CNI de Amazon VPC para Kubernetes requieren determinadas versiones de Kubernetes. Cuando se utilizan distintas características de Amazon EKS, si se requiere una versión específica del complemento, se indica en la documentación de características. A menos que tenga un motivo específico para ejecutar una versión anterior, le recomendamos elegir la versión más reciente.

Cómo crear la CNI de Amazon VPC (complemento de Amazon EKS)

Siga los siguientes pasos para crear el complemento CNI de Amazon VPC para Kubernetes en Amazon EKS.

Antes de empezar, revise las consideraciones. Para obtener más información, consulte the section called “Consideraciones”.

Requisitos previos

Los siguientes son requisitos previos para el complemento CNI de Amazon VPC para Kubernetes en Amazon EKS.

- Un clúster existente de Amazon EKS. Para implementar uno, consulte Introducción.

- Un proveedor existente de OpenID Connect (OIDC) de AWS Identity and Access Management (IAM) para su clúster. Para determinar si ya tiene un proveedor o para crear uno, consulte the section called “Proveedor de OIDC de IAM”.
- Un rol de IAM con la política de IAM AmazonEKS_CNI_Policy (si el clúster utiliza la familia `IPv4`) o una política de IPv6 (si el clúster utiliza la familia `IPv6`) adjuntas. Para obtener más información acerca de los roles de CNI de VPC, consulte the section called “Configuración para IRSA”. Para obtener más información acerca de la política IPv6, consulte the section called “Cree una política de IAM para clústeres que utilizan la familia `IPv6`”.

> **Important**
>
> Las versiones de la `v1.16.0` a la `v1.16.1` del complemento CNI de Amazon VPC para Kubernetes implementan la versión de especificación de CNI `v1.0.0`. Para obtener más información sobre la especificación de CNI `v1.0.0`, consulte Container Network Interface (CNI) Specification en GitHub.

Procedimiento

Después de completar los requisitos previos, siga los siguientes pasos para crear el complemento.

1. Consulte qué versión del complemento está instalada en el clúster.

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni: |
 cut -d : -f 3
```

   Un ejemplo de salida sería el siguiente.

```
v1.16.4-eksbuild.2
```

2. Consulte qué tipo del complemento está instalado en el clúster. Según la herramienta con la que haya creado el clúster, es posible que actualmente no tenga instalado el tipo de complemento Amazon EKS en el clúster. Reemplace *`my-cluster`* por el nombre de su clúster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

Si se devuelve el número de versión, tiene el tipo de complemento de Amazon EKS instalado en el clúster y no es necesario que complete los pasos restantes del procedimiento. Si se devuelve un error, no tiene el tipo de complemento de Amazon EKS instalado en el clúster. Complete los pasos restantes de este procedimiento para instalarlo.

3. Guarde la configuración del complemento instalado actualmente.

```
kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
```

4. Cree el complemento mediante la AWS CLI. Si desea utilizar Consola de administración de AWS o `eksctl` para crear el complemento, consultethe section called “Cómo crear un complemento” y especifique`vpc-cni` para el nombre del complemento. Copie el comando que sigue en su dispositivo. Realice las siguientes modificaciones en el comando según sea necesario y, a continuación, ejecute el comando modificado.
   - Reemplace *`my-cluster`* por el nombre de su clúster.
   - Sustituya *`v1.20.3-eksbuild.1`* por la versión más reciente que aparece en la tabla de versiones más recientes correspondiente a la versión del clúster. Para obtener la tabla de la última versión, consulte the section called “Versiones de CNI de Amazon VPC”.
   - Sustituya *`111122223333`* por el ID de la cuenta y *`AmazonEKSVPCCNIRole`* por el nombre del rol de IAM existente que creó. Para especificar un rol, es necesario disponer de un proveedor de OpenID Connect (OIDC) de IAM para el clúster. Para determinar si ya tiene uno para su clúster o si debe crearlo, consulte the section called “Proveedor de OIDC de IAM”.

```
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
```

   Si ha aplicado una configuración personalizada al complemento actual que entra en conflicto con la configuración predeterminada del complemento de Amazon EKS, es posible que se produzca un error en la creación. Si se produce un error en la creación, recibe un error que puede serle de utilidad para resolver el problema. Como alternativa, puede añadir `--resolve-conflicts OVERWRITE` al comando anterior. Esto permite que el complemento sobrescriba cualquier configuración personalizada existente. Una vez que haya creado el complemento, puede actualizarlo con la configuración personalizada.

5. Confirme que la versión más reciente del complemento de la versión de su clúster de Kubernetes se haya agregado al clúster. Reemplace *`my-cluster`* por el nombre de su clúster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

Es posible que la creación del complemento tarde varios segundos en completarse.

Un ejemplo de salida sería el siguiente.

```
v1.20.3-eksbuild.1
```

6. Si ha realizado ajustes personalizados en el complemento original, antes de crear el complemento de Amazon EKS, utilice la configuración que guardó en el paso anterior para actualizar el complemento de EKS con su configuración personalizada. Siga los pasos de the section called "Actualización (del complemento de EKS)".
7. (Opcional) Instale el `cni-metrics-helper` en su clúster. Extrae información de la interfaz de red elástica y la dirección IP, agrega métricas en todo el clúster y publica las métricas en Amazon CloudWatch. Para obtener más información, consulte cni-metrics-helper en GitHub.

Cómo actualizar la CNI de Amazon VPC (complemento de Amazon EKS)

Actualice el tipo de Amazon EKS del complemento CNI de Amazon VPC para Kubernetes. Si no ha agregado el tipo Amazon EKS del complemento a su clúster, puede instalarlo siguiendo los pasos en the section called "Creación". O bien, actualice el otro tipo de instalación de CNI de VPC con las instrucciones de the section called "Actualización (autoadministrado)".

1. Consulte qué versión del complemento está instalada en el clúster. Sustituya *my-cluster* por el nombre del clúster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 "addon.addonVersion" --output text
```

Un ejemplo de salida sería el siguiente.

```
v1.20.0-eksbuild.1
```

Compare la versión con la tabla de versiones más recientes que aparece en the section called "Versiones de CNI de Amazon VPC". Si la versión devuelta es la misma que la versión del clúster de Kubernetes en la tabla de versiones más recientes, significa que ya tiene la última versión

instalada en el clúster y no necesita completar el resto de este procedimiento. Si recibe un error, en lugar de un número de versión en la salida, significa que no tiene el tipo de versión de Amazon EKS en el clúster. Debe crear el complemento antes de poder actualizarlo mediante este procedimiento. Para crear el tipo de Amazon EKS del complemento CNI de VPC, puede seguir los pasos de the section called “Creación”.

2. Guarde la configuración del complemento instalado actualmente.

```
kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
```

3. Actualice el complemento con la AWS CLI. Si desea utilizar Consola de administración de AWS o `eksctl` para actualizar el complemento, consulte the section called “Cómo actualizar un complemento”. Copie el comando que sigue en su dispositivo. Realice las siguientes modificaciones en el comando según sea necesario y, a continuación, ejecute el comando modificado.
   - Reemplace *`my-cluster`* por el nombre de su clúster.
   - Sustituya *`v1.20.0-eksbuild.1`* por la versión más reciente que aparece en la tabla de versiones más recientes correspondiente a la versión del clúster.
   - Reemplace *`111122223333`* por el ID de su cuenta y *`AmazonEKSVPCCNIRole`* por el nombre del rol de IAM existente que creó. Para crear un rol de IAM para el CNI de la VPC, consulte the section called “Paso 1: creación del rol de IAM del complemento CNI de Amazon VPC para Kubernetes”. Para especificar un rol, es necesario disponer de un proveedor de OpenID Connect (OIDC) de IAM para el clúster. Para determinar si ya tiene uno para su clúster o si debe crearlo, consulte the section called “Proveedor de OIDC de IAM”.
   - La opción `--resolve-conflicts PRESERVE` conserva los valores de configuración existentes del complemento. Si ha establecido valores personalizados para la configuración del complemento y no utiliza esta opción, Amazon EKS sobrescribe los valores con los valores predeterminados. Si utiliza esta opción, le recomendamos que pruebe cualquier cambio de campo y valor en un clúster que no sea de producción antes de actualizar el complemento del clúster de producción. Si cambia este valor a `OVERWRITE`, todas las configuraciones cambiarán a los valores predeterminados de Amazon EKS. Si ha establecido valores personalizados para cualquier configuración, es posible que se sobrescriban con los valores predeterminados de Amazon EKS. Si cambia este valor a `none`, Amazon EKS no cambia el valor de ninguna configuración, pero la actualización podría fallar. Si se produce un error en la actualización, recibe un mensaje de error que lo ayuda a resolver el conflicto.
   - Si no va a actualizar un ajuste de configuración, elimine `--configuration-values '{`*`"env":{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}`*`}'`

del comando. Si va a actualizar una configuración, sustituya *"env": {"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}* por la configuración que desee establecer. En este ejemplo, la variable de entorno `AWS_VPC_K8S_CNI_EXTERNALSNAT` se establece en `true`. El valor que especifique debe ser válido para el esquema de configuración. Si no conoce el esquema de configuración, ejecute `aws eks describe-addon-configuration --addon-name vpc-cni --addon-version` *v1.20.0-eksbuild.1* , y reemplace *v1.20.0-eksbuild.1* por el número de versión del complemento cuya configuración desea ver. El esquema se devuelve en la salida. Si ya tiene alguna configuración personalizada, quiere eliminarla toda y volver a establecer los valores de toda la configuración en los valores predeterminados de Amazon EKS, elimine *"env": {"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}* del comando para que quede `{}` vacío. Para obtener una explicación de cada configuración, consulte las Variables de configuración de CNI en 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"}}'
```

La actualización puede tardar varios segundos en completarse.

4. Confirme que la versión del complemento se ha actualizado. Reemplace *my-cluster* por el nombre de su clúster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni
```

La actualización puede tardar varios segundos en completarse.

Un ejemplo de salida sería el siguiente.

```
{
    "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
\"}}"
    }
}
```

Cómo actualizar la CNI de Amazon VPC (complemento autoadministrado)

Important

Recomendamos agregar el tipo de complemento de Amazon EKS al clúster en lugar de utilizar el tipo de complemento autoadministrado. Si no está familiarizado con la diferencia entre los tipos, consulte the section called “Complementos de Amazon EKS”. Para obtener más información acerca de cómo agregar un complemento de Amazon EKS al clúster, consulte the section called “Cómo crear un complemento”. Si no puede usar el complemento de Amazon EKS, le recomendamos que envíe una pregunta sobre los motivos por los que no puede hacerlo al repositorio de GitHub de la hoja de ruta de contenedores.

1. Confirme que no tiene instalado en el clúster el tipo Amazon EKS del complemento autoadministrado. Reemplace *my-cluster* por el nombre de su clúster.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
    addon.addonVersion --output text
   ```

   Si se devuelve un mensaje de error, no tiene el tipo de complemento de Amazon EKS instalado en el clúster. Para autoadministrar el complemento, complete los pasos restantes de este procedimiento para actualizar el complemento. Si se devuelve el número de versión, tiene el tipo de complemento de Amazon EKS instalado en el clúster. Para actualizarlo, siga el procedimiento que aparece en the section called “Cómo actualizar un complemento”, en lugar de este procedimiento. Si no está familiarizado con las diferencias entre los tipos de complementos, consulte the section called “Complementos de Amazon EKS”.

2. Consulte qué versión de la imagen del contenedor está instalada actualmente en el clúster.

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni: |
 cut -d : -f 3
```

Un ejemplo de salida sería el siguiente.

```
v1.20.0-eksbuild.1
```

Es posible que su resultado no incluya el número de compilación.

3. Haga una copia de seguridad de la configuración actual para poder aplicar la misma configuración una vez que haya actualizado la versión.

```
kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
```

Consulte Releases en GitHub, para revisar las versiones disponibles y familiarizarse con los cambios efectuados en la versión a la que desea actualizar. Tenga en cuenta que le recomendamos que actualice a la misma versión `major.minor.patch` que aparece en la tabla de versiones más recientes disponibles, incluso si hay versiones posteriores disponibles en GitHub. Para obtener la tabla de la última versión disponible, consulte the section called “Versiones de CNI de Amazon VPC”. Las versiones de compilación que aparecen en la tabla no se especifican en las versiones autoadministradas que aparecen en GitHub. Actualice su versión al completar las tareas de una de las siguientes opciones:

- Si no tiene ninguna configuración personalizada para el complemento, ejecute el comando que aparece debajo del encabezado `To apply this release:` en GitHub para la versión a la que desea actualizar.
- Si tiene una configuración personalizada, descargue el archivo de manifiesto con el siguiente comando. Cambie *https://raw.githubusercontent.com/aws/amazon-vpc-cni-k8s/v1.20.0/config/master/aws-k8s-cni.yaml* por la URL de la versión de GitHub a la que está actualizando.

```
curl -O https://raw.githubusercontent.com/aws/amazon-vpc-cni-k8s/v1.20.3/config/
master/aws-k8s-cni.yaml
```

Si es necesario, modifique el archivo de manifiesto con la configuración personalizada de la copia de seguridad que hizo en un paso anterior y, a continuación, aplique el archivo modificado

al clúster. Si los nodos no tienen acceso a los repositorios privados de Amazon ECR de Amazon EKS de donde se extraen las imágenes (consulte las líneas que comienzan con `image:` en el manifiesto), tendrá que descargar las imágenes, copiarlas en su propio repositorio y modificar el manifiesto para que extraiga las imágenes de su repositorio. Para obtener más información, consulte the section called “Copiar una imagen en un repositorio”.

```
kubectl apply -f aws-k8s-cni.yaml
```

4. Confirme que la nueva versión ya esté instalada en el clúster.

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni: |
 cut -d : -f 3
```

Un ejemplo de salida sería el siguiente.

```
v1.20.3
```

5. (Opcional) Instale el `cni-metrics-helper` en su clúster. Extrae información de la interfaz de red elástica y la dirección IP, agrega métricas en todo el clúster y publica las métricas en Amazon CloudWatch. Para obtener más información, consulte cni-metrics-helper en GitHub.

Configuración del complemento de CNI de Amazon VPC para utilizar IRSA

El complemento CNI de Amazon VPC para Kubernetes es el complemento de red para redes de pod en clústeres de Amazon EKS. El complemento se encarga de asignar direcciones IP de la VPC a los pods de Kubernetes y establecer la configuración de red necesaria para los pods de cada nodo.

> **Note**
>
> El complemento CNI de Amazon VPC también es compatible con Amazon EKS Pod Identities. Para obtener más información, consulte the section called “Asignación del rol de IAM”.

El complemento:

- Requiere los permisos de AWS Identity and Access Management (IAM). Si el clúster utiliza la familia `IPv4`, los permisos se especifican en la política administrada de AWS

AmazonEKS_CNI_Policy. Si el clúster utiliza la familia IPv6, se deben agregar los permisos a una política de IAM que crea. Para obtener instrucciones, consulte the section called “Cree una política de IAM para clústeres que utilizan la familia IPv6”. Puede asociar la política al rol de IAM del nodo de Amazon EKS o a un rol de IAM independiente. Para obtener instrucciones sobre cómo asociar la política al rol de IAM del nodo de Amazon EKS, consulte the section called “Rol de IAM de nodo”. Le recomendamos que lo adjunte a un rol independiente, tal y como se detalla en este tema.

- Crea y está configurado para utilizar una cuenta de servicio de Kubernetes con el nombre aws-node cuando se implementa. La cuenta de servicio está vinculada a un clusterrole de Kubernetes denominado aws-node, al que se le asignan los permisos de Kubernetes necesarios.

> **Note**
>
> Los pods correspondientes al complemento CNI de Amazon VPC para Kubernetes tienen acceso a los permisos asignados al rol de IAM del nodo de Amazon EKS, a menos que se bloquee el acceso a IMDS. Para obtener más información, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.

- Requiere un clúster existente de Amazon EKS. Para implementar uno, consulte Introducción.
- Requiere un proveedor existente de OpenID Connect (OIDC) de AWS Identity and Access Management (IAM) para su clúster. Para determinar si ya tiene un proveedor o para crear uno, consulte the section called “Proveedor de OIDC de IAM”.

Paso 1: creación del rol de IAM del complemento CNI de Amazon VPC para Kubernetes

1. Determine la familia de IP del clúster.

```
aws eks describe-cluster --name my-cluster | grep ipFamily
```

Un ejemplo de salida sería el siguiente.

```
"ipFamily": "ipv4"
```

La salida puede devolver ipv6 en cambio.

2. Creación del rol de IAM. Puede utilizar eksctl o kubectl y la AWS CLI para crear el rol de IAM.

eksctl

- Cree un rol de IAM y adjunte la política de IAM al rol con el comando que coincida con la familia IP del clúster. Este comando crea e implementa una pila de AWS CloudFormation que crea un rol de IAM, adjunta la política que especifica para el rol y anota la cuenta de servicio de Kubernetes de `aws-node` existente con el ARN del rol de IAM que se crea.

  - `IPv4`

    Reemplace *`my-cluster`* con su propio valor.

    ```
    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`

    Reemplace *`my-cluster`* con su propio valor. Reemplace *`111122223333`* con su ID de cuenta y reemplace *`Amazoneks_CNI_IPv6_Policy`* con el nombre de su política `IPv6`. Si no dispone de una política `IPv6`, consulte the section called “Cree una política de IAM para clústeres que utilizan la familia `IPv6`” para crear uno. Para utilizar `IPv6` con su clúster, debe cumplir varios requisitos. Para obtener más información, consulte 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 y la AWS CLI

i. Visualización de la URL del proveedor de OIDC de su clúster.

```
aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text
```

Un ejemplo de salida sería el siguiente.

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

Si no se devuelve ninguna salida, debe crear un proveedor de OIDC de IAM para el clúster.

ii. Copie el siguiente contenido en un archivo con el nombre *vpc-cni-trust-policy.json*. Reemplace *111122223333* con su ID de cuenta y *EXAMPLED539D4633E53DE1B71EXAMPLE* con el resultado que obtuvo en el paso anterior. Reemplace *region-code* por la región de AWS en la que se encuentra el clúster.

```
{
    "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. Creación del rol. Puede reemplazar *AmazonEKSVPCCNIRole* por el nombre que elija.

```
aws iam create-role \
  --role-name AmazonEKSVPCCNIRole \
```

```
  --assume-role-policy-document file://"vpc-cni-trust-policy.json"
```

iv. Adjunte la política de IAM necesaria al rol de IAM Ejecute el comando que coincida con la familia IP del clúster.

- `IPv4`

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
  --role-name AmazonEKSVPCCNIRole
```

- `IPv6`

  Reemplace *`111122223333`* con su ID de cuenta y *`Amazoneks_CNI_IPv6_Policy`* con el nombre de su política `IPv6`. Si no dispone de una política `IPv6`, consulte the section called “Cree una política de IAM para clústeres que utilizan la familia `IPv6`” para crear uno. Para utilizar `IPv6` con su clúster, debe cumplir varios requisitos. Para obtener más información, consulte the section called “IPv6”.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy \
  --role-name AmazonEKSVPCCNIRole
```

v. Ejecute el siguiente comando para anotar la cuenta de servicio del `aws-node` con el ARN del rol de IAM que creó anteriormente. Sustituya los valores de ejemplo por sus propios valores.

```
kubectl annotate serviceaccount \
    -n kube-system aws-node \
    eks.amazonaws.com/role-arn=arn:aws:iam::111122223333:role/
AmazonEKSVPCCNIRole
```

3. (Opcional) Configure el tipo de punto de conexión de AWS Security Token Service que utiliza su cuenta de servicio de Kubernetes. Para obtener más información, consulte the section called “Puntos de conexión de STS”.

Paso 2: nueva implementación de pods del complemento CNI de Amazon VPC para Kubernetes

1. Elimine y vuelva a crear todos los pods existentes asociados a la cuenta de servicio para aplicar las variables de entorno de credenciales. La anotación no se aplica a los pods que se están

ejecutando actualmente sin la anotación. El siguiente comando elimina los pods de DaemonSet de `aws-node` existentes y los implementa con la anotación de cuenta de servicio.

```
kubectl delete Pods -n kube-system -l k8s-app=aws-node
```

2. Confirme que todos los pods se reiniciaron.

```
kubectl get pods -n kube-system -l k8s-app=aws-node
```

3. Describa uno de los pods y verifique que existan las variables de entorno `AWS_WEB_IDENTITY_TOKEN_FILE` y `AWS_ROLE_ARN`. Reemplace *cpjw7* por el nombre de uno de los pods que obtuvo en la salida del paso anterior.

```
kubectl describe pod -n kube-system aws-node-cpjw7 | grep 'AWS_ROLE_ARN:\|
AWS_WEB_IDENTITY_TOKEN_FILE:'
```

Un ejemplo de salida sería el siguiente.

```
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
```

Se devuelven dos conjuntos de resultados duplicados porque el pod contiene dos contenedores. Ambos contenedores tienen los mismos valores.

Si su pod está utilizando el punto de conexión regional de AWS, la siguiente línea también se devuelve en la salida anterior.

```
AWS_STS_REGIONAL_ENDPOINTS=regional
```

Paso 3: Eliminar la política de CNI del rol de IAM del nodo

Si actualmente el rol de IAM del nodo de Amazon EKS tiene asociada la política `AmazonEKS_CNI_Policy` de IAM (`IPv4`) o una política IPv6, y ha creado un rol de IAM independiente, le ha adjuntado la política y lo ha asignado a la cuenta de servicio de Kubernetes de

aws-node, le recomendamos eliminar la política del rol del nodo con el comando de la AWS CLI que coincida con la familia de IP del clúster. Reemplace *AmazonEKSNodeRole* con el nombre del rol de nodo.

- IPv4

```
aws iam detach-role-policy --role-name AmazonEKSNodeRole --policy-arn
 arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
```

- IPv6

  Reemplace *111122223333* con su ID de cuenta y *Amazoneks_CNI_IPv6_Policy* con el nombre de su política IPv6.

```
aws iam detach-role-policy --role-name AmazonEKSNodeRole --policy-arn
 arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy
```

Cree una política de IAM para clústeres que utilizan la familia **IPv6**

Si creó un clúster que utiliza la familia IPv6 y el clúster tiene configurada la versión 1.10.1 o posterior del complemento CNI de Amazon VPC para Kubernetes, debe crear una política de IAM que pueda asignar a un rol de IAM en un paso posterior. Si tiene un clúster existente que no configuró con la familia IPv6 cuando lo creó, deberá crear un clúster nuevo para poder utilizar IPv6. Para obtener más información acerca del uso de IPv6 con su clúster, consulte the section called “IPv6”.

1. Copie el siguiente texto y guárdelo en un archivo llamado 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. Cree la política de IAM.

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-document
 file://vpc-cni-ipv6-policy.json
```

Más información sobre los modos y la configuración de CNI de VPC

El complemento CNI de Amazon VPC para Kubernetes proporciona redes para pods. Utilice la siguiente tabla para obtener más información sobre las características de red disponibles.

Características de red	Más información
Configuración del clúster para asignar direcciones IPv6 a los clústeres, pods y servicios	the section called “IPv6”
Uso de la traducción de direcciones de red de origen IPv4 para pods	the section called “Tráfico de salida”
Restricción del tráfico de red de entrada y salida de pods	the section called “Restricción del tráfico”
Personalización de la interfaz de red secundaria en los nodos	the section called “Redes personalizadas”
Aumento de las direcciones IP para nodos	the section called “Aumento de las direcciones IP”

Características de red	Más información
Uso de grupos de seguridad para el tráfico de red del pod	the section called “Grupos de seguridad de pods”
Uso de varias interfaces de red para pods	the section called “Varias interfaces”

Información sobre la asignación de direcciones IPv6 a clústeres, pods y servicios

Aplicación: pods con instancias de Amazon EC2 y pods de Fargate

De forma predeterminada, Kubernetes asigna direcciones `IPv4` a sus pods y servicios. En lugar de asignar direcciones `IPv4` a sus pods y servicios, puede configurar el clúster para que les asigne direcciones `IPv6`. Amazon EKS no admite servicios o pods de doble pila, aunque Kubernetes sí lo hace. Como resultado, no puede asignar ambos tipos de direcciones, `IPv4` e `IPv6`, a sus pods y servicios.

Selecciona qué familia IP desea utilizar para su clúster cuando lo cree. Una vez creado el clúster, no se puede cambiar la familia.

Para ver un tutorial sobre cómo implementar un clúster `IPv6` de Amazon EKS, consulte the section called “Implementación”.

A continuación, se detallan consideraciones que se deben tener a la hora de utilizar esta característica:

Compatibilidad de características de **`IPv6`**

- Incompatibilidad con Windows: los pods y los servicios de Windows no son compatibles.
- Se requieren nodos EC2 basados en Nitro: solo puede utilizar `IPv6` con nodos de Amazon EC2 o Fargate de AWS basados en Nitro.
- Compatible con nodos EC2 y Fargate: se puede utilizar `IPv6` con the section called “Grupos de seguridad de pods” y nodos de Amazon EC2 y Fargate.
- Outposts no compatibles: no se puede utilizar `IPv6` con Amazon EKS en AWS Outposts.
- FSx for Lustre no es compatible: the section called “Amazon FSx para Lustre” no es compatible.
- Redes personalizadas no compatibles: si anteriormente utilizó the section called “Redes personalizadas” para ayudar a aliviar el agotamiento de la dirección IP, puede utilizar `IPv6` en

su lugar. No puede utilizar redes personalizadas con IPv6. Si utiliza redes personalizadas para el aislamiento de red, es posible que deba continuar utilizando redes personalizadas y la familia IPv4 para sus clústeres.

Asignaciones de dirección IP

- Servicios de Kubernetes: a los servicios de Kubernetes solo se les asigna una dirección IPv6. No se les asigna direcciones IPv4.
- Pods: a los pods se les asigna una dirección IPv6 y una dirección IPv4 local del host. La dirección IPv4 local del host se asigna mediante un complemento de la CNI local del host encadenado con la CNI de la VPC y la dirección no se informa al plano de control de Kubernetes. Solo se usa cuando un pod necesita comunicarse con un recurso IPv4 externo en otra Amazon VPC o en Internet. La dirección IPv4 local del host se asigna (mediante la CNI de la VPC) a la dirección IPv4 principal del ENI principal del nodo de trabajo.
- Pods y servicios: los pods y servicios reciben solo direcciones IPv6, no direcciones IPv4. Cuando los pods necesitan comunicarse con puntos de conexión IPv4 externos, utilizan la NAT en el propio nodo. Esta función NAT integrada elimina la necesidad de DNS64 y NAT64. Para el tráfico que requiere acceso a Internet público, el tráfico del pod se convierte mediante la dirección de red de origen a una dirección IP pública.
- Direcciones de enrutamiento: cuando un pod se comunica fuera de la VPC, se conserva su dirección IPv6 original (no se traduce a la dirección IPv6 del nodo). Este tráfico se enruta directamente a través de una puerta de enlace de Internet o una puerta de enlace de Internet de solo salida.
- Nodos: a todos los nodos se les asignan una dirección IPv4 e IPv6.
- Pods de Fargate: cada pod de Fargate recibe una dirección IPv6 de CIDR que se especifica para la subred en la que se implementa. La unidad de hardware subyacente que ejecuta pods de Fargate obtiene una dirección IPv4 e IPv6 única de los CIDR asignados a la subred en la que se implementa la unidad de hardware.

Cómo utilizar **IPv6** con EKS

- Crear un nuevo clúster: debe crear un nuevo clúster y especificar que desea utilizar la familia IPv6 para ese clúster. No puede habilitar la familia IPv6 para un clúster que haya actualizado desde una versión anterior. Para obtener instrucciones sobre cómo crear un clúster nuevo, consulte Consideraciones.

- Utilice la CNI de VPC reciente: implemente la versión `1.10.1` de la CNI de Amazon VPC o posterior. Esta versión o una posterior se implementa de forma predeterminada. Una vez que implemente el complemento, no podrá revertir la versión del complemento CNI de Amazon VPC a una versión inferior a `1.10.1` sin eliminar primero todos los nodos de todos los grupos de nodos del clúster.
- Configurar CNI de la VPC para **`IPv6`**: si utiliza nodos de Amazon EC2, debe configurar el complemento CNI de Amazon VPC con la delegación de prefijos IP e `IPv6`. Si elige la familia `IPv6` cuando cree el clúster, la versión `1.10.1` del complemento tendrá esta configuración como predeterminada. Este es el caso de un complemento autoadministrado o de Amazon EKS. Para obtener más información acerca de la delegación de prefijos IP, consulte the section called “Aumento de las direcciones IP”.
- Configurar las direcciones **`IPv4`** e **`IPv6`**: cuando crea un clúster, la VPC y las subredes que especifique deben tener un bloque de CIDR `IPv6` asignado a la VPC y las subredes que especifica. También deben tener un bloque de CIDR `IPv4` asignado. Esto se debe a que, incluso si solo desea utilizar `IPv6`, una VPC necesita un bloque de CIDR `IPv4` para funcionar. Para obtener más información, consulte Asociar un bloque de CIDR IPv6 a su VPC en la Guía del usuario de Amazon VPC.
- Asignar de forma automática direcciones IPv6 a nodos: cuando crea los nodos, debe especificar subredes que estén configuradas para asignar direcciones `IPv6` de forma automática. De lo contrario, no podrá implementar los nodos. Esta configuración está desactivada de forma predeterminada. Para obtener más información, consulte Modificar el atributo de direccionamiento IPv6 de su subred en la Guía del usuario de Amazon VPC.
- Establecer las tablas de enrutamiento para que utilicen **`IPv6`**: las tablas de enrutamiento que se asignen a las subredes deben tener rutas para direcciones `IPv6`. Para obtener más información, consulte Migración a IPv6 en la Guía del usuario de Amazon VPC.
- Establecer grupos de seguridad para **`IPv6`**: sus grupos de seguridad deben permitir direcciones `IPv6`. Para obtener más información, consulte Migración a IPv6 en la Guía del usuario de Amazon VPC.
- Configure el equilibrador de carga: utilice la versión `2.3.1` o posterior del Controlador del equilibrador de carga de AWS para equilibrar la carga de las aplicaciones HTTP mediante el the section called “Equilibrio de carga de aplicaciones” o el tráfico de red mediante el the section called “Equilibrio de carga de red” a pods `IPv6` a través de ambos equilibradores de carga en el modo IP, pero no en el modo de instancia. Para obtener más información, consulte the section called “ AWS Controlador del equilibrador de carga de ”.

- Añadir política de IAM de **IPv6**: debe adjuntar una política de IAM de IPv6 al IAM de su rol de IAM de CNI o nodo de IAM. Entre los dos, le recomendamos que lo adjunte a un rol de IAM de CNI. Para obtener más información, consulte the section called “Cree una política de IAM para clústeres que utilizan la familia IPv6” y the section called “Paso 1: creación del rol de IAM del complemento CNI de Amazon VPC para Kubernetes”.
- Evaluar todos los componentes: realice una evaluación exhaustiva de las aplicaciones, los complementos de Amazon EKS y los servicios de AWS con los que se integra antes de implementar clústeres IPv6. Esto es para garantizar que todo funcione según lo esperado con IPv6.

Implementación de un clúster **IPv6** de Amazon EKS y nodos administrados de Amazon Linux

En este tutorial, implementa una nube de Amazon VPC IPv6, un clúster de Amazon EKS con la familia IPv6 y un grupo de nodos administrado con nodos de Amazon Linux de Amazon EC2. No se pueden implementar nodos de Windows de Amazon EC2 en un clúster IPv6. También puede implementar nodos de Fargate en su clúster, aunque esas instrucciones no se proporcionarán en este tema por motivos de simplicidad.

Requisitos previos

Realice los siguientes pasos antes de comenzar el tutorial:

Instale y configure las siguientes herramientas y recursos que necesitará para crear y administrar un clúster de Amazon EKS.

- Recomendamos que conozca toda la configuración e implemente un clúster con la configuración que satisfaga sus requisitos. Para obtener más información, consulte the section called “Creación de un clúster ”, the section called “Grupos de nodos administrados” y las Consideraciones sobre este tema. Solo puede habilitar algunos ajustes de la configuración cuando cree su clúster.
- La herramienta de línea de comandos de kubectl está instalada en su dispositivo o AWS CloudShell. La versión puede ser la misma o hasta una versión secundaria anterior o posterior a la versión de Kubernetes de su clúster. Por ejemplo, si la versión del clúster es 1.29, puede usar la versión 1.28, 1.29 o 1.30 de kubectl con él. Para instalar o actualizar kubectl, consulte the section called “Configure kubectl y eksctl”.
- La entidad principal de seguridad de IAM que está utilizando debe contar con permisos para trabajar con los roles de IAM de Amazon EKS y los roles vinculados al servicio, AWS CloudFormation, además de una VPC y recursos relacionados. Para obtener más información, consulte Acciones y Uso de roles vinculados a servicios en la Guía del usuario de IAM.

- Si usa eksctl, instale la versión `0.215.0` o posterior en su equipo. Para instalar o actualizar esta versión, consulte la sección de Instalación en la documentación de `eksctl`.
- La versión `2.12.3` o posterior, o bien, la versión `1.27.160` o posterior de la AWS interfaz de la línea de comandos (AWS CLI) instalada y configurada en su dispositivo o AWS CloudShell. Para comprobar su versión actual, utilice `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente. Para actualizarla, consulte Instalación de la CLI de AWS en su directorio principal en la Guía del usuario de AWS CloudShell. Si utiliza AWS CloudShell, es posible que deba instalar la versión 2.12.3 o posterior, o bien, la versión 1.27.160 o posterior de la CLI de AWS, ya que la versión predeterminada de la CLI de AWS instalada en AWS CloudShell puede ser una versión anterior.

Puede utilizar eksctl o la CLI para implementar un clúster `IPv6`.

Implementación de un clúster IPv6 con eksctl

a. Creación del archivo `ipv6-cluster.yaml`. Copie el comando que sigue en su dispositivo. Realice las siguientes modificaciones en el comando según sea necesario y, a continuación, ejecute el comando modificado:
   - Reemplace *`my-cluster`* por el nombre de su clúster. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster.
   - Reemplace *`region-code`* por cualquier región de AWS en la que se admita Amazon EKS. Para ver una lista de las regiones de AWS, consulte Puntos de conexión y cuotas de Amazon EKS en la Guía de referencia general de AWS.
   - El valor de `version` con la versión de su clúster. Para obtener más información, consulte las versiones compatibles de Amazon EKS.
   - Reemplace *`my-nodegroup`* con un nombre para su grupo de nodos. El nombre del grupo de nodos no puede tener más de 63 caracteres. Debe empezar por una letra o un dígito, pero también puede incluir guiones y guiones bajos como caracteres no iniciales.
   - Reemplace *`t3.medium`* por cualquier tipo de instancia de 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. Creación de su clúster.

```
eksctl create cluster -f ipv6-cluster.yaml
```

La creación del clúster tarda varios minutos. No continúe hasta que vea la última línea de salida, que se parece a la siguiente salida.

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

c. Confirme que a los pods predeterminados se les asignan direcciones `IPv6`.

```
kubectl get pods -n kube-system -o wide
```

Un ejemplo de salida sería el siguiente.

```
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. Confirme que a los servicios predeterminados se les asignan direcciones `IPv6`.

```
kubectl get services -n kube-system -o wide
```

Un ejemplo de salida sería el siguiente.

```
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. (Opcional) Implemente una aplicación de muestra o implemente el Controlador del equilibrador de carga de AWS y una aplicación de muestra para equilibrar la carga de aplicaciones HTTP con

the section called “Equilibrio de carga de aplicaciones” o el tráfico de red con the section called “Equilibrio de carga de red” en pods de IPv6.

f. Cuando haya terminado con el clúster y los nodos que creó para este tutorial, debería limpiar los recursos que creó con el siguiente comando.

```
eksctl delete cluster my-cluster
```

Implementación de un clúster IPv6 con la CLI de AWS

> **Important**
>
> - Debe completar todos los pasos de este procedimiento como el mismo usuario. Ejecute el siguiente comando para comprobar el usuario actual:
>
> ```
> aws sts get-caller-identity
> ```
>
> - Debe completar todos los pasos de este procedimiento en el mismo intérprete de comandos. Varios pasos utilizan variables establecidas en pasos anteriores. Los pasos que utilizan variables no funcionarán de forma adecuada si los valores de las variables se establecen en un intérprete de comandos diferente. Si utiliza AWS CloudShell para completar el siguiente procedimiento, recuerde que, si no interactúa con él a través del teclado o el puntero durante 20 o 30 minutos aproximadamente, se cerrará la sesión del intérprete de comandos. Los procesos en ejecución no cuentan como interacciones.
> - Las instrucciones están escritas para el intérprete de comandos Bash y es posible que deban ajustarse para otros intérpretes de comandos.

Reemplace todos los valores de ejemplo en los pasos de este procedimiento por sus propios valores.

a. Ejecute los siguientes comandos para establecer algunas variables utilizadas en pasos posteriores. Reemplace *region-code* por la región de AWS en la que desea implementar sus recursos. El valor puede ser cualquier región de AWS que sea compatible con Amazon EKS. Para ver una lista de las regiones de AWS, consulte Puntos de conexión y cuotas de Amazon EKS en la Guía de referencia general de AWS. Reemplace *my-cluster* por el nombre de su clúster. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones. Debe comenzar con un carácter alfanumérico y no puede tener más de

100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster. Reemplace *my-nodegroup* con un nombre para su grupo de nodos. El nombre del grupo de nodos no puede tener más de 63 caracteres. Debe empezar por una letra o un dígito, pero también puede incluir guiones y guiones bajos como caracteres no iniciales. Reemplace *111122223333* por el ID de su cuenta.

```
export region_code=region-code
export cluster_name=my-cluster
export nodegroup_name=my-nodegroup
export account_id=111122223333
```

b. Cree una Amazon VPC con subredes privadas y públicas que cumplan con los requisitos de Amazon EKS y IPv6.

i. Ejecute el siguiente comando para establecer una variable para el nombre de su pila de AWS CloudFormation. Puede reemplazar *my-eks-ipv6-vpc* por el nombre que elija.

```
export vpc_stack_name=my-eks-ipv6-vpc
```

ii. Creación de una VPC IPv6 a partir de una plantilla de AWS CloudFormation.

```
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
```

La pila tarda unos minutos en crearse. Ejecute el siguiente comando. No continúe con el siguiente paso hasta que la salida del comando sea CREATE_COMPLETE.

```
aws cloudformation describe-stacks --region $region_code --stack-name
 $vpc_stack_name --query Stacks[].StackStatus --output text
```

iii. Recupere los ID de las subredes públicas que se crearon.

```
aws cloudformation describe-stacks --region $region_code --stack-name
 $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SubnetsPublic`].OutputValue' --output
 text
```

Un ejemplo de salida sería el siguiente.

```
subnet-0a1a56c486EXAMPLE,subnet-099e6ca77aEXAMPLE
```

iv. Habilite la opción de asignación automática de direcciones IPv6 para las subredes públicas que se crearon.

```
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. Recupere los nombres de las subredes y los grupos de seguridad creados mediante la plantilla desde la pila de AWS implementada y almacénelos en variables para utilizarlos en un paso posterior.

```
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. Cree un rol de IAM de clúster y adjúntelo a la política administrada de IAM de Amazon EKS. Los clústeres de Kubernetes administrados por Amazon EKS realizan llamadas a otros servicios de AWS en su nombre para administrar los recursos que utiliza con el servicio.

i. Ejecute el siguiente comando para crear un archivo `eks-cluster-role-trust-policy.json`.

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

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

ii. Ejecute el siguiente comando para establecer una variable para el nombre de su rol. Puede reemplazar *myAmazonEKSClusterRole* por cualquier nombre que elija.

```
export cluster_role_name=myAmazonEKSClusterRole
```

iii. Creación del rol.

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

iv. Recupere el ARN del rol de IAM y almacénelo en una variable para un paso posterior.

```
CLUSTER_IAM_ROLE=$(aws iam get-role --role-name $cluster_role_name --
query="Role.Arn" --output text)
```

v. Adjunte la política administrada de IAM por Amazon EKS requerida al rol.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name $cluster_role_name
```

d. Creación de su clúster.

```
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. NOTA: Es posible que reciba un error que indique que una de las zonas de disponibilidad de su solicitud no tiene capacidad suficiente para crear un clúster de Amazon EKS. Si esto ocurre, el mensaje de error indicará las zonas de disponibilidad que admiten un clúster nuevo. Intente crear el clúster de nuevo con al menos dos subredes ubicadas en las zonas de

disponibilidad admitidas para su cuenta. Para obtener más información, consulte the section called “Capacidad insuficiente”.

El clúster tarda varios minutos en crearse. Ejecute el siguiente comando. No continúe con el siguiente paso hasta que la salida del comando sea `ACTIVE`.

```
aws eks describe-cluster --region $region_code --name $cluster_name --query
 cluster.status
```

e. Creación o actualización de un archivo `kubeconfig` para su clúster para que pueda comunicarse con él.

```
aws eks update-kubeconfig --region $region_code --name $cluster_name
```

De forma predeterminada, el archivo de `config` se crea en `~/.kube` o la configuración del clúster nuevo se agrega a un archivo de `config` existente en `~/.kube`.

f. Creación de un rol de IAM de nodo.

i. Ejecute el siguiente comando para crear un archivo `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/*"
            ]
```

```
        }
    ]
}
```

ii. Creación de la política de IAM.

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-document
 file://vpc-cni-ipv6-policy.json
```

iii. Ejecute el siguiente comando para crear un archivo `node-role-trust-relationship.json`.

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

iv. Ejecute el siguiente comando para establecer una variable para el nombre de su rol. Puede reemplazar *`AmazonEKSNodeRole`* por el nombre que elija.

```
export node_role_name=AmazonEKSNodeRole
```

v. Creación del rol de IAM.

```
aws iam create-role --role-name $node_role_name --assume-role-policy-document
 file://"node-role-trust-relationship.json"
```

vi. Adjunte la política de IAM al rol de IAM.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::$account_id:policy/
AmazonEKS_CNI_IPv6_Policy \
    --role-name $node_role_name
```

> **Important**
>
> Para simplificar este tutorial, la política se adjunta a este rol de IAM. Sin embargo, en un clúster de producción, recomendamos adjuntar la política a un rol de IAM independiente. Para obtener más información, consulte the section called "Configuración para IRSA".

vii. Adjunte al rol de IAM las dos políticas administradas por IAM necesarias.

```
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. Recupere el ARN del rol de IAM y almacénelo en una variable para un paso posterior.

```
node_iam_role=$(aws iam get-role --role-name $node_role_name --query="Role.Arn" --
output text)
```

g. Creación de un grupo de nodos administrados.

i. Visualización de los ID de las subredes que creó en un paso anterior.

```
echo $subnets
```

Un ejemplo de salida sería el siguiente.

```
subnet-0a1a56c486EXAMPLE,subnet-099e6ca77aEXAMPLE,subnet-0377963d69EXAMPLE,subnet-0c05f819
```

ii. Creación de grupo de nodos. Sustituya *0a1a56c486EXAMPLE*, *099e6ca77aEXAMPLE*, *0377963d69EXAMPLE* y *0c05f819d5EXAMPLE* por los valores devueltos en la salida del paso anterior. Asegúrese de eliminar las comas entre los ID de subredes de la salida anterior en el siguiente comando. Puede reemplazar *t3.medium* por cualquier tipo de instancia de AWS Nitro System.

```
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
```

El grupo de nodos tarda unos minutos en crearse. Ejecute el siguiente comando. No continúe con el siguiente paso hasta que la salida devuelta sea `ACTIVE`.

```
aws eks describe-nodegroup --region $region_code --cluster-name $cluster_name --
nodegroup-name $nodegroup_name \
    --query nodegroup.status --output text
```

h. Confirme que a los pods predeterminados se les asignen direcciones `IPv6` en la columna `IP`.

```
kubectl get pods -n kube-system -o wide
```

Un ejemplo de salida sería el siguiente.

```
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. Confirme que a los servicios predeterminados se les asignen direcciones `IPv6` en la columna `IP`.

```
kubectl get services -n kube-system -o wide
```

Un ejemplo de salida sería el siguiente.

```
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. (Opcional) Implemente una aplicación de muestra o implemente el Controlador del equilibrador de carga de AWS y una aplicación de muestra para equilibrar la carga de aplicaciones HTTP con the section called “Equilibrio de carga de aplicaciones” o el tráfico de red con the section called “Equilibrio de carga de red” en pods de IPv6.

k. Cuando haya terminado con el clúster y los nodos que creó para este tutorial, debería limpiar los recursos que creó con los siguientes comandos. Asegúrese de no estar utilizando ninguno de los recursos fuera de este tutorial antes de eliminarlos.

   i. Si realiza este paso en un intérprete de comandos diferente al que utilizó para completar los pasos anteriores, establezca los valores de todas las variables utilizadas previamente y reemplace los valores de ejemplo por los que especificó en los pasos anteriores. Si está completando este paso en el mismo intérprete de comandos en el que completó los pasos anteriores, vaya al siguiente paso.

```
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. Eliminación de su grupo de nodos.

```
aws eks delete-nodegroup --region $region_code --cluster-name $cluster_name --
nodegroup-name $nodegroup_name
```

La eliminación tarda unos minutos. Ejecute el siguiente comando. No continúe con el siguiente paso si se devuelve alguna salida.

```
aws eks list-nodegroups --region $region_code --cluster-name $cluster_name --query
 nodegroups --output text
```

iii. Elimine el clúster.

```
aws eks delete-cluster --region $region_code --name $cluster_name
```

El clúster tarda unos minutos en eliminarse. Antes de continuar, asegúrese de que el clúster se elimine con el siguiente comando.

```
aws eks describe-cluster --region $region_code --name $cluster_name
```

No continúe con el siguiente paso hasta que la salida sea similar a la siguiente.

```
An error occurred (ResourceNotFoundException) when calling the DescribeCluster
 operation: No cluster found for name: my-cluster.
```

iv. Eliminación de los resultados de IAM que creó. Reemplace *AmazonEKS_CNI_IPv6_Policy* por el nombre que eligió si eligió un nombre diferente al que utilizó en los pasos anteriores.

```
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. Eliminación de la pila de AWS CloudFormation que creó la VPC.

```
aws cloudformation delete-stack --region $region_code --stack-name $vpc_stack_name
```

Habilitación del acceso a Internet saliente para pods

Aplicación: en nodos de `IPv4` de Linux de Fargate, y nodos de Linux con instancias de Amazon EC2

Si ha implementado el clúster mediante la familia `IPv6`, la información de este tema no se aplica a su clúster, porque las direcciones `IPv6` no están traducidas en red. Para obtener más información acerca del uso de `IPv6` con su clúster, consulte the section called “IPv6”.

De forma predeterminada, a cada pod del clúster se le asigna una dirección `IPv4` privada de un bloque de enrutamiento entre dominios sin clases (CIDR) asociado a la VPC en la que se implementa el pod. Los pods en la misma VPC se comunican entre sí utilizando estas direcciones IP privadas como puntos de conexión. Cuando un pod se comunica con cualquier dirección `IPv4` que no se encuentra dentro de un bloque de CIDR asociado a la VPC, el complemento CNI de Amazon VPC (para Linux o Windows) traduce la dirección `IPv4` del pod a la dirección `IPv4` privada principal de la interfaz de red elástica principal del nodo en el que se está ejecutando el pod, que es * de forma predeterminada.

> **Note**
>
> En el caso de los nodos de Windows, hay detalles adicionales que se deben tener en cuenta. De forma predeterminada, el complemento CNI de Amazon VPC para Windows se define con una configuración de red en la que se excluye el tráfico a un destino dentro de la misma VPC para SNAT. Esto significa que la comunicación interna de la VPC tiene la SNAT desactivada y la dirección IP asignada a un pod se puede enrutar dentro de la VPC. Sin embargo, el tráfico a un destino fuera de la VPC tiene la IP del pod de origen vinculada mediante la SNAT a la dirección IP principal de la ENI de la instancia. Esta configuración predeterminada de Windows garantiza que el pod pueda acceder a redes fuera de la VPC de la misma manera que a la instancia del host.

Debido a este comportamiento:

- Los pods pueden comunicarse con recursos de Internet solo si el nodo en el que se están ejecutando tiene una dirección IP pública o elástica asignada y se encuentra en una subred pública. Una subred pública es una subred asociada a la tabla de enrutamiento que tiene una ruta a la puerta de enlace de internet. Recomendamos implementar nodos en subredes privadas, siempre que sea posible.
- Para versiones del complemento anteriores a `1.8.0`, los recursos que se encuentran en redes o VPC que están conectadas a la VPC del clúster mediante un emparejamiento de VPC, una

VPC de tránsito o AWS Direct Connect no pueden iniciar la comunicación con sus pods detrás de interfaces de redes elásticas secundarias. Sin embargo, sus pods pueden iniciar la comunicación con esos recursos y recibir respuestas de ellos.

Si alguna de las siguientes afirmaciones es verdadera en su entorno, cambie la configuración predeterminada con el comando siguiente.

- Tiene recursos en redes o VPC que están conectados a la VPC de su clúster mediante emparejamiento de VPC, una VPC de tránsito o AWS Direct Connect que deben iniciar la comunicación con sus pods mediante una dirección `IPv4` y la versión del complemento es anterior a la `1.8.0`.
- Sus pods se encuentran en una subred privada y necesitan comunicación saliente a Internet. La subred tiene una ruta hacia una puerta de enlace NAT.

```
kubectl set env daemonset -n kube-system aws-node AWS_VPC_K8S_CNI_EXTERNALSNAT=true
```

> **Note**
>
> Las variables de configuración de CNI `AWS_VPC_K8S_CNI_EXTERNALSNAT` y `AWS_VPC_K8S_CNI_EXCLUDE_SNAT_CIDRS` no se aplican a los nodos de Windows. No se admite la desactivación de SNAT para Windows. En cuanto a la exclusión de una lista de CIDR `IPv4` de SNAT, puede especificar el parámetro `ExcludedSnatCIDRs` en el script de arranque de Windows para definirla. Para obtener más información acerca del uso de este parámetro, consulte the section called “Parámetros de configuración del script de arranque”.

Red del host

* Si las especificaciones del pod contienen `hostNetwork=true` (el valor predeterminado es `false`), su dirección IP no se traduce a otra dirección. Este es el caso de `kube-proxy` y el complemento CNI de Amazon VPC para Kubernetes, que se ejecutan en el clúster, de forma predeterminada. Para estos pods, la dirección IP es la misma que la dirección IP principal del nodo, por lo que la dirección IP del pod no está traducida. Para obtener más información acerca de la configuración de `hostNetwork` del pod, consulte PodSpec v1 core en la referencia de API de Kubernetes.

Limitación del tráfico de un pod con políticas de red de Kubernetes

Descripción general

De forma predeterminada, no hay restricciones en Kubernetes para las direcciones IP, puertos o conexiones entre cualquier pod de su clúster o entre sus pods y recursos en cualquier otra red. Puede usar una política de red de Kubernetes para restringir el tráfico de red que entra y sale de sus pods. Para obtener más información, consulte Network Policies en la documentación de Kubernetes.

Política de red estándar

Puede usar la `NetworkPolicy` estándar para segmentar el tráfico de pod a pod en el clúster. Estas políticas de red permiten operar en las capas 3 y 4 del modelo de red OSI, lo que le permite controlar el flujo de tráfico en la dirección IP o el puerto dentro del clúster de Amazon EKS. El ámbito de las políticas de red estándar es el espacio de nombres.

Casos de uso

- Segmente el tráfico de red entre las cargas de trabajo para garantizar que solo las aplicaciones relacionadas puedan comunicarse entre sí.
- Aísle los inquilinos del espacio de nombres mediante políticas que impongan la separación de la red.

Ejemplo

En la política siguiente, se restringe el tráfico de salida de los pods webapp en el espacio de nombres sun.

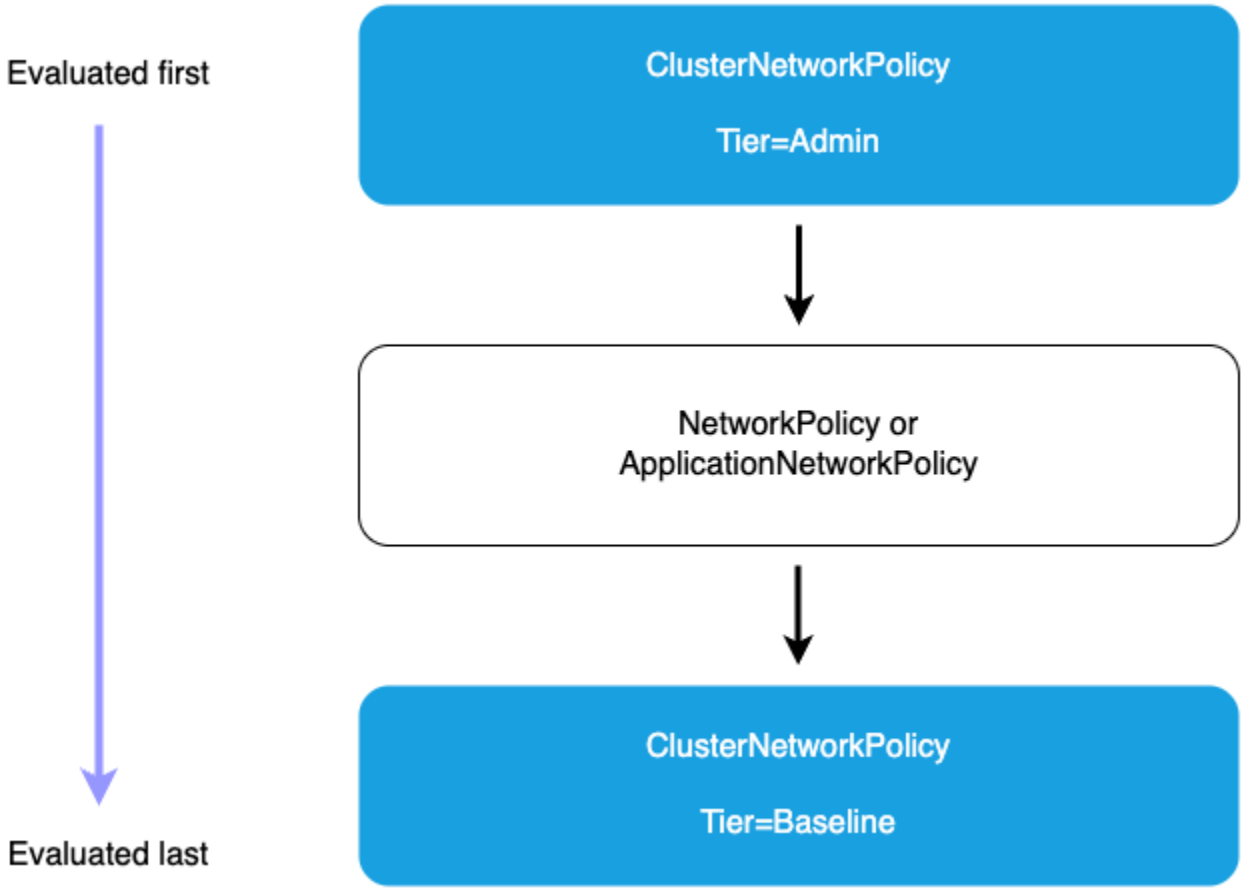
```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: webapp-egress-policy
  namespace: sun
spec:
  podSelector:
    matchLabels:
      role: webapp
  policyTypes:
  - Egress
  egress:
```

```
  - 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
```

La política se aplica a los pods con la etiqueta `role: webapp` en el espacio de nombres `sun`.

- Tráfico permitido: pods con la etiqueta `role: frontend` en el espacio de nombres `moon` del puerto TCP `8080`
- Tráfico permitido: pods con la etiqueta de rol de frontend en el espacio de nombres `stars` del puerto TCP `8080`
- Tráfico bloqueado: se deniega implícitamente el resto del tráfico saliente de los pods `webapp`

Política de red de administración (o del clúster)



Puede usar la `ClusterNetworkPolicy` para aplicar un estándar de seguridad de red a todo el clúster. En lugar de definir y mantener de forma repetitiva una política distinta para cada espacio de nombres, puede usar una política única para administrar de forma centralizada los controles de acceso a la red para las diferentes cargas de trabajo del clúster, independientemente de su espacio de nombres.

Casos de uso

- Administre de forma centralizada los controles de acceso a la red para todas las cargas de trabajo (o un subconjunto de ellas) del clúster de EKS.
- Defina una posición de seguridad de red predeterminada en todo el clúster.
- Amplíe los estándares de seguridad de la organización al ámbito del clúster de una manera más eficiente desde el punto de vista operativo.

Ejemplo

En la política siguiente, puede bloquear de forma explícita el tráfico del clúster desde otros espacios de nombres para impedir el acceso de la red al espacio de nombres confidencial de la carga de trabajo.

```
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
```

Notas importantes

Las políticas de red del complemento CNI de Amazon VPC para Kubernetes se admiten en las configuraciones que se enumeran a continuación.

- Versión 1.21.0 (o posterior) del complemento CNI de Amazon VPC para políticas de red estándar y de administración.
- Clúster configurado para las direcciones `IPv4` o `IPv6`.
- Puede utilizar las políticas de red con los grupos de seguridad para pods. Con las políticas de red, puede controlar todas las comunicaciones dentro del clúster. Con los grupos de seguridad para pods, puede controlar el acceso a servicios de AWS desde aplicaciones dentro de un pod.
- Puede utilizar las políticas de red con las redes personalizadas y la delegación de prefijos.

Consideraciones

Arquitectura

- Cuando se aplican las políticas de red del complemento CNI de Amazon VPC para Kubernetes a su clúster con el complemento CNI de Amazon VPC para Kubernetes, se pueden aplicar las políticas únicamente a los nodos de Linux de Amazon EC2. No se pueden aplicar las políticas a los nodos de Fargate o Windows.
- Las políticas de red solo aplican direcciones `IPv4` o `IPv6`, pero no ambas. En un clúster `IPv4`, la CNI de VPC asigna direcciones `IPv4` a los pods y aplica políticas `IPv4`. En un clúster `IPv6`, la CNI de VPC asigna direcciones `IPv6` a los pods y aplica políticas `IPv6`. Se ignora cualquier regla de política de red `IPv4` aplicada a un clúster `IPv6`. Se ignora cualquier regla de política de red `IPv6` aplicada a un clúster `IPv4`.

Políticas de red

- Las políticas de red solo se aplican a los pods que forman parte de una implementación. No se pueden aplicar políticas de red a los pods independientes que no tengan un conjunto de `metadata.ownerReferences`.
- Puede aplicar varias políticas de red al mismo pod. Cuando se han configurado dos o más políticas que seleccionan el mismo pod, todas las políticas se aplican al pod.
- La cantidad máxima de combinaciones de puertos y protocolos para un único rango de direcciones IP (CIDR) es de 24 en todas las políticas de red. Los selectores, como `namespaceSelector`, se resuelven en uno o más CIDR. Si varios selectores se resuelven en un único CIDR o si especifica el mismo CIDR directo varias veces en la misma o en diferentes políticas de red, todos estos cuentan para este límite.
- Para cualquiera de sus servicios de Kubernetes, el puerto de servicio debe ser el mismo que el puerto del contenedor. Si utiliza puertos con nombre, utilice también el mismo nombre en la especificación del servicio.

Políticas de red de administración

1. Políticas del nivel de administración (se evalúan primero): todas las ClusterNetworkPolicies del nivel de administración se evalúan antes que cualquier otra política. En el nivel de administración, las políticas se procesan por orden de prioridad (se comienza por el número de prioridad más baja). El tipo de acción determina lo que sucede a continuación.
   - Acción de denegación (máxima prioridad): cuando una política de administración con una acción de denegación coincide con el tráfico, ese tráfico se bloquea inmediatamente, independientemente de cualquier otra política. No se procesan más reglas de

ClusterNetworkPolicy ni NetworkPolicy. Esto garantiza que las políticas del espacio de nombres no puedan anular los controles de seguridad de toda la organización.

- Acción de permiso: una vez evaluadas las reglas de denegación, las políticas de administración que incluyen acciones de permiso se procesan por orden de prioridad (se comienza por el número de prioridad más baja). Cuando una acción de permiso coincide, se acepta el tráfico y no se lleva a cabo ninguna otra evaluación de la política. Estas políticas pueden conceder el acceso a varios espacios de nombres en función de los selectores de etiquetas, lo que proporciona un control centralizado sobre qué cargas de trabajo pueden acceder a recursos específicos.
- Acción de transferencia: las acciones de transferencia en las políticas del nivel de administración delegan la toma de decisiones a los niveles inferiores. Cuando el tráfico coincide con una regla de transferencia, la evaluación omite todas las reglas del nivel de administración restantes para ese tráfico y continúa directamente con el nivel de la NetworkPolicy. Esto permite a los administradores delegar explícitamente el control de determinados patrones de tráfico a los equipos de aplicaciones. Por ejemplo, puede usar reglas de transferencia para delegar la administración del tráfico dentro del espacio de nombres a los administradores del espacio de nombres y, al mismo tiempo, mantener controles estrictos sobre el acceso externo.

2. Nivel de política de red: si ninguna política de administración coincide con ninguna acción de denegación o permiso, o si coincide con una acción de transferencia, se evalúan a continuación los recursos de la NetworkPolicy del ámbito del espacio de nombres. Estas políticas proporcionan un control detallado dentro de los espacios de nombres individuales y las administran los equipos de aplicaciones. Las políticas relacionadas con el espacio de nombres solo pueden ser más restrictivas que las políticas de administración. No pueden anular ninguna decisión de denegación de una política de administración, pero pueden restringir aún más el tráfico permitido o transferido por las políticas de administración.
3. Políticas de administración del nivel de línea de base: si ninguna política de administración o del ámbito del espacio de nombres coincide con el tráfico, se evalúan las ClusterNetworkPolicies del nivel de línea de base. Proporcionan posiciones de seguridad predeterminadas que pueden ser anuladas por políticas del ámbito del espacio de nombres, lo que permite a los administradores establecer valores predeterminados para toda la organización y, al mismo tiempo, ofrecer a los equipos la flexibilidad necesaria para personalizarlas según sea necesario. Las políticas de referencia se evalúan por orden de prioridad (se comienza por el número de prioridad más baja).
4. Denegación de forma predeterminada (si ninguna política coincide): este comportamiento de denegación de forma predeterminada garantiza que solo se permitan las conexiones explícitamente permitidas, lo que mantiene una posición de seguridad sólida.

Migración

- Si su clúster utiliza actualmente una solución de terceros para la administración de políticas de red de Kubernetes, puede usar esas mismas políticas con el complemento CNI de Amazon VPC para Kubernetes. Sin embargo, debe eliminar la solución existente para que no administre las mismas políticas.

> **Warning**
>
> Se recomienda que, después de eliminar una solución de política de red, reemplace todos los nodos a los que se haya aplicado la solución de política de red. Esto se debe a que las reglas de tráfico podrían quedar desfasadas si un pod de la solución se cierra repentinamente.

Instalación

- La característica de política de red crea y requiere una definición de recurso personalizada (CRD) de `PolicyEndpoint` llamada `policyendpoints.networking.k8s.aws`. Amazon EKS administra los objetos `PolicyEndpoint` del recurso personalizado. No se deben modificar ni eliminar estos recursos.
- Si ejecuta pods que utilizan las credenciales de IAM del rol de instancia o se conectan al IMDS de EC2, compruebe si hay políticas de red que puedan bloquear el acceso al IMDS de EC2. Puede que tenga que añadir una política de red para permitir el acceso al IMDS de EC2. Para obtener más información, consulte Metadatos de instancia y datos de usuario en la guía del usuario de Amazon EC2.

  Los pods que utilizan los roles de IAM para cuentas de servicio o Pod Identity de EKS no acceden al IMDS de EC2.
- El complemento CNI de Amazon VPC para Kubernetes no aplica políticas de red a las interfaces de red adicionales de cada pod, solo a la interfaz principal de cada pod (`eth0`). Esta característica afecta a las siguientes arquitecturas:
  - Pods `IPv6` con la variable `ENABLE_V4_EGRESS` establecida en `true`. Esta variable habilita la característica de salida `IPv4` para conectar los pods IPv6 a puntos de conexión `IPv4`, como aquellos fuera del clúster. La característica de salida `IPv4` funciona mediante la creación de una interfaz de red adicional con una dirección IPv4 de bucle invertido local.

- Cuando se utilizan complementos de red encadenados, como Multus. Debido a que estos complementos añaden interfaces de red a cada pod, las políticas de red no se aplican a los complementos de red encadenados.

Restricción del tráfico de red de pods con políticas de red de Kubernetes

Puede usar la política de red de Kubernetes para restringir el tráfico de red que entra y sale de pods. Para obtener más información, consulte Network Policies en la documentación de Kubernetes.

Debe configurar lo siguiente para poder utilizar esta característica:

1. Configure la aplicación de políticas al inicio del pod. Esto se hace en el de contenedor `aws-node` del `DaemonSet` de la CNI de la VPC.
2. Habilite el parámetro de política de red para el complemento.
3. Configuración del clúster para que use la política de red de Kubernetes

Antes de empezar, revise las consideraciones. Para obtener más información, consulte the section called “Consideraciones”.

Requisitos previos

A continuación se indican los requisitos previos de esta característica:

Versión mínima del clúster

Un clúster existente de Amazon EKS. Para implementar uno, consulte Introducción. El clúster debe estar ejecutando una de las versiones de Kubernetes y versiones de la plataforma que se enumeran en la tabla siguiente. Tenga en cuenta que también se admite cualquier versión de Kubernetes y de la plataforma posterior a las enumeradas. Para comprobar la versión actual de Kubernetes, reemplace *my-cluster* en el siguiente comando por el nombre del clúster y, a continuación, ejecute el comando modificado:

```
aws eks describe-cluster --name my-cluster --query cluster.version --output text
```

Versión de Kubernetes	Versión de la plataforma
1.27.4	eks.5

Versión de Kubernetes	Versión de la plataforma
1.26.7	eks.6

Versión mínima de CNI de VPC

Para crear políticas de red de administración y políticas de red de Kubernetes estándar, debe ejecutar la versión 1.21 del complemento CNI de VPC. Puede comprobar qué versión utiliza actualmente con el siguiente comando.

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni: |
 cut -d : -f 3
```

Si su versión es anterior a la 1.21, actualice the section called “Actualización (del complemento de EKS)” a la versión 1.21 o posterior.

Versión mínima del kernel de Linux

Sus nodos deben tener la versión del núcleo de Linux 5.10 o posterior. Puede comprobar la versión del núcleo con uname -r. Si utiliza las versiones más recientes de las AMI optimizadas para Amazon EKS de Amazon Linux, las AMI optimizadas para Amazon EKS de Amazon Linux acelerado y las AMI de Bottlerocket, estas ya tienen la versión de kernel necesaria.

La versión v20231116 de la AMI de Amazon Linux acelerada y optimizada de Amazon EKS o posterior tiene una versión 5.10 de núcleo.

Paso 1: configuración de la aplicación de políticas al inicio del pod

El complemento CNI de Amazon VPC para Kubernetes configura las políticas de red para los pods en paralelo con el aprovisionamiento de los pods. A menos que se hayan configurado todas las políticas para el nuevo pod, los contenedores del nuevo pod se iniciarán con una política de permisos predeterminada. Esto se denomina modo estándar. Una política de permisos predeterminada significa que se permite todo el tráfico de entrada y salida desde y hacia los nuevos pods. Por ejemplo, no se aplicará ninguna regla de firewall a los pods (se permite todo el tráfico) hasta que el nuevo pod se actualice con las políticas activas.

Con la variable NETWORK_POLICY_ENFORCING_MODE configurada en strict, los pods que utilizan la CNI de la VPC se inician con una política de denegación predeterminada y, a continuación, se

configuran las políticas. Esto se denomina modo estricto. En el modo estricto, debe tener una política de red para cada punto de conexión del clúster al que los pods necesiten acceder. Tenga en cuenta que este requisito se aplica a los pods de CoreDNS. La política de denegación predeterminada no está configurada para los pods con redes de Host.

Para cambiar la política de red predeterminada, debe configurar la variable de entorno `NETWORK_POLICY_ENFORCING_MODE` como `strict` en el contenedor `aws-node` del `DaemonSet` de la CNI de la VPC.

```
env:
  - name: NETWORK_POLICY_ENFORCING_MODE
    value: "strict"
```

Paso 2: habilite el parámetro de política de red para el complemento

La característica de la política de red utiliza el puerto `8162` del nodo para las métricas de manera predeterminada. Además, la característica utiliza el puerto `8163` para las sondas de estado.
Si ejecuta otra aplicación en los nodos o dentro de pods que necesite utilizar estos puertos, la aplicación no se puede ejecutar. En la versión de CNI de VPC `v1.14.1` o posterior, puede cambiar estos puertos.

Use el siguiente procedimiento para habilitar el parámetro de política de red para el complemento.

Consola de administración de AWS

1. Abra la consola de Amazon EKS.
2. En el panel de navegación izquierdo, seleccione Clústeres y, a continuación, seleccione el nombre del clúster para el que desea configurar el complemento CNI de Amazon VPC.
3. Elija la pestaña Complementos.
4. Seleccione la casilla situada en la parte superior derecha del cuadro y, a continuación, elija Edit (Editar).
5. En la página Configuración de **`Amazon VPC CNI`** , haga lo siguiente:
   a. Seleccione la versión `v1.14.0-eksbuild.3` o posterior en la lista Versión.
   b. Seleccione Ajustes de configuración opcionales.
   c. Introduzca la clave JSON `"enableNetworkPolicy":` y el valor `"true"` en Valores de configuración. El texto resultante debe ser un objeto JSON válido. Si esta clave y este valor son los únicos datos del cuadro de texto, rodee la clave y el valor entre corchetes `{ }`.

El siguiente ejemplo tiene activada la característica de la política de red y las métricas y sondas de estado están configuradas con los números de puertos predeterminados:

```
{
    "enableNetworkPolicy": "true",
    "nodeAgent": {
        "healthProbeBindAddr": "8163",
        "metricsBindAddr": "8162"
    }
}
```

Helm

Si ha instalado el complemento CNI de Amazon VPC para Kubernetes mediante `helm`, puede actualizar la configuración para cambiar los puertos.

1. Ejecute el siguiente comando para cambiar los puertos. Establezca el número de puerto en el valor para la clave `nodeAgent.metricsBindAddr` o `nodeAgent.healthProbeBindAddr`, respectivamente.

```
helm upgrade --set nodeAgent.metricsBindAddr=8162 --set
 nodeAgent.healthProbeBindAddr=8163 aws-vpc-cni --namespace kube-system eks/aws-vpc-
cni
```

kubectl

1. Abra el `DaemonSet` de `aws-node` en el editor.

```
kubectl edit daemonset -n kube-system aws-node
```

2. Reemplace los números de puertos en los siguientes argumentos de comando de `args:` del contenedor `aws-network-policy-agent` del manifiesto del daemonset de `aws-node` de CNI de VPC.

```
    - args:
            - --metrics-bind-addr=:8162
            - --health-probe-bind-addr=:8163
```

Paso 3: configuración del clúster para que use las políticas de red de Kubernetes

Esta configuración se puede usar para un complemento de Amazon EKS o para un complemento autoadministrado.

Complemento de Amazon EKS

Con la AWS CLI, puede configurar el clúster para que utilice las políticas de red de Kubernetes mediante la ejecución del siguiente comando. Reemplace `my-cluster` por el nombre del clúster y el ARN del rol de IAM por el rol que va a usar.

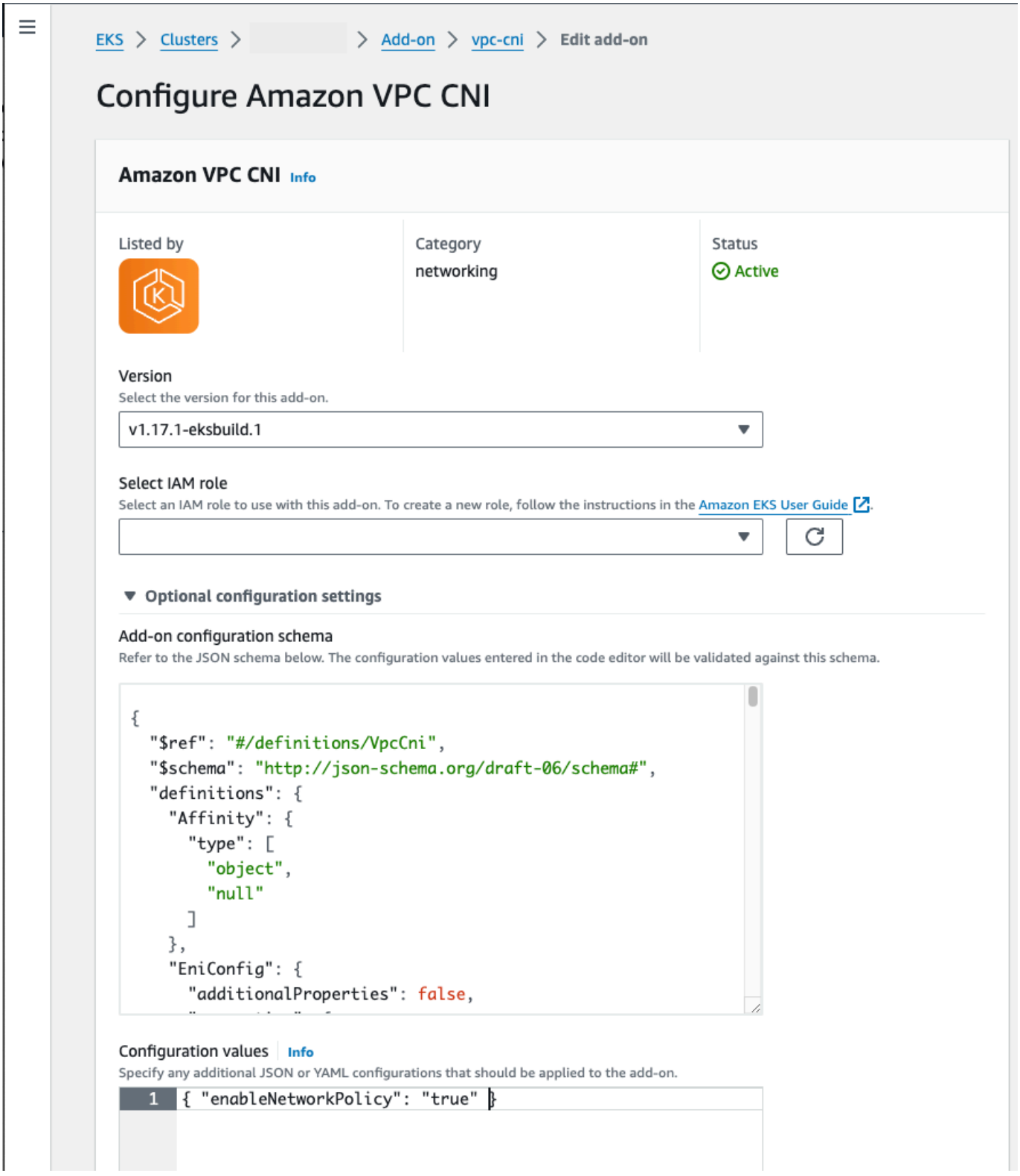
```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-version
 v1.14.0-eksbuild.3 \
    --service-account-role-arn arn:aws:iam::123456789012:role/AmazonEKSVPCCNIRole \
    --resolve-conflicts PRESERVE --configuration-values '{"enableNetworkPolicy":
 "true"}'
```

Para configurarlo mediante la Consola de administración de AWS, siga los pasos que se indican a continuación:

1. Abra la consola de Amazon EKS.
2. En el panel de navegación izquierdo, seleccione Clústeres y, a continuación, seleccione el nombre del clúster para el que desea configurar el complemento CNI de Amazon VPC.
3. Elija la pestaña Complementos.
4. Seleccione la casilla situada en la parte superior derecha del cuadro y, a continuación, elija Edit (Editar).
5. En la página Configuración de **`Amazon VPC CNI`** , haga lo siguiente:
   a. Seleccione la versión `v1.14.0-eksbuild.3` o posterior en la lista Versión.
   b. Seleccione Ajustes de configuración opcionales.
   c. Introduzca la clave JSON `"enableNetworkPolicy":` y el valor `"true"` en Valores de configuración. El texto resultante debe ser un objeto JSON válido. Si esta clave y este valor son los únicos datos del cuadro de texto, rodee la clave y el valor entre corchetes `{ }`. En el siguiente ejemplo se muestra que la política de red está habilitada:

      ```
      { "enableNetworkPolicy": "true" }
      ```

      En la siguiente captura de pantalla se muestra un ejemplo de este escenario.


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" }

Complemento autoadministrado

Helm

Si ha instalado el complemento CNI de Amazon VPC para Kubernetes mediante `helm`, puede actualizar la configuración para permitir la política de red.

1. Ejecute el siguiente comando para habilitar la política de red.

```
helm upgrade --set enableNetworkPolicy=true aws-vpc-cni --namespace kube-system eks/
aws-vpc-cni
```

kubectl

1. Abra el `ConfigMap` de `amazon-vpc-cni` en el editor.

```
kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
```

2. Añada la línea siguiente al `data` de `ConfigMap`.

```
enable-network-policy-controller: "true"
```

   Una vez que haya añadido la línea, su `ConfigMap` debería tener un aspecto similar al siguiente ejemplo.

```
apiVersion: v1
 kind: ConfigMap
 metadata:
  name: amazon-vpc-cni
  namespace: kube-system
 data:
  enable-network-policy-controller: "true"
```

3. Abra el `DaemonSet` de `aws-node` en el editor.

```
kubectl edit daemonset -n kube-system aws-node
```

a. Sustituya el `false` por `true` en el argumento del comando `--enable-network-policy=false` en el `args:` del contenedor `aws-network-policy-agent` en el manifiesto del daemonset de `aws-node` del CNI de VPC.

```
    - args:
       - --enable-network-policy=true
```

Paso 4. Siguientes pasos

Después de completar la configuración, confirme que los pods `aws-node` se estén ejecutando en el clúster.

```
kubectl get pods -n kube-system | grep 'aws-node\|amazon'
```

Un ejemplo de salida sería el siguiente.

```
aws-node-gmqp7                                          2/2     Running   1 (24h ago)
 24h
aws-node-prnsh                                          2/2     Running   1 (24h ago)
 24h
```

Hay 2 contenedores en los pods `aws-node` en las versiones `1.14` y posteriores. En versiones anteriores, y si la política de red está deshabilitada, solo hay un contenedor en los pods de `aws-node`.

Ahora puede implementar políticas de red de Kubernetes en su clúster.

Para implementar políticas de red de Kubernetes, puede crear objetos `NetworkPolicy` o `ClusterNetworkPolicy` de Kubernetes e implementarlos en su clúster. Los objetos `NetworkPolicy` están limitados a un espacio de nombres, mientras que los objetos `ClusterNetworkPolicy` se pueden limitar a todo el clúster o varios espacios de nombres. Implemente políticas para permitir o denegar el tráfico entre pods en función de los selectores de etiquetas, los espacios de nombres y los rangos de direcciones IP. Para obtener más información sobre cómo crear objetos `NetworkPolicy`, consulte Network Policies en la documentación de Kubernetes.

La aplicación de objetos `NetworkPolicy` de Kubernetes se implementa con Extended Berkeley Packet Filter (eBPF). En relación con las implementaciones basadas en `iptables`, ofrece

características de rendimiento y latencia más bajas, que incluyen una menor utilización de la CPU y evitan las búsquedas secuenciales. Además, las sondas de eBPF proporcionan acceso a datos en contexto que ayudan a depurar problemas complejos a nivel del núcleo y a mejorar la observabilidad. Amazon EKS es compatible con un exportador basado en eBPF que aprovecha las sondas para registrar los resultados de las políticas en cada nodo y exportar los datos a recopiladores de registros externos para facilitar la depuración. Para obtener más información, consulte la documentación de eBPF.

Cómo deshabilitar las políticas de red de Kubernetes para el tráfico de red del pod de Amazon EKS

Cómo deshabilitar las políticas de red de Kubernetes para dejar de restringir el tráfico de red del pod de Amazon EKS

1. Enumere todas las políticas de red de Kubernetes.

```
kubectl get netpol -A
```

2. Elimine cada política de red de Kubernetes. Debe eliminar todas las políticas de red antes de desactivarlas.

```
kubectl delete netpol <policy-name>
```

3. Abra el DaemonSet aws-node en el editor.

```
kubectl edit daemonset -n kube-system aws-node
```

4. Sustituya el `true` por `false` en el argumento del comando `--enable-network-policy=true` en el `args:` del contenedor `aws-network-policy-agent` en el manifiesto del daemonset de `aws-node` del CNI de VPC.

```
    - args:
       - --enable-network-policy=true
```

Resolución de problemas de políticas de red de Kubernetes para Amazon EKS

Esta es la guía de solución de problemas para la característica de la política de red de la CNI de Amazon VPC.

Esta guía aborda los siguientes temas:

- Información de instalación, CRD y permisos the section called “Nuevos permisos y CRD `policyendpoints`” de RBAC
- Registros que se deben examinar durante el diagnóstico de problemas the section called “Registros de políticas de red” en la política de la red
- Ejecución de la colección de herramientas del SDK de eBPF para solucionar problemas
- Problemas conocidos y soluciones the section called “Problemas conocidos y soluciones”

> **Note**
>
> Tenga en cuenta que las políticas de red solo se aplican a los pods creados por Kubernetes Deployments. Para obtener más información sobre las limitaciones de las políticas de red en la CNI de la VPC, consulte the section called “Consideraciones”.

Puede solucionar problemas e investigar las conexiones de red que utilizan políticas de red leyendo los registros de políticas de red y ejecutando las herramientas del eBPF SDK.

Nuevos permisos y CRD **`policyendpoints`**

- CRD: `policyendpoints.networking.k8s.aws`
- Kubernetes API: `apiservice` denominado `v1.networking.k8s.io`
- Recurso de Kubernetes: `Kind: NetworkPolicy`
- RBAC: `ClusterRole` denominado `aws-node` (CNI de la VPC), `ClusterRole` llamó a `eks:network-policy-controller` (controlador de políticas de red en el plano de control del clúster de EKS)

Para la política de red, la CNI de la VPC crea una nueva `CustomResourceDefinition` (CRD) llamada `policyendpoints.networking.k8s.aws`. La CNI de la VPC debe tener permisos para crear la CRD y crear CustomResources (CR) de esta y de las otras CRD instaladas por la CNI de la VPC (`eniconfigs.crd.k8s.amazonaws.com`). Ambas CRD están disponibles en el archivo `crds.yaml` de GitHub. En concreto, la CNI de la VPC debe tener los permisos verbales “get”, “list” y “watch” para `policyendpoints`.

La política de red de Kubernetes forma parte del `apiservice` denominado `v1.networking.k8s.io`, que es `apiversion: networking.k8s.io/v1` en los archivos YAML

de la política. La CNI de la VPC `DaemonSet` debe tener permisos para usar esta parte de la API de Kubernetes.

Los permisos de la CNI de la VPC están en un `ClusterRole` denominado `aws-node`. Tenga en cuenta que los objetos `ClusterRole` no se agrupan en los espacios de nombres. A continuación se muestra el `aws-node` de un clúster:

```
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
```

Además, se ejecuta un nuevo controlador en el plano de control de cada clúster de EKS. El controlador usa los permisos del `ClusterRole` denominado `eks:network-policy-controller`. A continuación se muestra el `eks:network-policy-controller` de un clúster:

```
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
```

Registros de políticas de red

Cada decisión que la CNI de la VPC toma con respecto a si las políticas de red permiten o deniegan las conexiones se registra en los registros de flujo. Los registros de políticas de red de cada nodo incluyen los registros de flujo de cada pod que tiene una política de red. Los registros de políticas de red se almacenan en `/var/log/aws-routed-eni/network-policy-agent.log`. A continuación se muestra un ejemplo de un archivo `network-policy-agent.log`:

```
{"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"}
```

Los registros de políticas de red está deshabilitados de manera predeterminada. Para habilitar los registros de políticas de red, siga estos pasos:

> **Note**
>
> Los registros de políticas de red requieren 1 vCPU adicional para el contenedor `aws-network-policy-agent` del manifiesto `aws-node DaemonSet` de la CNI de la VPC.

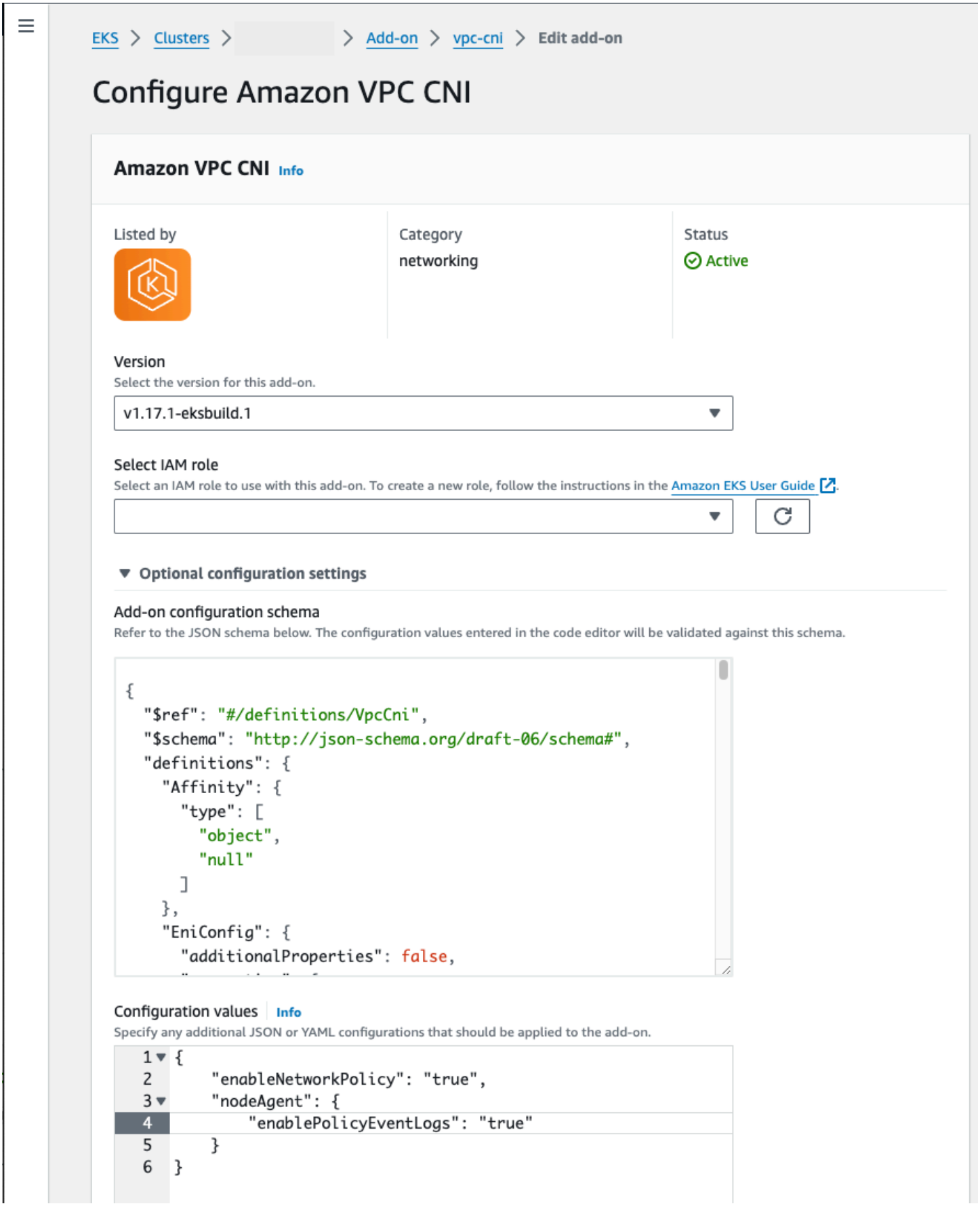
Complemento de Amazon EKS

Consola de administración de AWS

a. Abra la consola de Amazon EKS.
b. En el panel de navegación izquierdo, seleccione Clústeres y, a continuación, seleccione el nombre del clúster para el que desea configurar el complemento CNI de Amazon VPC.
c. Elija la pestaña Complementos.
d. Seleccione la casilla situada en la parte superior derecha del cuadro y, a continuación, elija Edit (Editar).
e. En la página Configurar ***CNI de Amazon VPC***:
   i. Seleccione la versión `v1.14.0-eksbuild.3` o posterior en la lista desplegable Versión.
   ii. Seleccione Ajustes de configuración opcionales.
   iii. Introduzca la clave de JSON de nivel superior `"nodeAgent":` y el valor en un objeto con una clave `"enablePolicyEventLogs":` y un valor de `"true"` en Valores de configuración. El texto resultante debe ser un objeto JSON válido. En el siguiente ejemplo se muestra que la política de red y los registros de políticas de red están habilitados, y que estos últimos se envían a los Registros de CloudWatch:

```
{
    "enableNetworkPolicy": "true",
    "nodeAgent": {
        "enablePolicyEventLogs": "true"
    }
}
```

En la siguiente captura de pantalla se muestra un ejemplo de este escenario.


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. Ejecute el siguiente comando de AWS CLI. Reemplace `my-cluster` por el nombre del clúster y el ARN del rol de IAM por el rol que va a usar.

```
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"}}'
```

Complemento autoadministrado

Helm

Si ha instalado el complemento CNI de Amazon VPC para Kubernetes mediante `helm`, puede actualizar la configuración para escribir los registros de la política de red.

a. Ejecute el siguiente comando para habilitar la política de red.

```
helm upgrade --set nodeAgent.enablePolicyEventLogs=true aws-vpc-cni --namespace
 kube-system eks/aws-vpc-cni
```

kubectl

Si ha instalado el complemento CNI de Amazon VPC para Kubernetes mediante `kubectl`, puede actualizar la configuración para escribir los registros de la política de red.

a. Abra el `DaemonSet` de `aws-node` en el editor.

```
kubectl edit daemonset -n kube-system aws-node
```

b. Sustituya el `false` por `true` en el argumento del comando `--enable-policy-event-logs=false` en el `args:` del contenedor `aws-network-policy-agent` en el manifiesto `aws-node` `DaemonSet` de la CNI de la VPC.

```
    - args:
       - --enable-policy-event-logs=true
```

Enviar los registros de políticas de red a los Registros de Amazon CloudWatch

Puede supervisar los registros de políticas de red mediante servicios como los Registros de Amazon CloudWatch. Puede usar los siguientes métodos para enviar los registros de políticas de red a los Registros de CloudWatch.

En el caso de los clústeres de EKS, los registros de políticas se ubicarán en `/aws/eks/`*`cluster-name`*`/cluster/` y, en el caso de los clústeres K8S autoadministrados, los registros se colocarán en `/aws/k8s-cluster/cluster/`.

Envío de registros de la política de red con el complemento CNI de Amazon VPC para Kubernetes

Si habilita la política de red, se añade un segundo contenedor a los pods de `aws-node` para un agente de nodos. Este agente de nodos puede enviar los registros de políticas de red a los Registros de CloudWatch.

> **Note**
>
> El agente de nodos solo envía los registros de políticas de red. No se incluyen otros registros creados por el CNI de VPC.

Requisitos previos

- Añada los siguientes permisos como una política individual o independiente al rol de IAM que está utilizando para el CNI de VPC.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups",
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        }
```

```
    ]
}
```

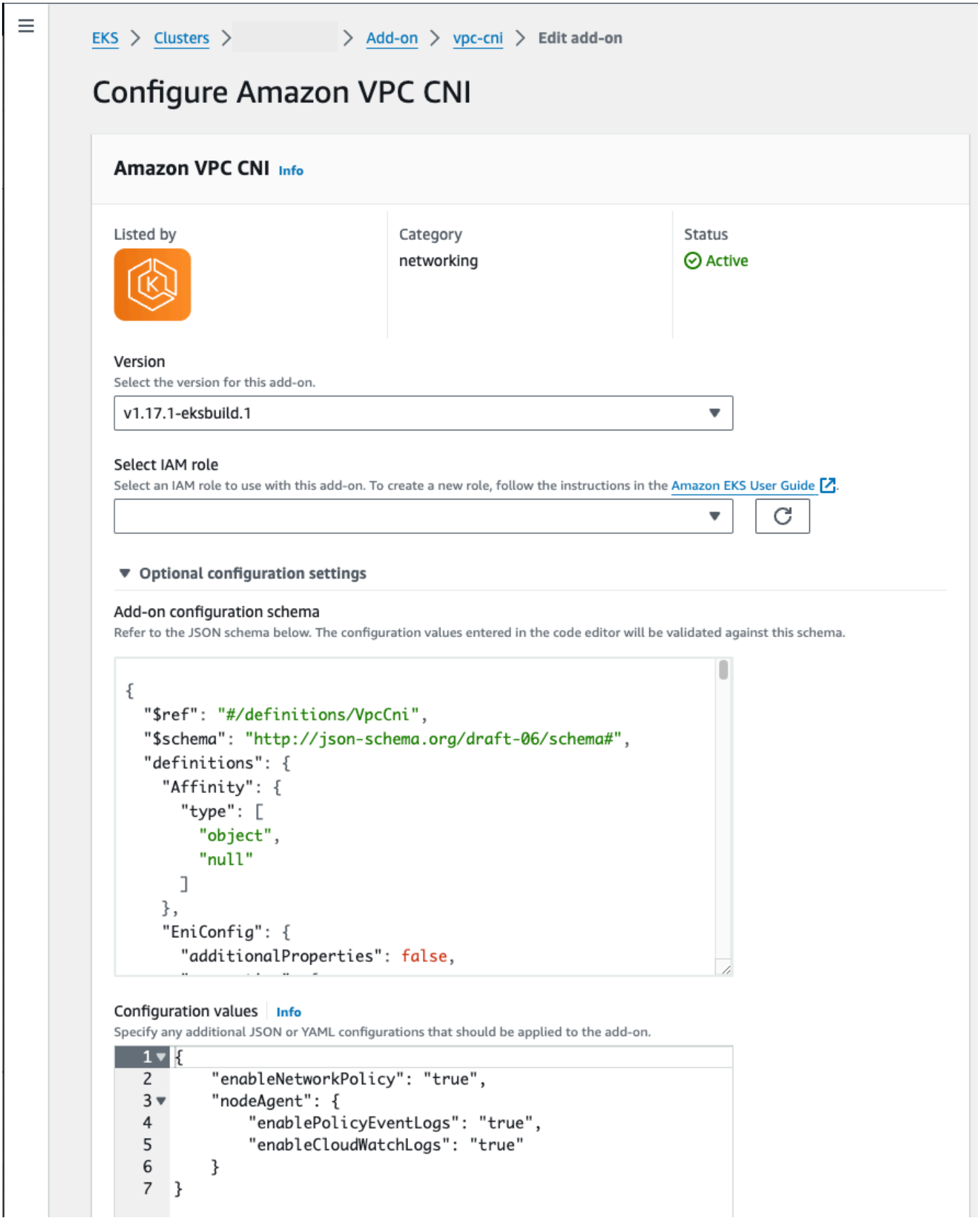
Complemento de Amazon EKS

Consola de administración de AWS

a. Abra la consola de Amazon EKS.
b. En el panel de navegación izquierdo, seleccione Clústeres y, a continuación, seleccione el nombre del clúster para el que desea configurar el complemento CNI de Amazon VPC.
c. Elija la pestaña Complementos.
d. Seleccione la casilla situada en la parte superior derecha del cuadro y, a continuación, elija Edit (Editar).
e. En la página Configurar ***CNI de Amazon VPC***:
   i. Seleccione la versión `v1.14.0-eksbuild.3` o posterior en la lista desplegable Versión.
   ii. Seleccione Ajustes de configuración opcionales.
   iii. Introduzca la clave de JSON de nivel superior `"nodeAgent":` y el valor en un objeto con una clave `"enableCloudWatchLogs":` y un valor de `"true"` en Valores de configuración. El texto resultante debe ser un objeto JSON válido. En el siguiente ejemplo se muestra que la política de red y los registros de políticas de red están habilitados, y que estos últimos se envían a los Registros de CloudWatch:

```
{
    "enableNetworkPolicy": "true",
    "nodeAgent": {
        "enablePolicyEventLogs": "true",
        "enableCloudWatchLogs": "true",
    }
}
```

En la siguiente captura de pantalla se muestra un ejemplo de este escenario.


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. Ejecute el siguiente comando de AWS CLI. Reemplace `my-cluster` por el nombre del clúster y el ARN del rol de IAM por el rol que va a usar.

```
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"}}'
```

Complemento autoadministrado

Helm

Si ha instalado el complemento CNI de Amazon VPC para Kubernetes mediante `helm`, puede enviar la configuración para enviar registros de la política de red a Registros de CloudWatch.

a. Ejecute el siguiente comando para habilitar los registros de políticas de red y enviarlos a los Registros de CloudWatch.

```
helm upgrade --set nodeAgent.enablePolicyEventLogs=true --set
 nodeAgent.enableCloudWatchLogs=true aws-vpc-cni --namespace kube-system eks/aws-
vpc-cni
```

kubectl

a. Abra el `DaemonSet` de `aws-node` en el editor.

```
kubectl edit daemonset -n kube-system aws-node
```

b. Sustituya `false` por `true` en los dos argumentos de comando `--enable-policy-event-logs=false` y `--enable-cloudwatch-logs=false` de `args:` del contenedor `aws-network-policy-agent` en el manifiesto `aws-node` `DaemonSet` de la CNI de la VPC.

```
    - args:
       - --enable-policy-event-logs=true
       - --enable-cloudwatch-logs=true
```

Envío de registros de políticas de red con un **`DaemonSet`** de Fluent Bit

Si utiliza Fluent Bit en un `DaemonSet` para enviar registros desde sus nodos, puede agregar una configuración para incluir los registros de políticas de red de las políticas de red. Puede utilizar la siguiente configuración de ejemplo:

```
    [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
```

SDK de eBPF incluido

El complemento CNI de Amazon VPC para Kubernetes instala el conjunto de herramientas del SDK de eBPF en los nodos. Puede utilizar las herramientas del SDK de eBPF para identificar problemas con las políticas de red. Por ejemplo, el siguiente comando muestra los programas que se ejecutan en el nodo.

```
sudo /opt/cni/bin/aws-eks-na-cli ebpf progs
```

Para ejecutar este comando, puede utilizar cualquier método para conectarse al nodo.

Problemas conocidos y soluciones

En las siguientes secciones se describen los problemas conocidos con la característica de la política de red de la CNI de Amazon VPC y sus soluciones.

Los registros de políticas de red se generan a pesar de que enable-policy-event-logs esté establecido en false

Problema: la CNI de VPC de EKS genera registros de políticas de red incluso cuando la configuración `enable-policy-event-logs` está establecida en `false`.

Solución: la configuración `enable-policy-event-logs` solo deshabilita los registros de “decisiones” de las políticas, pero no deshabilita todos los registros del agente de políticas de red. Este comportamiento se documenta en el archivo README aws-network-policy-agent de GitHub.

Para deshabilitar por completo el registro, es posible que tenga que ajustar otras configuraciones de registro.

Problemas de limpieza de asignación de políticas de red

Problema: los problemas con la red `policyendpoint` siguen existiendo y no se están limpiando después de eliminar los pods.

Solución: Esto se debe a un problema con la versión 1.19.3-eksbuild.1 del complemento CNI de VPC. Actualice el complemento CNI de VPC a una versión más reciente para resolver este problema.

No se aplican políticas de red

Problema: la característica de políticas de red está habilitada en el complemento CNI de Amazon VPC, pero las políticas de red no se aplican correctamente.

Si crea una política de red `kind: NetworkPolicy` y no impacta en el pod, compruebe que el objeto policyendpoint se haya creado en el mismo espacio de nombres que el pod. Si no hay objetos `policyendpoint` en los espacios de nombres, el controlador de políticas de red (parte del clúster de EKS) no pudo crear reglas de políticas de red para que las aplicara el agente de políticas de red (parte de la CNI de la VPC).

Solución: la solución consiste en corregir los permisos de la CNI de la VPC (`ClusterRole - aws-node`) y del controlador de políticas de red (`ClusterRole - eks:network-policy-controller`) y permitir estas acciones en cualquier herramienta de aplicación de políticas, como Kyverno. Asegúrese de que las políticas de Kyverno no bloqueen la creación de objetos `policyendpoint`. Consulte la sección anterior para conocer los permisos necesarios en the section called “Nuevos permisos y CRD `policyendpoints`”.

Los pods no vuelven al estado de denegación predeterminado después de eliminar la política en modo estricto

Problema: cuando las políticas de red están habilitadas en modo estricto, los pods comienzan con una política de denegación predeterminada. Una vez aplicadas las políticas, se permite el tráfico a los puntos de conexión especificados. Sin embargo, cuando se eliminan las políticas, el pod no vuelve al estado de denegación predeterminado, sino que pasa a un estado de permiso predeterminado.

Solución: este problema se solucionó en la versión 1.19.3 de la CNI de la VPC, que incluía la versión 1.2.0 del agente de políticas de red. Tras la corrección, con el modo estricto habilitado, una

vez eliminadas las políticas, el pod vuelve al estado de denegación predeterminado, tal y como se esperaba.

Latencia de inicio de los grupos de seguridad para los pods

Problema: cuando se utiliza la característica Grupos de seguridad para los pods en EKS, aumenta la latencia de inicio de los pods.

Solución: la latencia se debe a la limitación de velocidad del controlador de recursos debido a la limitación de la API de `CreateNetworkInterface`, que el controlador de recursos de la VPC utiliza para crear ENI de rama para los pods. Compruebe los límites de la API de su cuenta para esta operación y considere la posibilidad de solicitar un aumento del límite si es necesario.

FailedScheduling debido a una cantidad insuficiente de vpc.amazonaws.com/pod-eni

Problema: los pods no se programan y se produce el error `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.`

Solución: al igual que en el problema anterior, la asignación de grupos de seguridad a los pods aumenta la latencia de programación de los pods y puede sobrepasar el umbral de la CNI para añadir cada ENI, lo que provoca errores al iniciar los pods. Este es el comportamiento previsto cuando se utilizan grupos de seguridad para los pods. Considere las implicaciones de la programación al momento de diseñar la arquitectura de su carga de trabajo.

Problemas de conectividad de IPAM y errores de segmentación

Problema: se producen varios errores, incluidos problemas de conectividad de IPAM, solicitudes de limitación y errores de segmentación.

- `Checking for IPAM connectivity …`
- `Throttling request took 1.047064274s`
- `Retrying waiting for IPAM-D`
- `panic: runtime error: invalid memory address or nil pointer dereference`

Solución: este problema se produce si se instala `systemd-udev` en AL2023, ya que el archivo se reescribe con una política de interrupción. Esto puede ocurrir cuando se actualiza a un `releasever`

diferente que tiene un paquete actualizado o cuando se actualiza manualmente el paquete en sí. Evite instalar o actualizar `systemd-udev` en los nodos AL2023.

Error al buscar el dispositivo por su nombre

Problema: mensaje de error `{"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})"}`

Solución: este problema se ha identificado y corregido en las versiones más recientes del agente de políticas de red CNI de Amazon VPC (v1.2.0). Actualice a la última versión de la CNI de VPC para resolver este problema.

Vulnerabilidades de CVE en la imagen CNI de Multus

Problema: el informe CVE mejorado de EKS ImageScan identifica vulnerabilidades en la versión v4.1.4-eksbuild.2_thick de la imagen CNI de Multus.

Solución: actualice a la nueva versión de la imagen CNI de Multus y a la nueva imagen del Controlador de políticas de red, que no tienen vulnerabilidades. El escáner se puede actualizar para corregir las vulnerabilidades encontradas en la versión anterior.

Veredictos de denegación del flujo de información en los registros

Problema: los registros de políticas de red muestran veredictos de denegación `{"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"}`

Solución: este problema se ha resuelto en la nueva versión del Controlador de políticas de red. Actualice a la versión más reciente de la plataforma de EKS para resolver los problemas de registro.

Problemas de comunicación entre pods después de migrar desde Calico

Problema: tras actualizar un clúster de EKS a la versión 1.30 y cambiar de Calico a la CNI de Amazon VPC para la política de red, la comunicación entre pods falla cuando se aplican las políticas de red. La comunicación se restablece cuando se eliminan las políticas de red.

Solución: el agente de políticas de red de la CNI de la VPC no puede tener tantos puertos especificados como Calico. En su lugar, utilice rangos de puertos en las políticas de red. El número

máximo de combinaciones únicas de puertos para cada protocolo en cada selector de `ingress:` o `egress:` en una política de red es 24. Utilice rangos de puertos para reducir la cantidad de puertos únicos y evitar esta limitación.

El agente de políticas de red no admite pods independientes

Problema: las políticas de red aplicadas a los pods independientes pueden tener un comportamiento inconsistente.

Solución: actualmente, el agente de políticas de red solo admite los pods que se implementan como parte de un conjunto de réplicas/implementación. Si las políticas de red se aplican a los pods independientes, es posible que se produzcan algunas inconsistencias en el comportamiento. Esto se documenta en la parte superior de esta página, en the section called “Consideraciones”, y en aws-network-policy-agent GitHub issue #327 en GitHub. Implemente los pods como parte de un conjunto de réplicas o implementación para lograr un comportamiento consistente de las políticas de red.

Demostración de Stars de política de red para Amazon EKS

Esta demostración crea un frontend, un backend y un servicio cliente en el clúster de Amazon EKS. La demostración también crea una interfaz gráfica de usuario de administración que muestra las rutas de entrada y salida disponibles entre los servicios. Le recomendamos que complete la demostración en un clúster en el que no ejecute cargas de trabajo de producción.

Cuando aún no se ha creado ninguna política de red, todos los servicios pueden comunicarse en ambas direcciones. Después de aplicar las políticas de red, puede ver que el cliente solo puede comunicarse con el servicio frontend y el backend solo puede aceptar tráfico del frontend.

1. Aplique los servicios de frontend, backend, cliente e interfaz de usuario de administración:

```
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. Visualice todos los pods en el clúster.

```
kubectl get pods -A
```

   Un ejemplo de salida sería el siguiente.

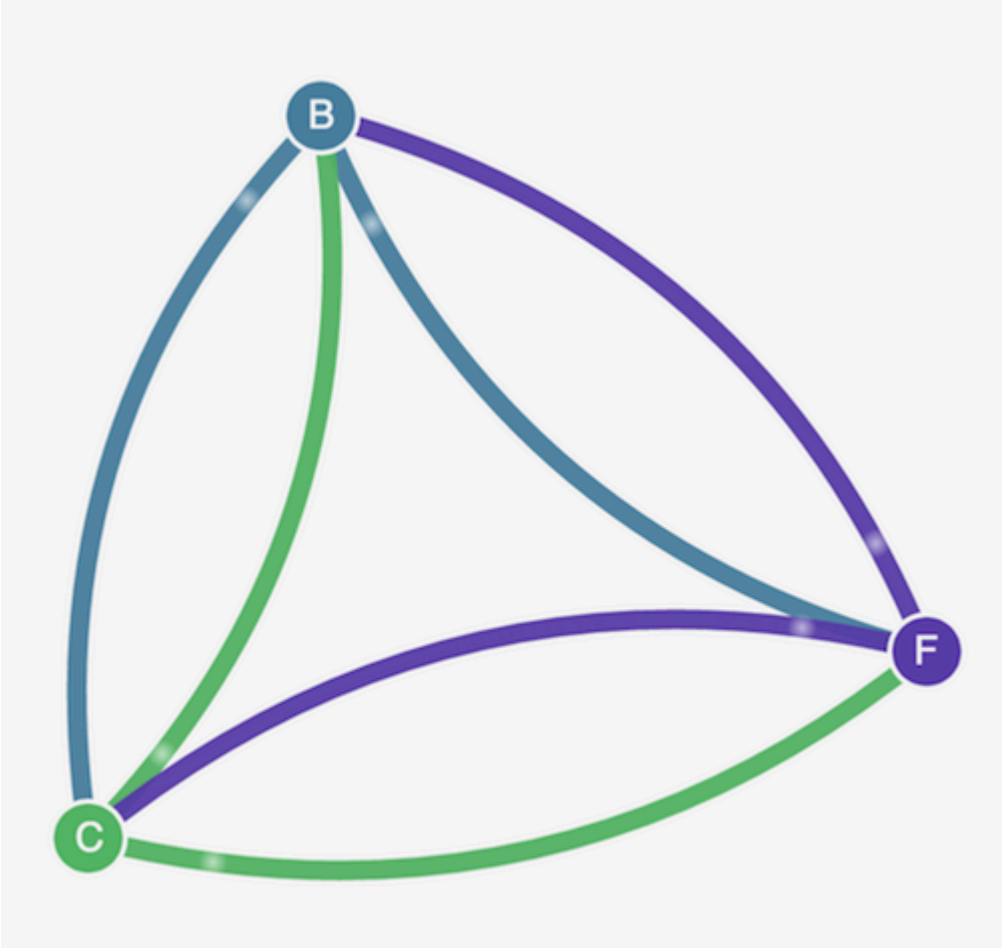
   En la salida, debería ver pods en los espacios de nombres que se muestran en la siguiente salida. Los valores de *NAMES* y el número de pods de la columna READY son diferentes de los de la siguiente salida. No continúe hasta que vea pods con nombres similares y todos tengan `Running` en la columna STATUS.

```
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. Para conectarse a la interfaz de usuario de administración, conéctese a la `EXTERNAL-IP` del servicio que se ejecuta en el clúster:

```
kubectl get service/management-ui -n management-ui
```

4. Abra el navegador en la ubicación del paso anterior. Debería ver la interfaz de usuario de administración. El nodo C es el servicio cliente, el nodo F es el servicio frontend y el nodo B es el servicio backend. Cada nodo tiene acceso de comunicación completo a todos los demás nodos, como indican las líneas gruesas de colores.



5. Aplique la siguiente política de red en los espacios de nombres de `stars` y `client` para aislar los servicios entre sí:

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: default-deny
spec:
  podSelector:
    matchLabels: {}
```

Puede usar los siguientes comandos para aplicar la política a ambos espacios de nombres:

```
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. Actualice su navegador. Verá que la interfaz de usuario de administración ya no tiene acceso a los nodos, que dejan de aparecer en la interfaz de usuario.
7. Aplique las siguientes políticas de red distintas para permitir el acceso de la interfaz de usuario de administración a los servicios. Aplique esta política para permitir la interfaz de usuario:

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: allow-ui
spec:
  podSelector:
    matchLabels: {}
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: management-ui
```

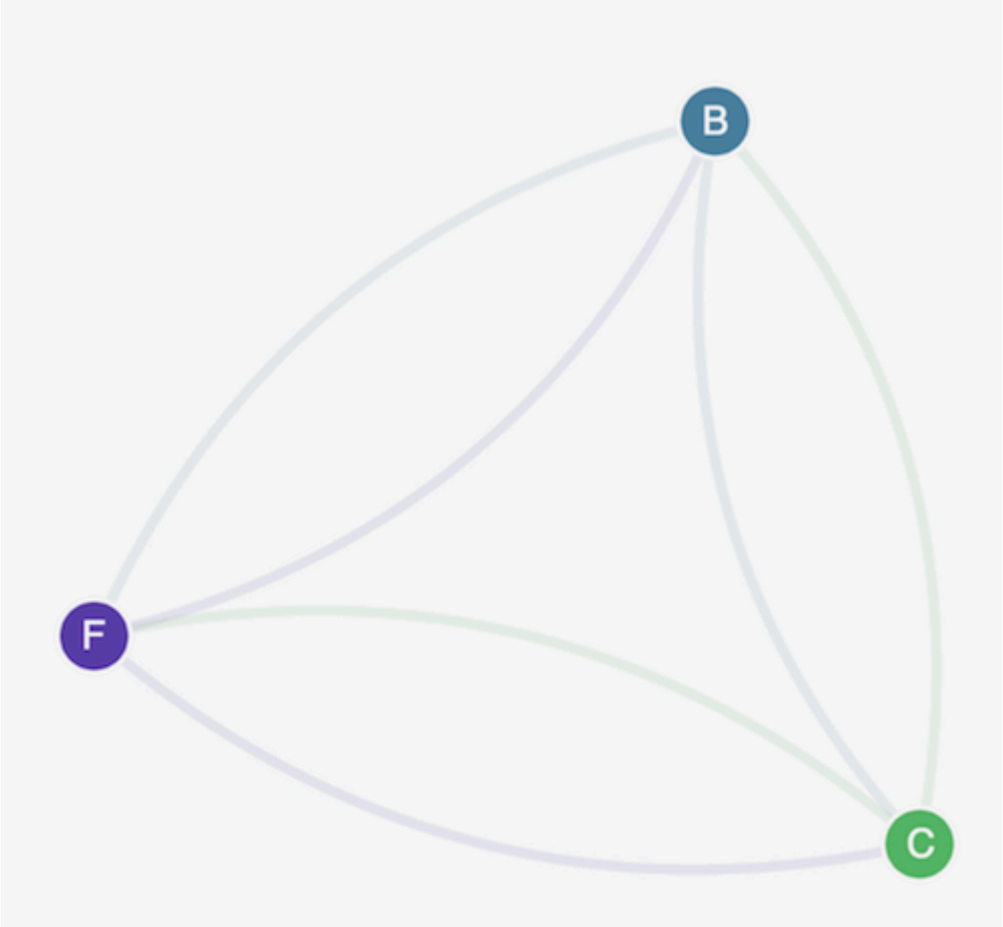
Aplique esta política para permitir el cliente:

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: client
  name: allow-ui
spec:
  podSelector:
    matchLabels: {}
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: management-ui
```

Puede usar los siguientes comandos para aplicar ambas políticas:

```
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. Actualice su navegador. Verá que la interfaz de usuario de administración tiene de nuevo acceso a los nodos, pero los nodos no pueden comunicarse entre sí.

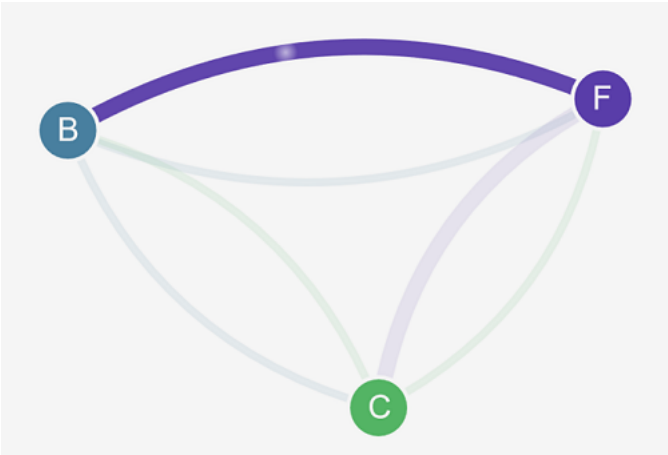


9. Aplique la siguiente política de red para permitir el tráfico desde el servicio frontend hacia el servicio backend:

```
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 Actualice su navegador. Puede ver que el frontend puede comunicarse con el backend.

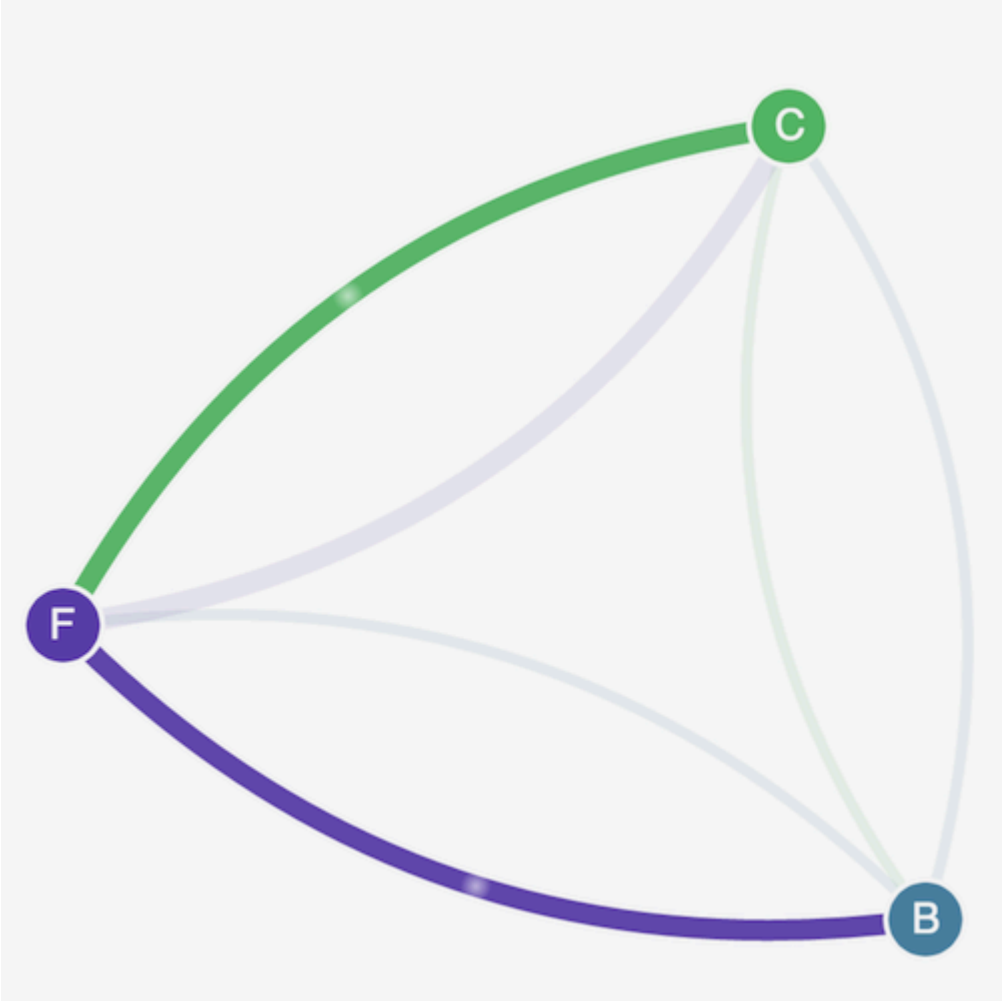


11 Aplique la siguiente política de red para permitir el tráfico desde el cliente hacia el servicio frontend:

```
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. Actualice su navegador. Verá que el cliente puede comunicarse con el servicio frontend. El servicio frontend puede seguir comunicándose con el servicio backend.



13. (Opcional) Cuando haya terminado con la demostración, puede eliminar sus recursos.

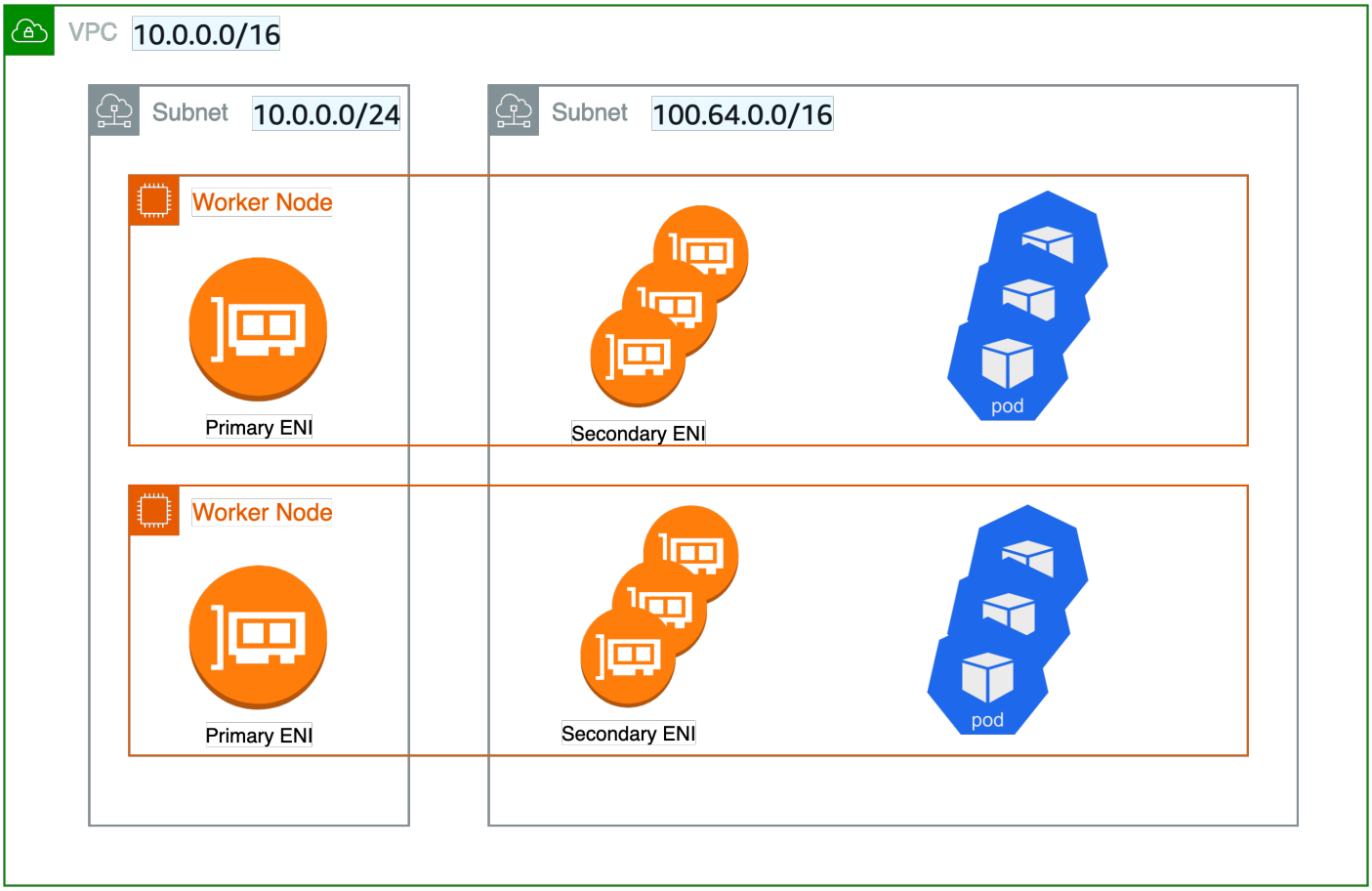
```
kubectl delete -f https://raw.githubusercontent.com/aws-samples/eks-
workshop/2f9d29ed3f82ed6b083649e975a0e574fb8a4058/content/beginner/120_network-
policies/calico/stars_policy_demo/create_resources.files/client.yaml
kubectl delete -f https://raw.githubusercontent.com/aws-samples/eks-
workshop/2f9d29ed3f82ed6b083649e975a0e574fb8a4058/content/beginner/120_network-
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
```

Incluso después de eliminar los recursos, todavía puede haber puntos de conexión de las políticas de red en los nodos que podrían interferir de manera inesperada con las redes en el clúster. La única forma segura de eliminar estas reglas es reiniciar los nodos o terminar todos los nodos y reciclarlos. Para terminar todos los nodos, establezca el recuento deseado del grupo de Auto Scaling en 0 y, a continuación, realice una copia de seguridad en el número deseado o simplemente termine los nodos.

Implementación de pods en subredes alternativas con redes personalizadas

Aplicación: en nodos de `IPv4` de Linux de Fargate, y nodos de Linux con instancias de Amazon EC2



De forma predeterminada, cuando el complemento CNI de Amazon VPC para Kubernetes crea interfaces de red elásticas secundarias (interfaces de red) para el nodo de Amazon EC2, las crea en la misma subred que la interfaz de red principal del nodo. También asocia los mismos grupos de seguridad a la interfaz de red secundaria asociada a la interfaz de red principal. Por uno o varios de los siguientes motivos, es posible que desee que el complemento cree interfaces de red secundarias en una subred diferente o desee asociar distintos grupos de seguridad a las interfaces de red secundarias, o ambas:

- Hay un número limitado de direcciones `IPv4` que están disponibles en la subred en la que se encuentra la interfaz de red principal. Esto podría limitar el número de pods que puede crear en la subred. Si utiliza una subred diferente para las interfaces de red secundarias, puede aumentar el número de direcciones `IPv4` disponibles para pods.
- Por motivos de seguridad, sus pods podrían tener que utilizar distintos grupos de seguridad o subredes que la interfaz de red principal del nodo.
- Los nodos se encuentran configurados en subredes públicas y debe ubicar los pods en subredes privadas. La tabla de enrutamiento asociada a una subred pública incluye una puerta de enlace de Internet. La tabla de enrutamiento asociada a una subred privada no incluye ninguna puerta de enlace de Internet.

> **Tip**
>
> También puede agregar una subred nueva o existente directamente al clúster de Amazon EKS, sin necesidad de utilizar redes personalizadas. Para obtener más información, consulte the section called “Cómo agregar una subred de VPC existente a un clúster de Amazon EKS desde la consola de administración”.

Consideraciones

A continuación, se detallan consideraciones que se deben tener a la hora de utilizar esta característica.

- Con las redes personalizadas habilitadas, no se asignan direcciones IP asignadas a la interfaz de red principal a los pods. Solo se asignan direcciones IP de interfaces de red secundarias a pods.
- Si el clúster utiliza la familia `IPv6`, no puede utilizar redes personalizadas.

- Si planea utilizar redes personalizadas solo para ayudar a aliviar el agotamiento de direcciones `IPv4`, puede crear un clúster mediante la familia `IPv6` en su lugar. Para obtener más información, consulte the section called “IPv6”.
- Si bien los pods implementados en las subredes especificadas para interfaces de red secundarias pueden utilizar subredes y grupos de seguridad diferentes a los de la interfaz de la red principal del nodo, las subredes y los grupos de seguridad deben estar en la misma VPC que el nodo.
- En el caso de Fargate, las subredes se controlan mediante el perfil de Fargate. Para obtener más información, consulte the section called “Definición de perfiles”.

Personalización de la interfaz de red secundaria en los nodos de Amazon EKS

Antes de comenzar este tutorial, complete lo siguiente:

- Revisión de las consideraciones
- Haberse familiarizado con cómo el complemento CNI de Amazon VPC para Kubernetes crea interfaces de red secundarias y asigna direcciones IP a los pods. Para obtener más información, consulte Asignación de ENI en GitHub.
- La versión `2.12.3` o posterior, o bien, la versión `1.27.160` o posterior de la AWS interfaz de la línea de comandos (AWS CLI) instalada y configurada en su dispositivo o AWS CloudShell. Para comprobar su versión actual, utilice `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente. Para actualizarla, consulte Instalación de la CLI de AWS en su directorio principal en la Guía del usuario de AWS CloudShell.
- La herramienta de línea de comandos de `kubectl` está instalada en su dispositivo o AWS CloudShell. Para instalar o actualizar `kubectl`, consulte the section called “Configure kubectl y eksctl”.
- Le recomendamos que siga los pasos de este tema en un intérprete de comandos Bash. Si no está utilizando un intérprete de comandos Bash, algunos comandos de script, como los caracteres de continuación de línea y la forma en que se establecen y utilizan las variables, requieren ajustes para su intérprete de comandos. Además, las reglas de entrecomillado y escape de su intérprete de comandos pueden ser diferentes. Para obtener más información, consulte Uso de la AWS CLI con Amazon SNS en la Guía del usuario de la interfaz de línea de comandos de AWS.

Para este tutorial, le recomendamos utilizar los valores de ejemplo, excepto en los casos en que se indique que se los debe reemplazar. Puede reemplazar cualquier ejemplo de valor al completar los pasos de un clúster de producción. Recomendamos completar todos los pasos en la misma terminal. Esto se debe a que las variables se establecen y utilizan a lo largo de los pasos y no existirán en terminales diferentes.

Los comandos de este tema se formatean según las convenciones que se indican en Uso de los ejemplos de la AWS CLI. Si ejecuta comandos desde la línea de comandos con recursos que se encuentran en una región de AWS diferente a la región de AWS predeterminada definida en el perfil de la AWS CLI que está utilizando, tendrá que agregar `--region us-west-2` a los comandos, después de sustituir `us-west-2` por la región de AWS.

Cuando desee implementar redes personalizadas en su clúster de producción, vaya a the section called “Paso 2: Configurar la VPC”.

Paso 1: crear una VPC de prueba y un clúster

Los siguientes procedimientos le ayudan a crear una VPC de prueba y un clúster, y a configurar redes personalizadas para ese clúster. No recomendamos utilizar el clúster de pruebas para cargas de trabajo de producción porque en este tema no se cubren varias características no relacionadas que podría utilizar en el clúster de producción. Para obtener más información, consulte the section called “Creación de un clúster ”.

1. Ejecute el siguiente comando para definir la variable `account_id`.

```
account_id=$(aws sts get-caller-identity --query Account --output text)
```

2. Cree una VPC.

   a. Si va a realizar la implementación en un sistema de prueba, cree una VPC con una plantilla AWS CloudFormation de Amazon EKS.

```
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. La pila de AWS CloudFormation tarda unos minutos en crearse. Para verificar el estado de implementación de la pila, ejecute el siguiente comando.

```
aws cloudformation describe-stacks --stack-name my-eks-custom-networking-vpc --
query Stacks\[\].StackStatus  --output text
```

No continúe con el siguiente paso hasta que la salida del comando sea `CREATE_COMPLETE`.

c. Defina variables con los valores de los ID de subred privada creados por la plantilla.

```
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. Defina variables con las zonas de disponibilidad de las subredes recuperadas en el paso anterior.

```
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. Cree un rol de IAM de clúster.

a. Ejecute el siguiente comando para crear un archivo de política de confianza JSON de IAM.

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

```
}
```

b. Cree el rol de IAM del clúster de Amazon EKS. Si es necesario, prefacio `eks-cluster-role-trust-policy.json` con la ruta del equipo en la que escribió el archivo en el paso anterior. El comando asocia la política de confianza creada en el paso anterior al rol. Para crear un rol de IAM, a la entidad principal de IAM que está creando el rol se le debe asignar la acción `iam:CreateRole` (permiso).

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

c. Adjunte la política administrada de Amazon EKS llamada AmazonEKSClusterPolicy al rol. Para adjuntar una política de IAM a una entidad principal de IAM, se debe asignar una de las siguientes acciones de IAM (permisos) a la entidad principal que adjunta la política: `iam:AttachUserPolicy` o `iam:AttachRolePolicy`.

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

4. Cree un clúster de Amazon EKS y configure su dispositivo para que se comunique con él.

a. Cree un clúster.

```
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**
>
> Es posible que reciba un error que indique que una de las zonas de disponibilidad de la solicitud no tiene capacidad suficiente para crear un clúster de Amazon EKS. Si esto ocurre, el mensaje de error indicará las zonas de disponibilidad que admiten un clúster nuevo. Intente crear el clúster de nuevo con al menos dos subredes ubicadas en las zonas de disponibilidad admitidas para su cuenta. Para obtener más información, consulte the section called “Capacidad insuficiente”.

b. El clúster tarda varios minutos en crearse. Ejecute el siguiente comando para verificar el estado de implementación del clúster.

```
aws eks describe-cluster --name my-custom-networking-cluster --query
 cluster.status
```

No continúe con el siguiente paso hasta que la salida del comando sea `"ACTIVE"`.

c. Configure `kubectl` para comunicarse con el clúster.

```
aws eks update-kubeconfig --name my-custom-networking-cluster
```

Paso 2: Configurar la VPC

Este tutorial requiere una VPC creada en the section called “Paso 1: crear una VPC de prueba y un clúster”. En el caso de un clúster de producción, ajuste los pasos correspondientes a su VPC sustituyendo todos los valores de ejemplo con los suyos propios.

1. Confirme que el complemento CNI de Amazon VPC para Kubernetes instalado actualmente es la última versión. Para determinar la versión más reciente del tipo de complemento de Amazon EKS y actualizar su versión a ella, consulte the section called “Cómo actualizar un complemento”. Para determinar la versión más reciente del tipo de complemento autoadministrado y actualizar su versión a ella, consulte the section called “CNI de Amazon VPC”.
2. Recupere el ID de la VPC de su clúster y guárdelo en una variable para utilizarlo en un paso posterior.

```
vpc_id=$(aws eks describe-cluster --name my-custom-networking-cluster --query
 "cluster.resourcesVpcConfig.vpcId" --output text)
```

3. Asocie un bloque de enrutamiento entre dominios sin clases (CIDR) con la VPC de su clúster. El bloque CIDR no se puede solapar con ningún bloque CIDR asociado existente.

a. Vea los bloques CIDR actuales asociados a la VPC.

```
aws ec2 describe-vpcs --vpc-ids $vpc_id \
    --query 'Vpcs[*].CidrBlockAssociationSet[*].{CIDRBlock: CidrBlock, State:
 CidrBlockState.State}' --out table
```

Un ejemplo de salida sería el siguiente.

```
---------------------------------
|          DescribeVpcs         |
```

```
+-----------------+-------------+
|    CIDRBlock    |    State    |
+-----------------+-------------+
|  192.168.0.0/24 |  associated |
+-----------------+-------------+
```

b. Asocie un bloque CIDR adicional a su VPC. Sustituya el valor del bloque de CIDR en el siguiente comando. Para obtener más información, consulte Asociar un bloque adicional de CIDR IPv4 a su VPC en la Guía del usuario de Amazon VPC.

```
aws ec2 associate-vpc-cidr-block --vpc-id $vpc_id --cidr-block 192.168.1.0/24
```

c. Confirme que el nuevo bloque está asociado.

```
aws ec2 describe-vpcs --vpc-ids $vpc_id --query
 'Vpcs[*].CidrBlockAssociationSet[*].{CIDRBlock: CidrBlock, State:
 CidrBlockState.State}' --out table
```

Un ejemplo de salida sería el siguiente.

```
---------------------------------
|          DescribeVpcs         |
+-----------------+-------------+
|    CIDRBlock    |    State    |
+-----------------+-------------+
|  192.168.0.0/24 |  associated |
|  192.168.1.0/24 |  associated |
+-----------------+-------------+
```

No continúe con el siguiente paso hasta que el `State` del nuevo bloque CIDR sea `associated`.

4. Cree tantas subredes como desee utilizar en cada zona de disponibilidad en la que se encuentran las subredes existentes. Especifique un bloque CIDR que se encuentra dentro del bloque CIDR que asoció a la VPC en un paso anterior.

a. Crea nuevas subredes. Sustituya los valores del bloque de CIDR en el siguiente comando. Las subredes deben crearse en un bloque CIDR de VPC diferente al que están las subredes existentes, pero en las mismas zonas de disponibilidad que las subredes existentes. En este ejemplo, se crea una subred en el nuevo bloque CIDR de cada zona de disponibilidad en la que existen las subredes privadas actuales. Los ID de las subredes creadas se almacenan en variables para usarlas en los pasos posteriores. Los valores `Name` coinciden con los valores

asignados a las subredes creadas mediante la plantilla de Amazon EKS VPC en un paso anterior. No se requieren nombres. Puede utilizar diferentes nombres.

```
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**
>
> De forma predeterminada, las nuevas subredes están asociadas implícitamente con su tabla de enrutamiento principal de VPC. Esta tabla de enrutamiento permite la comunicación entre todos los recursos que se implementan en la VPC. Sin embargo, no permite la comunicación con recursos que tienen direcciones IP que están fuera de los bloques CIDR asociados a la VPC. Puede asociar su propia tabla de enrutamiento a las subredes para cambiar este comportamiento. Para obtener más información, consulte Tablas de enrutamiento de subredes en la Guía del usuario de Amazon VPC.

b. Para ver las subredes actuales en su VPC.

```
aws ec2 describe-subnets --filters "Name=vpc-id,Values=$vpc_id" \
    --query 'Subnets[*].{SubnetId: SubnetId,AvailabilityZone:
 AvailabilityZone,CidrBlock: CidrBlock}' \
    --output table
```

Un ejemplo de salida sería el siguiente.

```
-------------------------------------------------------------------
|                          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        |
+------------------+--------------------+----------------------------+
```

Puede ver que las subredes en el bloque CIDR `192.168.1.0` que ha creado se encuentran en las mismas zonas de disponibilidad que las subredes del bloque CIDR `192.168.0.0`.

Paso 3: Configurar los recursos de Kubernetes

1. Establezca la variable de entorno de `AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG` a `true` en el DaemonSet de `aws-node`.

```
kubectl set env daemonset aws-node -n kube-system
 AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG=true
```

2. Recupere el ID del grupo de seguridad del clúster y guárdelo en una variable para utilizarlo en un paso posterior. Amazon EKS crea automáticamente este grupo de seguridad cuando crea el clúster.

```
cluster_security_group_id=$(aws eks describe-cluster --name my-custom-networking-
cluster --query cluster.resourcesVpcConfig.clusterSecurityGroupId --output text)
```

3. Cree un recurso personalizado `ENIConfig` para cada subred en la que desee programar pods.

   a. Cree un archivo único para cada configuración de interfaz de red.

   Los siguientes comandos crean archivos `ENIConfig` por separado de las dos subredes que se han creado en un paso anterior. El valor de `name` debe ser único. El nombre es el mismo que la zona de disponibilidad en la que se encuentra la subred. El grupo de seguridad del clúster está asignado al `ENIConfig`.

```
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
```

Para un clúster de producción, puede realizar los siguientes cambios en los comandos anteriores:

- Sustituya $cluster_security_group_id por el ID de un grupo de seguridad existente que quiera usar para cada `ENIConfig`.
- Recomendamos nombrar su `ENIConfigs` igual que la zona de disponibilidad para la que utilizará la `ENIConfig`, siempre que sea posible. Es posible que tenga que utilizar nombres diferentes para su `ENIConfigs` que los nombres de las zonas de disponibilidad por diversas razones. Por ejemplo, si tiene más de dos subredes en la misma zona de disponibilidad y desea utilizarlas con redes personalizadas, necesita varias `ENIConfigs` para la misma zona de disponibilidad. Dado que cada `ENIConfig` requiere un nombre único, no puede nombrar más de una de sus `ENIConfigs` utilizando el nombre de zona de disponibilidad.

  Si los nombres de `ENIConfig` no son todos iguales que los nombres de las zonas de disponibilidad, reemplace $az_1 y $az_2 por sus propios nombres en los comandos anteriores y anote los nodos con ENIConfig más adelante en este tutorial.

  > **Note**
  >
  > Si no especifica un grupo de seguridad válido para utilizarlo con un clúster de producción y utiliza:

- la versión `1.8.0` o posterior del complemento CNI de Amazon VPC para Kubernetes, se utilizan los grupos de seguridad asociados a la principal interfaz de redes elástica del nodo.

- una versión del complemento CNI de Amazon VPC para Kubernetes anterior a `1.8.0`, el grupo de seguridad predeterminado para la VPC se asignará a interfaces de red elásticas secundarias.

> **Important**
>
> - `AWS_VPC_K8S_CNI_EXTERNALSNAT=false` es un ajuste predeterminado de la configuración del complemento CNI de Amazon VPC para Kubernetes. Si utiliza la configuración predeterminada, el tráfico destinado a direcciones IP que no se encuentran dentro de uno de los bloques CIDR asociados a la VPC utiliza los grupos de seguridad y las subredes de la interfaz de la red principal del nodo. Las subredes y los grupos de seguridad definidos en las `ENIConfigs` que se utilizan para crear interfaces de red secundarias no se utilizan para este tráfico. Para obtener más información sobre esta configuración, consulte the section called “Tráfico de salida”.
> - Si también utiliza grupos de seguridad para pods, el grupo de seguridad especificado en `SecurityGroupPolicy` se utiliza en lugar del grupo de seguridad especificado en `ENIConfigs`. Para obtener más información, consulte the section called “Grupos de seguridad de pods”.

b. Aplique al clúster cada archivo de recursos personalizados que creó anteriormente con los siguientes comandos:

```
kubectl apply -f $az_1.yaml
kubectl apply -f $az_2.yaml
```

4. Confirmación de que se crearon `ENIConfigs`.

```
kubectl get ENIConfigs
```

Un ejemplo de salida sería el siguiente.

```
NAME         AGE
us-west-2a   117s
us-west-2d   105s
```

5. Si va a habilitar redes personalizadas en un clúster de producción y da un nombre a `ENIConfigs` que no sea el de la zona de disponibilidad para la que las utiliza, entonces pase al siguiente paso para implementar nodos de Amazon EC2.

Habilite Kubernetes para aplicar automáticamente la `ENIConfig` para una zona de disponibilidad en cualquier nuevo nodo de Amazon EC2 creado en su clúster.

a. Para ver el clúster de pruebas de este tutorial, vaya al paso siguiente.

   En el caso de un clúster de producción, compruebe si existe una anotación con la clave `k8s.amazonaws.com/eniConfig` para la variable de entorno `ENI_CONFIG_ANNOTATION_DEF` en la especificación del contenedor para el DaemonSet `aws-node`.

   ```
   kubectl describe daemonset aws-node -n kube-system | grep
    ENI_CONFIG_ANNOTATION_DEF
   ```

   Si se devuelve la salida, la anotación existe. Si no se devuelve ninguna salida, entonces la variable no se establece. Para un clúster de producción, puede utilizar esta configuración o la configuración del siguiente paso. Si utiliza este ajuste, anule el ajuste en el siguiente paso. En este tutorial, se utiliza la configuración del siguiente paso.

b. Actualice su DaemonSet `aws-node` para aplicar automáticamente la `ENIConfig` específica de la zona de disponibilidad en cualquier nuevo nodo de Amazon EC2 que se cree en su clúster.

   ```
   kubectl set env daemonset aws-node -n kube-system
    ENI_CONFIG_LABEL_DEF=topology.kubernetes.io/zone
   ```

Paso 4: Implementar nodos de Amazon EC2

1. Creación de un rol de IAM de nodo.

   a. Ejecute el siguiente comando para crear un archivo de política de confianza JSON de IAM.

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

```
  ]
}
```

b. Cree un rol de IAM y almacene el nombre de recurso de Amazon (ARN) devuelto en una variable para usarla en un paso posterior.

```
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. Adjunte al rol de IAM las tres políticas administradas por IAM necesarias.

```
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**
>
> Para simplificar este tutorial, la política AmazonEKS_CNI_Policy se adjunta al rol de IAM de nodo. Sin embargo, en un clúster de producción, recomendamos adjuntar la política a un rol de IAM independiente que se usa solo con el complemento CNI de Amazon VPC para Kubernetes. Para obtener más información, consulte the section called “Configuración para IRSA”.

2. Cree uno de los siguientes tipos de grupos de nodos. Para determinar el tipo de instancia que desea implementar, consulte the section called “Tipos de instancias de Amazon EC2”. Para este tutorial, complete la opción Administrado, Sin plantilla de lanzamiento o con plantilla de lanzamiento sin un ID de AMI especificado. Si planea utilizar el grupo de nodos para cargas de trabajo en entornos de producción, se recomienda familiarizarse con todas las opciones disponibles para grupos de nodos administrados y autoadministrados antes de proceder con su implementación.

   - Administrado: implemente el grupo de nodos mediante una de las siguientes opciones:

- Sin plantilla de lanzamiento o con plantilla de lanzamiento sin un ID de AMI especificado: complete el procedimiento indicado. Para este tutorial, utilice los valores de ejemplo. Para un grupo de nodos de producción, reemplace todos los valores de ejemplo con sus propios valores. El nombre del grupo de nodos no puede tener más de 63 caracteres. Debe empezar por una letra o un dígito, pero también puede incluir guiones y guiones bajos como caracteres no iniciales.

```
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
```

- Con una plantilla de lanzamiento con un ID de AMI especificado

  A. Determine el número máximo de pods que recomienda Amazon EKS para sus nodos. Siga las instrucciones que aparecen en Cantidad máxima de pods recomendada por Amazon EKS para cada tipo de instancia de Amazon EC2, con la adición de `--cni-custom-networking-enabled` al paso 3 de ese tema. Observe la salida de su uso en el siguiente paso.

  B. En la plantilla de lanzamiento, especifique un ID de AMI optimizado para Amazon EKS o una AMI personalizada creada a partir de la AMI optimizada para Amazon EKS y, luego, implemente el grupo de nodos mediante una plantilla de lanzamiento y proporcione los siguientes datos de usuario en la plantilla de lanzamiento. Estos datos de usuario pasan los argumentos a la especificación `NodeConfig`. Para obtener más información sobre NodeConfig, consulte la referencia de la API de NodeConfig. Puede sustituir `20` por el valor del paso anterior (recomendado) o por un valor propio.

```
---
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
```

Si ha creado una AMI personalizada, pero no a partir de la AMI optimizada de Amazon EKS, debe crear personalmente la configuración.

- Autoadministrado

  i. Determine el número máximo de pods que recomienda Amazon EKS para sus nodos. Siga las instrucciones de the section called “Número máximo de pods recomendado por Amazon EKS para cada tipo de instancia de Amazon EC2”, agregando `--cni-custom-networking-enabled` al paso tres de ese tema. Observe la salida de su uso en el siguiente paso.

  ii. Implemente el grupo de nodos con las instrucciones en the section called “Amazon Linux”.

> **Note**
>
> Si desea que los nodos de un clúster de producción admitan un número significativamente mayor de pods, ejecute el script en the section called “Número máximo de pods recomendado por Amazon EKS para cada tipo de instancia de Amazon EC2” de nuevo. Además, agregue la opción `--cni-prefix-delegation-enabled` al siguiente comando. Por ejemplo, se devuelve `110` para un tipo de instancia `m5.large`. Para obtener instrucciones sobre cómo habilitar esta capacidad, consulte the section called “Aumento de las direcciones IP”. Puede utilizar esta capacidad con redes personalizadas.

3. La creación de grupos de nodos tarda varios minutos. Puede comprobar el estado de la creación de un grupo de nodos administrados con el siguiente comando.

```
aws eks describe-nodegroup --cluster-name my-custom-networking-cluster --nodegroup-name my-nodegroup --query nodegroup.status --output text
```

   No continúe con el siguiente paso hasta que la salida devuelta sea `ACTIVE`.

4. Para ver el tutorial, puede omitir este paso.

   En caso de un clúster de producción, si no ha nombrado su `ENIConfigs` con el mismo nombre que la zona de disponibilidad para la que la utiliza, entonces debe anotar sus nodos con el nombre de `ENIConfig` que debe utilizarse con el nodo. Este paso no es necesario si tiene solo una subred en cada zona de disponibilidad y ha nombrado sus `ENIConfigs` con los mismos nombres

que dichas zonas. Esto se debe a que el complemento CNI de Amazon VPC para Kubernetes asocia automáticamente la `ENIConfig` correcta con el nodo cuando lo habilitó para esto en un paso anterior.

a. Obtenga la lista de nodos del clúster.

```
kubectl get nodes
```

Un ejemplo de salida sería el siguiente.

```
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. Determine en qué zona de disponibilidad se encuentra cada nodo. Ejecute el siguiente comando para cada nodo que se devolvió en el paso anterior y reemplace las direcciones IP en función del resultado anterior.

```
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}'
```

Un ejemplo de salida sería el siguiente.

```
[
    {
        "AvailabilityZone": "us-west-2d",
        "SubnetId": "subnet-Example5"
    }
]
```

c. Anote cada nodo con la `ENIConfig` que creó para el ID de subred y la zona de disponibilidad. Solo puede anotar un nodo con una `ENIConfig`, aunque se pueden anotar varios nodos con la misma `ENIConfig`. Sustituya los valores de ejemplo por sus propios valores.

```
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. Si tenía nodos en un clúster de producción con pods en ejecución antes de comenzar a utilizar la característica de redes personalizadas, lleve a cabo las siguientes tareas:
   a. Asegúrese de tener nodos disponibles que utilizan la característica de red personalizada.
   b. Acordone y drene los nodos para apagar correctamente los pods. Para obtener más información, consulte Drene un nodo de manera segura en la documentación de Kubernetes.
   c. Termine los nodos. Si los nodos se encuentran en un grupo de nodos administrado existente, puede eliminar el grupo de nodos. Ejecute el siguiente comando.

   ```
   aws eks delete-nodegroup --cluster-name my-custom-networking-cluster --nodegroup-
   name my-nodegroup
   ```

   Solo los nodos nuevos que se registran con la etiqueta `k8s.amazonaws.com/eniConfig` utilizan la característica de redes personalizadas.
6. Confirme que los pods estén asignados en una dirección IP de un bloque de CIDR asociado a una de las subredes que creó en un paso anterior.

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

   Un ejemplo de salida sería el siguiente.

   ```
   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>
```

Puede ver que a los pods de CoreDNS se les asignan direcciones IP desde el bloque de CIDR `192.168.1.0` que ha agregado a la VPC. Sin redes personalizadas, se les habrían asignado direcciones desde el bloque CIDR `192.168.0.0`, porque era el único bloque CIDR asociado originalmente a la VPC.

Si la `spec` de un pod contiene `hostNetwork=true`, se le asigna la dirección IP principal del nodo. No se le asigna una dirección de las subredes que ha agregado. De forma predeterminada, este valor se establece en `false`. Este valor se establece en `true` para `kube-proxy` y los pods del complemento CNI de Amazon VPC para Kubernetes (`aws-node`) que se ejecutan en su clúster. Esta es la razón por la que a `kube-proxy` y a los pods de `aws-node` del complemento no se les asignan direcciones 192.168.1.x en la salida anterior. Para obtener más información acerca de la configuración de `hostNetwork` del pod, consulte PodSpec v1 core en la referencia de API de Kubernetes.

Paso 5: eliminar recursos del tutorial

Una vez completado el tutorial, le recomendamos que elimine los recursos que creó. Puede ajustar los pasos para habilitar las redes personalizadas para un clúster de producción.

1. Si el grupo de nodos que creó fue solo para pruebas, bórrelo.

```
aws eks delete-nodegroup --cluster-name my-custom-networking-cluster --nodegroup-name
 my-nodegroup
```

2. Incluso después de que la salida de la AWS CLI indica que el clúster se ha eliminado, es posible que el proceso de eliminación no esté completo. El proceso de eliminación tarda unos minutos. Confirme que se ha completado al ejecutar el siguiente comando.

```
aws eks describe-nodegroup --cluster-name my-custom-networking-cluster --nodegroup-
name my-nodegroup --query nodegroup.status --output text
```

No continúe hasta que la salida devuelta sea similar a la siguiente.

```
An error occurred (ResourceNotFoundException) when calling the DescribeNodegroup
 operation: No node group found for name: my-nodegroup.
```

3. Si el grupo de nodos que creó fue solo para pruebas, elimine el rol de IAM de nodo.

   a. Separe la política del rol.

   ```
   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. Elimine el rol.

   ```
   aws iam delete-role --role-name myCustomNetworkingNodeRole
   ```

4. Elimine el clúster.

   ```
   aws eks delete-cluster --name my-custom-networking-cluster
   ```

   Confirme que el clúster se elimine con el siguiente comando.

   ```
   aws eks describe-cluster --name my-custom-networking-cluster --query cluster.status
    --output text
   ```

   Cuando se devuelve un resultado similar al siguiente, el clúster se eliminará correctamente.

   ```
   An error occurred (ResourceNotFoundException) when calling the DescribeCluster
    operation: No cluster found for name: my-custom-networking-cluster.
   ```

5. Elimine el rol de IAM del clúster.

   a. Separe la política del rol.

   ```
   aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSClusterRole --
   policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy
   ```

   b. Elimine el rol.

```
aws iam delete-role --role-name myCustomNetworkingAmazonEKSClusterRole
```

6. Elimine las subredes que creó en un paso anterior.

```
aws ec2 delete-subnet --subnet-id $new_subnet_id_1
aws ec2 delete-subnet --subnet-id $new_subnet_id_2
```

7. Elimine la VPC que ha creado.

```
aws cloudformation delete-stack --stack-name my-eks-custom-networking-vpc
```

Asignación de más direcciones IP a los nodos de Amazon EKS con prefijos

Aplicación: en nodos de Linux y Windows con instancias de Amazon EC2

Aplicación: en subredes públicas y privadas

Cada instancia de Amazon EC2 admite una cantidad máxima de interfaces de red elásticas y una cantidad máxima de direcciones IP que se pueden asignar a cada interfaz de red. Cada nodo requiere una dirección IP para cada interfaz de red. Se pueden asignar todas las demás direcciones IP disponibles a `Pods`. Cada uno `Pod` requiere su propia dirección IP. Como resultado, es posible que tenga nodos que tengan recursos informáticos y de memoria disponibles, pero que no puedan acomodar `Pods` adicionales porque el nodo se quedó sin direcciones IP para asignar a `Pods`.

Puede aumentar la cantidad de direcciones IP que los nodos pueden asignar a `Pods` mediante la asignación de prefijos de IP, en lugar de asignar direcciones IP secundarias individuales a sus nodos. Cada prefijo incluye varias direcciones IP. Si no configura su clúster para la asignación de prefijos IP, deberá hacer más llamadas desde su clúster a la interfaz de programación de aplicaciones (API) de Amazon EC2 para configurar las interfaces de red y las direcciones IP necesarias para la conectividad de los pods. A medida que los clústeres crecen a tamaños más grandes, la frecuencia de estas llamadas a la API puede generar tiempos de lanzamiento de instancias y pods más prolongados. Esto causa retrasos en el escalado para satisfacer la demanda de cargas de trabajo grandes y exigentes, y agrega costos y gastos generales de administración, ya que necesita aprovisionar clústeres y VPC adicionales para cumplir con los requisitos de escalado. Para obtener más información, consulte Umbrales de escalabilidad de Kubernetes en GitHub.

Compatibilidad con las características del complemento CNI de Amazon VPC para Kubernetes

Puede utilizar prefijos de IP con las siguientes características:

- Traducción de direcciones de red de origen IPv4 (para obtener más información, consulte the section called “Tráfico de salida”).
- Direcciones IPv6 para clústeres, pods y servicios (para obtener más información, consulte the section called “IPv6”).
- Restricción del tráfico con políticas de red de Kubernetes. Para obtener más información, consulte the section called “Políticas de Kubernetes”.

La siguiente lista proporciona información sobre la configuración del complemento de CNI de Amazon VPC que se aplica. Para obtener más información acerca de cada configuración, consulte amazon-vpc-cni-k8s en GitHub.

- `WARM_IP_TARGET`
- `MINIMUM_IP_TARGET`
- `WARM_PREFIX_TARGET`

Consideraciones

Cuando utilice esta característica, tenga en cuenta lo siguiente:

- Cada tipo de instancia de Amazon EC2 admite una cantidad máxima de pods. Si el grupo de nodos administrado consta de varios tipos de instancias, el número de pods máximo menor de una instancia del clúster se aplica a todos los nodos del clúster.
- De forma predeterminada, el número máximo de `Pods` que puede ejecutar en un nodo es 110, pero puede cambiar ese número. Si cambia el número y tiene un grupo de nodos administrado existente, la siguiente AMI o la actualización de la plantilla de lanzamiento de su grupo de nodos generará nuevos nodos con el valor modificado.
- Al pasar de la asignación de direcciones IP a la asignación de prefijos de IP, le recomendamos que cree nuevos grupos de nodos para aumentar la cantidad de direcciones IP disponibles, en lugar de realizar un reemplazo gradual de los nodos existentes. La ejecución de pods en un nodo que tiene direcciones IP y prefijos asignados puede generar incoherencias en la capacidad de direcciones IP anunciada, lo que afecta a las futuras cargas de trabajo en el nodo. Para conocer la forma recomendada de efectuar la transición, consulte Prefix Delegation mode for Linux en la Guía de prácticas recomendadas de Amazon EKS.
- El alcance del grupo de seguridad se encuentra en el nivel del nodo. Para obtener más información, consulte Grupo de seguridad.

- Los prefijos de IP asignados a una interfaz de red admiten una alta densidad de pods por nodo y tienen el mejor tiempo de lanzamiento.
- Los prefijos IP y las direcciones IP están asociados a las interfaces de red elásticas estándar de Amazon EC2. A los pods que requieren grupos de seguridad específicos se les asigna la dirección IP principal de una interfaz de red de ramificación. Puede mezclar pods que obtengan direcciones IP o direcciones IP de prefijos IP con pods que obtengan interfaces de red de ramificación en el mismo nodo.
- Solo para clústeres con nodos Linux.
  - Luego de configurar el complemento para asignar prefijos a las interfaces de red, no podrá degradar el complemento CNI de Amazon VPC para Kubernetes a una versión anterior a `1.9.0` (o `1.10.1`) sin eliminar todos los nodos de todos los grupos de nodos en el clúster.
  - Si también utiliza grupos de seguridad para pods, con `POD_SECURITY_GROUP_ENFORCING_MODE=standard` y `AWS_VPC_K8S_CNI_EXTERNALSNAT=false`, cuando los pods se comunican con los puntos de conexión fuera de su VPC, se utilizan los grupos de seguridad del nodo, en lugar de cualquier grupo de seguridad que haya asignado a sus pods.

    Si también utiliza grupos de seguridad para pods, con `POD_SECURITY_GROUP_ENFORCING_MODE=strict`, cuando los `Pods` se comunican con los puntos de conexión fuera de la VPC, se utilizan los grupos de seguridad de `Pod's`.

Aumento de las direcciones IP disponibles para su nodo de Amazon EKS

Puede aumentar la cantidad de direcciones IP que los nodos pueden asignar a pods mediante la asignación de prefijos de IP, en lugar de asignar direcciones IP secundarias individuales a sus nodos.

Requisitos previos

- Necesita un clúster existente. Para implementar uno, consulte the section called “Creación de un clúster ”.
- Las subredes en las que se encuentran sus nodos de Amazon EKS deben tener suficientes bloques contiguos `/28` (para clústeres `IPv4`) o `/80` (para clústeres `IPv6`) enrutamiento entre dominios sin clases (CIDR). Solo puede tener nodos Linux en un clúster `IPv6`. El uso de prefijos IP puede fallar si las direcciones IP están dispersas por toda la subred de CIDR. Le recomendamos lo siguiente:

  - Utilizar una reserva de CIDR de subred para que, aunque se siga utilizando alguna dirección IP dentro del rango reservado, una vez publicada, no se reasignen las direcciones IP. Esto garantiza que los prefijos estén disponibles para su asignación sin segmentación.
  - Utilice nuevas subredes que se usen específicamente para ejecutar las cargas de trabajo a las que se asignan los prefijos IP. Tanto las cargas de trabajo de Windows como las de Linux pueden ejecutarse en la misma subred cuando se asignan prefijos de IP.
- Para asignar prefijos de IP a sus nodos, sus nodos deben estar basados en AWS Nitro. Las instancias que no se basan en Nitro continúan asignando direcciones IP secundarias individuales, pero tienen un número significativamente menor de direcciones IP para asignar a los pods que las instancias basadas en Nitro.
- Solo para clústeres con nodos de Linux: si su clúster está configurado para la familia `IPv4`, debe tener instalada la versión `1.9.0` o posterior del complemento CNI de Amazon VPC para Kubernetes. Puede comprobar su versión actual con el siguiente comando.

  ```
  kubectl describe daemonset aws-node --namespace kube-system | grep Image | cut -d "/"
   -f 2
  ```

  Si su clúster está configurado para la familia `IPv6`, debe tener instalada la versión `1.10.1` o del complemento. Si la versión de su complemento es anterior a las versiones requeridas, debe actualizarlo. Para obtener más información, consulte las secciones de actualización del artículo Asignación de direcciones IP a pods con CNI de Amazon VPC.
- Solo para clústeres con nodos de Linux
  - Debe tener la compatibilidad con Windows habilitada para el clúster. Para obtener más información, consulte the section called “Habilitación de la compatibilidad con Windows”.

Asignación de prefijos de direcciones IP a los nodos

Configure el clúster para asignar prefijos de direcciones IP a los nodos. Complete el procedimiento que coincida con el sistema operativo de su nodo.

Linux

1. Habilite el parámetro a fin de asignar prefijos a las interfaces de red para el DaemonSet de CNI de Amazon VPC. Cuando implementa un clúster, la versión `1.10.1` o posterior del complemento CNI de Amazon VPC para Kubernetes se implementa con él. Si ha creado el clúster con la familia

IPv6, este ajuste se configuró en true de forma predeterminada. Si ha creado el clúster con la familia IPv4, este ajuste se configuró en false de forma predeterminada.

```
kubectl set env daemonset aws-node -n kube-system ENABLE_PREFIX_DELEGATION=true
```

> **Important**
>
> Incluso si la subred tiene direcciones IP disponibles, si la subred no tiene disponible ningún bloque /28 contiguo, verá el siguiente error en los registros del complemento CNI de Amazon VPC para Kubernetes.
>
> ```
> InsufficientCidrBlocks: The specified subnet does not have enough free cidr
>  blocks to satisfy the request
> ```
>
> Esto puede ocurrir debido a la fragmentación de las direcciones IP secundarias existentes distribuidas por una subred. Para resolver este error, cree una nueva subred y lance pods allí, o utilice una reserva de CIDR de subred de Amazon EC2 para reservar espacio dentro de una subred para utilizarla con la asignación de prefijos. Para obtener más información, consulte Reservas de la subred de CIDR en la Guía del usuario de Amazon VPC.

2. Si planea implementar un grupo de nodos administrado sin una plantilla de lanzamiento o con una plantilla de lanzamiento en la que no ha especificado un ID de AMI, y utiliza una versión del complemento CNI de Amazon VPC para Kubernetes igual o posterior a las versiones enumeradas en los requisitos previos, continúe con el siguiente paso. Los grupos de nodos administrados calculan automáticamente el número máximo de pods.

   Si va a implementar un grupo de nodos autoadministrado o un grupo de nodos administrado con una plantilla de lanzamiento en la que ha especificado un ID de AMI, debe determinar el número máximo de pods recomendados por Amazon EKS para los nodos. Siga las instrucciones que aparecen en Cantidad máxima de pods recomendada por Amazon EKS para cada tipo de instancia de Amazon EC2, y con la adición de --cni-prefix-delegation-enabled al paso 3. Observe la salida de su uso en un paso posterior.

   > **Important**
   >
   > Los grupos de nodos administrados aplican un número máximo en el valor maxPods. Para las instancias con menos de 30 vCPU, el número máximo es 110 y para todas las demás

instancias el número máximo es 250. Este número máximo se aplica independientemente de si la delegación de prefijos está habilitada o no.

3. Si utiliza un clúster configurado para `IPv6`, continúe con el siguiente paso.

   Especifique los parámetros en una de las siguientes opciones. Para determinar qué opción es la adecuada y qué valor debe proporcionarle, consulte WARM_PREFIX_TARGET, WARM_IP_TARGET, and MINIMUM_IP_TARGET em GitHub.

   Puede reemplazar example values por un valor mayor a cero.

   - `WARM_PREFIX_TARGET`

     ```
     kubectl set env ds aws-node -n kube-system WARM_PREFIX_TARGET=1
     ```

   - `WARM_IP_TARGET` o `MINIMUM_IP_TARGET`: si se establece este valor, sustituye a cualquier valor establecido para `WARM_PREFIX_TARGET`.

     ```
     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. Cree uno de los siguientes tipos de grupos de nodos con al menos un tipo de instancia Nitro de Amazon Linux 2023 de Amazon EC2. A fin de obtener una lista de los tipos de instancias de Nitro, consulte Instancias integradas en el sistema Nitro en la Guía del usuario de Amazon EC2. Esta capacidad no es compatible con Windows. En el caso de las opciones que incluyen *`110`*, reemplácelo por el valor del paso tres (recomendado) o un valor propio.

   - Autoadministrado: implementa el grupo de nodos según las instrucciones que aparecen en Creación de nodos autoadministrados de Amazon Linux. Antes de crear la pila de CloudFormation, abra el archivo de plantilla y ajuste `UserData` en la `NodeLaunchTemplate` para que sea como se muestra a continuación.

     ```
     ...
                 apiVersion: node.eks.aws/v1alpha1
                 kind: NodeConfig
                 spec:
                   cluster:
                     name: ${ClusterName}
                     apiServerEndpoint: ${ApiServerEndpoint}
                     certificateAuthority: ${CertificateAuthorityData}
     ```

```
                cidr: ${ServiceCidr}
              kubelet:
                config:
                  maxPods: 110
...
```

Si utiliza `eksctl` para crear el grupo de nodos, utilice el siguiente comando.

```
eksctl create nodegroup --cluster my-cluster --managed=false --max-pods-per-node
 110
```

- Administrado: implemente el grupo de nodos mediante una de las siguientes opciones:
  - Sin una plantilla de lanzamiento o con una plantilla de lanzamiento sin un ID de AMI especificado: complete el procedimiento indicado en Creación de un grupo de nodos administrados para un clúster. Los grupos de nodos administrados calculan automáticamente el valor `max-pods` recomendados por Amazon EKS.
  - Con una plantilla de lanzamiento con un ID de AMI especificado: en la plantilla de lanzamiento, especifique un ID de AMI optimizada para Amazon EKS o una AMI personalizada creada a partir de la AMI optimizada para Amazon EKS y, a continuación, implemente el grupo de nodos mediante una plantilla de lanzamiento y proporcione los siguientes datos de usuario en la plantilla de lanzamiento. Estos datos de usuario se transfieren a un objeto `NodeConfig` para que la herramienta `nodeadm` los lea en el nodo. Para obtener más información acerca de `nodeadm`, consulte la documentación de nodeadm.

    ```
    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`
--//--
```

Si utiliza `eksctl` para crear el grupo de nodos, utilice el siguiente comando.

```
eksctl create nodegroup --cluster my-cluster --max-pods-per-node 110
```

Si ha creado una AMI personalizada, pero no a partir de la AMI optimizada de Amazon EKS, debe crear personalmente la configuración.

> **Note**
>
> Si también desea asignar direcciones IP a pods de una subred diferente a la de la instancia, debe habilitar la capacidad en este paso. Para obtener más información, consulte the section called “Redes personalizadas”.

Windows

1. Habilite la asignación de prefijos IP.
   a. Abra el `ConfigMap` de `amazon-vpc-cni` para editar.

   ```
   kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
   ```

   b. Añada la siguiente línea a la sección `data`:

   ```
     enable-windows-prefix-delegation: "true"
   ```

   c. Guarde el archivo y cierre el editor.
   d. Confirme que la línea se agregó a `ConfigMap`.

   ```
   kubectl get configmap -n kube-system amazon-vpc-cni -o "jsonpath={.data.enable-
   windows-prefix-delegation}"
   ```

   Si la salida devuelta no es `true`, es posible que haya ocurrido un error. Intente completar el paso de nuevo.

> **Important**
>
> Incluso si la subred tiene direcciones IP disponibles, si la subred no tiene disponible ningún bloque /28 contiguo, verá el siguiente error en los registros del complemento CNI de Amazon VPC para Kubernetes.
>
> ```
> InsufficientCidrBlocks: The specified subnet does not have enough free cidr
>  blocks to satisfy the request
> ```
>
> Esto puede ocurrir debido a la fragmentación de las direcciones IP secundarias existentes distribuidas por una subred. Para resolver este error, cree una nueva subred y lance pods allí, o utilice una reserva de CIDR de subred de Amazon EC2 para reservar espacio dentro de una subred para utilizarla con la asignación de prefijos. Para obtener más información, consulte Reservas de la subred de CIDR en la Guía del usuario de Amazon VPC.

2. (Opcional) Especifique una configuración adicional para controlar el comportamiento de escalado previo y dinámico del clúster. Para obtener más información, consulte Opciones de configuración con modo de delegación de prefijo en Windows en GitHub.

   a. Abra el `ConfigMap` de `amazon-vpc-cni` para editar.

   ```
   kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
   ```

   b. Sustituya los valores de ejemplo por un valor superior a cero y agregue las entradas que necesite a la sección `data` de `ConfigMap`. Si establece un valor para `warm-ip-target` o `minimum-ip-target`, el valor anula cualquier valor establecido para `warm-prefix-target`.

   ```
   warm-prefix-target: "1"
   warm-ip-target: "5"
   minimum-ip-target: "2"
   ```

   c. Guarde el archivo y cierre el editor.

3. Cree grupos de nodos de Windows con al menos un tipo de instancia Nitro de Amazon EC2. A fin de obtener una lista de los tipos de instancias de Nitro, consulte Instancias integradas en el sistema Nitro en la Guía del usuario de Amazon EC2. De forma predeterminada, la cantidad máxima de pods que puede implementar en un nodo es 110. Si desea aumentar o disminuir

ese número, especifique lo siguiente en los datos de usuario para la configuración de arranque. Reemplace *max-pods-quantity* con su valor máximo de pods.

```
-KubeletExtraArgs '--max-pods=max-pods-quantity'
```

Si está implementando grupos de nodos administrados, esta configuración debe agregarse en la plantilla de lanzamiento. Para obtener más información, consulte the section called “Plantillas de inicialización”. Para obtener más información sobre los parámetros de configuración para el script de arranque de Windows, consulte the section called “Parámetros de configuración del script de arranque”.

Cómo determinar el número máximo de pods y las direcciones IP disponibles

1. Una vez que se implementan los nodos, consulte los nodos del clúster.

   ```
   kubectl get nodes
   ```

   Un ejemplo de salida sería el siguiente.

   ```
   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. Describa uno de los nodos para determinar el valor de `max-pods` para el nodo y el número de direcciones IP disponibles. Reemplace *192.168.30.193* con la dirección `IPv4` en el nombre de uno de sus nodos devueltos en la salida anterior.

   ```
   kubectl describe node ip-192-168-30-193.region-code.compute.internal | grep 'pods\|
   PrivateIPv4Address'
   ```

   Un ejemplo de salida sería el siguiente.

   ```
   pods:                                110
   vpc.amazonaws.com/PrivateIPv4Address:  144
   ```

En el resultado anterior, `110` es el número máximo de pods que Kubernetes implementará en el nodo, aunque haya *`144`* direcciones IP disponibles.

Asignación de los grupos de seguridad a pods individuales

Aplicación: en nodos de Linux con instancias de Amazon EC2

Aplicación: en subredes privadas

Los grupos de seguridad para pods integran los grupos de seguridad de Amazon EC2 con los pods de Kubernetes. Puede utilizar grupos de seguridad de Amazon EC2 para definir reglas que permitan el tráfico de red entrante y saliente hacia y desde los pods que implemente en nodos que se ejecutan en muchos tipos de instancias de Amazon EC2 y Fargate. Para obtener una explicación más detallada de esta capacidad, consulte la publicación de blog Presentación de los grupos de seguridad de pods.

Compatibilidad con las características del complemento CNI de Amazon VPC para Kubernetes

Puede usar grupos de seguridad para pods con las siguientes características:

- Traducción de direcciones de red de origen IPv4 (para obtener más información, consulte) the section called “Tráfico de salida”.
- Direcciones IPv6 para clústeres, pods y servicios (para obtener más información, consulte) the section called “IPv6”.
- Restricción del tráfico con políticas de red de Kubernetes. Para obtener más información, consulte the section called “Políticas de Kubernetes”.

Consideraciones

Antes de implementar grupos de seguridad para pods, tenga en cuenta las siguientes limitaciones y condiciones:

- Los grupos de seguridad de pods no se pueden utilizar con nodos de Windows ni el modo automático de EKS.
- Los grupos de seguridad para pods pueden utilizarse en clústeres configurados para la familia `IPv6` que contengan nodos de Amazon EC2 mediante la versión 1.16.0 o posterior del complemento CNI de Amazon VPC. Puede usar grupos de seguridad para pods con clústeres

configurados para la familia IPv6 que contengan solo nodos de Fargate mediante la versión 1.7.7 o posterior del complemento CNI de Amazon VPC. Para obtener más información, consulte the section called “IPv6”

- Los grupos de seguridad para pods son compatibles con la mayoría de las familias de instancias de Amazon EC2 basadas en Nitro, aunque no en todas las generaciones de una familia. Por ejemplo, las generaciones y familia de instancias m5, c5, r5, m6g, c6g y r6g son compatibles. No se admite ningún tipo de instancia de la familia t. Para obtener una lista completa de los tipos de instancias compatibles, consulte el archivo limits.go en GitHub. Sus nodos deben ser uno de los tipos de instancias enumerados que tienen IsTrunkingCompatible: true en ese archivo.
- Si utiliza conjuntamente redes personalizadas y grupos de seguridad de pods, se utiliza el grupo de seguridad especificado por los grupos de seguridad de pods en lugar del grupo de seguridad especificado en ENIConfig.
- Si utiliza la versión 1.10.2 o anterior del complemento CNI de Amazon VPC e incluye la configuración de terminationGracePeriodSeconds en la especificación del pod, el valor de la configuración no puede ser cero.
- Si utiliza la versión 1.10 o anterior del complemento CNI de Amazon VPC o la versión 1.11 con POD_SECURITY_GROUP_ENFORCING_MODE=strict, que es la configuración predeterminada, entonces los servicios de Kubernetes de tipo NodePort y LoadBalancer con destinos de instancia con un conjunto externalTrafficPolicy establecido en Local no son compatibles con los pods a los que asigna grupos de seguridad. Para obtener más información sobre el uso de un equilibrador de carga con destinos de instancia, consulte the section called “Equilibrio de carga de red”.
- Si utiliza la versión 1.10 o anterior del complemento CNI de Amazon VPC o la versión 1.11 con POD_SECURITY_GROUP_ENFORCING_MODE=strict, que es la configuración predeterminada, el NAT de origen está deshabilitado para el tráfico saliente de pods con grupos de seguridad asignados a fin de que se apliquen las reglas de grupo de seguridad salientes. Para acceder a Internet, los pods con grupos de seguridad asignados deben lanzarse en nodos que se implementen en una subred privada configurada con una instancia o puerta de enlace de NAT. Los pods con grupos de seguridad asignados implementados en subredes públicas no pueden acceder a Internet.

  Si utiliza la versión 1.11 o posterior del plugin con POD_SECURITY_GROUP_ENFORCING_MODE=standard, el tráfico del pod destinado para salir de la VPC se traduce a la dirección IP de la interfaz de red principal de la instancia. Para este tráfico, se utilizan las reglas de los grupos de seguridad de la interfaz de red principal, en lugar de las reglas de los grupos de seguridad de los pods.

- Para utilizar la política de red de Calico con pods que tienen grupos de seguridad asociados, debe utilizar la versión `1.11.0` o posterior del complemento CNI de Amazon VPC y establecerla en `POD_SECURITY_GROUP_ENFORCING_MODE=standard`. De otro modo, el flujo de tráfico hacia y desde los pods con grupos de seguridad asociados no están sujetos al cumplimiento de la política de red de Calico y solo se limitan a la aplicación de grupos de seguridad de Amazon EC2. Para actualizar la versión del CNI de Amazon VPC, consulte the section called “CNI de Amazon VPC”
- Los pods que funcionan en nodos de Amazon EC2 y que utilizan grupos de seguridad en clústeres y NodeLocal DNSCache solo se admiten con la versión `1.11.0` o posterior del complemento CNI de Amazon VPC y con `POD_SECURITY_GROUP_ENFORCING_MODE=standard`. Para actualizar la versión del complemento CNI de Amazon VPC, consulte the section called “CNI de Amazon VPC”
- Los grupos de seguridad para pods pueden generar un aumento en la latencia de inicio de un pod en el caso de los pods con alta pérdida. Esto se debe a la limitación de velocidad en el controlador de recursos.
- El alcance del grupo de seguridad de EC2 se encuentra en el pod. Para obtener más información, consulte Grupo de seguridad.

  Si configuró `POD_SECURITY_GROUP_ENFORCING_MODE=standard` y `AWS_VPC_K8S_CNI_EXTERNALSNAT=false`, el tráfico destinado a puntos de conexión fuera de la VPC utiliza los grupos de seguridad del nodo, no los grupos de seguridad del pod.

Configuración del complemento CNI de Amazon VPC para Kubernetes para grupos de seguridad de pods de Amazon EKS

Si utiliza pods con instancias de Amazon EC2, debe configurar el complemento CNI de Amazon VPC con los grupos de seguridad.

Si solo utiliza pods de Fargate y no tiene nodos de Amazon EC2 en su clúster, consulte the section called “SecurityGroupPolicy”.

1. Verifique su versión actual del complemento CNI de Amazon VPC para Kubernetes con el siguiente comando:

   ```
   kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni: |
    cut -d : -f 3
   ```

   Un ejemplo de salida sería el siguiente.

```
v1.7.6
```

Si su versión del complemento CNI de Amazon VPC para Kubernetes es anterior a la `1.7.7`, actualice el complemento a la versión `1.7.7` o posterior. Para obtener más información, consulte the section called “CNI de Amazon VPC”

2. Agregue la política de IAM administrada AmazonEKSVPCResourceController al rol de clúster asociado al clúster de Amazon EKS. La política permite que el rol administre las interfaces de red, sus direcciones IP privadas y su vinculación y desvinculación desde y hacia las instancias de red.

   a. Recupere el nombre del rol de IAM de su clúster y guárdelo en una variable. Reemplace *my-cluster* por el nombre de su clúster.

   ```
   cluster_role=$(aws eks describe-cluster --name my-cluster --query cluster.roleArn
    --output text | cut -d / -f 2)
   ```

   b. Asocie la política de al rol.

   ```
   aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
   AmazonEKSVPCResourceController --role-name $cluster_role
   ```

3. Habilitación del complemento CNI de Amazon VPC para administrar las interfaces de red de los pods al establecer la variable `ENABLE_POD_ENI` en `true` en el DaemonSet `aws-node`. Una vez que esta configuración se establece en `true`, el complemento crea un recurso personalizado `cninode` para cada nodo del clúster. El controlador de recursos de VPC crea y adjunta una interfaz de red especial denominada interfaz de red troncal con la descripción `aws-k8s-trunk-eni`.

   ```
   kubectl set env daemonset aws-node -n kube-system ENABLE_POD_ENI=true
   ```

   > **Note**
   >
   > La interfaz de red troncal se incluye en el número máximo de interfaces de red que admite el tipo de instancia. A fin de obtener una lista del número máximo de interfaces de red que admite cada tipo de instancia, consulte Direcciones IP por interfaz de red por tipo de instancia en la Guía del usuario de Amazon EC2. Si su nodo ya cuenta con el número máximo de interfaces de red estándar adjuntas, el controlador de recursos de VPC reservará un espacio. Tendrá que reducir verticalmente los pods en ejecución lo suficiente

para que el controlador desconecte y elimine una interfaz de red estándar, cree la interfaz de red troncal y la adjunte a la instancia.

4. Si desea ver cuál de sus nodos tiene un recurso personalizado `CNINode` establecido, utilice el siguiente comando. Si se devuelve `No resources found`, espere varios segundos e inténtelo de nuevo. El paso anterior requiere reiniciar los pods del complemento CNI de Amazon VPC para Kubernetes, lo cual tarda varios segundos.

```
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"}]
```

Si utiliza versiones de CNI de VPC anteriores a `1.15`, se utilizaron etiquetas de nodo en lugar del recurso personalizado `CNINode`. Si desea ver cuál de sus nodos tienen la etiqueta de nodo `aws-k8s-trunk-eni` establecida en `true`, utilice el siguiente comando. Si se devuelve `No resources found`, espere varios segundos e inténtelo de nuevo. El paso anterior requiere reiniciar los pods del complemento CNI de Amazon VPC para Kubernetes, lo cual tarda varios segundos.

```
kubectl get nodes -o wide -l vpc.amazonaws.com/has-trunk-attached=true
```

Una vez que se crea la interfaz de red troncal, se pueden asignar direcciones IP secundarias a los pods desde las interfaces de red troncales o estándar. La interfaz troncal se elimina de forma automática si se elimina el nodo.

Cuando implementa un grupo de seguridad para un pod en un paso posterior, el controlador de recursos de VPC crea una interfaz de red especial denominada interfaz de red de ramificación con una descripción de `aws-k8s-branch-eni` y les asocia los grupos de seguridad. Se crean las interfaces de red de ramificación además de las interfaces de red estándar y troncal adjuntas al nodo.

Si utiliza sondeos de estado o preparación, también necesita desactivar el demux temprano de TCP, de modo que el `kubelet` pueda conectarse a los pods en las interfaces de red de ramificación a través de TCP. Para desactivar el demux temprano de TCP, ejecute el siguiente comando:

```
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**
>
> Si utiliza la versión `1.11.0` o posterior del complemento CNI de Amazon VPC para Kubernetes y lo establece en `POD_SECURITY_GROUP_ENFORCING_MODE=standard`, como se describe en el siguiente paso, no es necesario ejecutar el comando anterior.

5. Si su clúster usa `NodeLocal DNSCache` o desea usar la política de red de Calico con los pods que tienen sus propios grupos de seguridad, o si tiene servicios de Kubernetes de tipo `NodePort` y `LoadBalancer` mediante los destinos de instancia con una `externalTrafficPolicy` establecida en `Local` para pods a los que desea asignar grupos de seguridad, debe usar la versión `1.11.0` o posterior del complemento CNI de Amazon VPC para Kubernetes y habilitar la siguiente configuración:

   ```
   kubectl set env daemonset aws-node -n kube-system
    POD_SECURITY_GROUP_ENFORCING_MODE=standard
   ```

   IMPORTANTE: Las reglas del grupo de seguridad del pod no se aplican al tráfico entre pods o entre pods y servicios, como **`kubelet`** o **`nodeLocalDNS`**, que se encuentran en el mismo nodo. Los pods que utilizan diferentes grupos de seguridad en el mismo nodo no pueden comunicarse porque están configurados en diferentes subredes y el enrutamiento está deshabilitado entre estas subredes. El tráfico saliente de los pods a direcciones fuera de la VPC es la dirección de red traducida a la dirección IP de la interfaz de red principal de la instancia (a menos que también haya configurado `AWS_VPC_K8S_CNI_EXTERNALSNAT=true`). Para este tráfico, se utilizan las reglas de los grupos de seguridad de la interfaz de red principal, en lugar de las reglas de los grupos de seguridad de los pods. ** Para que esta configuración se aplique a los pods existentes, debe reiniciar los pods o los nodos en que se están ejecutando los pods.

6. Para ver cómo utilizar una política de grupo de seguridad para su pods, consulte the section called “SecurityGroupPolicy”.

Uso de una política de grupo de seguridad para un pod de Amazon EKS

Para usar grupos de seguridad con pods, debe tener un grupo de seguridad existente. En los siguientes pasos se muestra cómo utilizar la política de grupo de seguridad para un pod. A menos

que se indique lo contrario, complete todos los pasos del mismo terminal ya que en los siguientes pasos se utilizan variables que no persisten en los terminales.

Si tiene un pod con instancias de Amazon EC2, debe configurar el complemento antes de utilizar este procedimiento. Para obtener más información, consulte the section called “Configuración”.

1. Cree un espacio de nombres de Kubernetes en el que implementar los recursos de . Puede reemplazar *my-namespace* por el nombre del espacio de nombres que desee usar.

```
kubectl create namespace my-namespace
```

2. Implemente una SecurityGroupPolicy de Amazon EKS a su clúster.
   a. Copie los siguientes contenidos en su dispositivo. Puede reemplazar *podSelector* por serviceAccountSelector si prefiere seleccionar pods en función de las etiquetas de cuenta de servicio. Debe especificar un selector o el otro. Un podSelector vacío (ejemplo: podSelector: {}) selecciona todos los pods del espacio de nombres. Puede cambiar *my-role* por el nombre de su rol. Un serviceAccountSelector vacío selecciona todas las cuentas de servicio del espacio de nombres. Puede reemplazar *my-security-group-policy* por un nombre para su SecurityGroupPolicy y *my-namespace* por el espacio de nombres en el que desea crear la SecurityGroupPolicy.

   Debe reemplazar *my_pod_security_group_id* por el ID de un grupo de seguridad existente. Si no dispone de un grupo de seguridad existente, debe crear uno. Para obtener más información, consulte Grupos de seguridad de Amazon EC2 para instancias de Linux en la Guía del usuario de Amazon EC2. Puede especificar de uno a cinco ID de grupo de seguridad. Si especifica más de un ID, la combinación de todas las reglas de todos los grupos de seguridad será efectiva para los pods seleccionados.

```
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**
>
> El grupo o los grupos de seguridad que especifique para sus pods deben cumplir los siguientes criterios:
>
> - Deben existir. Si no existen, cuando implementa un pod que coincida con el selector, el pod permanece atascado en el proceso de creación. Si describe el pod, verá un mensaje de error similar al siguiente: `An error occurred (InvalidSecurityGroupID.NotFound) when calling the CreateNetworkInterface operation: The securityGroup ID 'sg-05b1d815d1EXAMPLE' does not exist.`
> - Deben permitir la comunicación entrante desde el grupo de seguridad de clúster aplicado a sus nodos (para `kubelet`) a través de los puertos para los que haya configurado sondeos.
> - Deben permitir la comunicación saliente a través de los puertos `TCP` y `UDP` 53 a un grupo de seguridad asignado a los pods (o los nodos en los que los pods se ejecutan) que ejecutan CoreDNS. El grupo de seguridad de sus pods de CoreDNS debe permitir el tráfico entrante del puerto `TCP` y `UDP` 53 del grupo de seguridad que especifique.
> - Deben tener las reglas entrantes y salientes necesarias para comunicarse con otros pods con los que deben comunicarse.
> - Deben tener reglas que permitan a los pods comunicarse con el plano de control de Kubernetes si utilizan el grupo de seguridad con Fargate. La forma más sencilla de hacerlo es especificar el grupo de seguridad de clúster como uno de los grupos de seguridad.
>
> Las políticas de grupos de seguridad solo se aplican a los nuevos pods programados. No afectan a los pods en ejecución.

b. Implemente la política.

```
kubectl apply -f my-security-group-policy.yaml
```

3. Implemente una aplicación de muestra con una etiqueta que coincida con el valor *`my-role`* para *`podSelector`* que especificó en un paso anterior.

a. Copie los siguientes contenidos en su dispositivo. Reemplace los valores de ejemplo por los suyos y, a continuación, ejecute el comando modificado. Si reemplaza *my-role*, asegúrese de que sea igual al valor que especificó para el selector en un paso anterior.

```
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. Implemente la aplicación con el siguiente comando. Cuando implementa la aplicación, el complemento CNI de Amazon VPC para Kubernetes con la etiqueta de `role` y los grupos de seguridad que especificó en el paso anterior se aplican al pod.

```
kubectl apply -f sample-application.yaml
```

4. Visualización de los pods implementados con la aplicación de muestra. En el resto de este tema, se hace referencia a este terminal como `TerminalA`.

```
kubectl get pods -n my-namespace -o wide
```

Un ejemplo de salida sería el siguiente.

```
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**
>
> Pruebe estos consejos si hay pods atascados.
>
> - Si hay pods atascados en estado `Waiting`, ejecute `kubectl describe pod` *`my-deployment-xxxxxxxxxx-xxxxx`* `-n` *`my-namespace`* . Si ve `Insufficient permissions: Unable to create Elastic Network Interface.`, confirme que agregó la política de IAM al rol de clúster de IAM en un paso anterior.
> - Si algún pod se encuentra atascado en estado `Pending`, confirme que el tipo de instancia del nodo aparece en limits.go y que el producto del número máximo de interfaces de red de ramificación admitidas por el tipo de instancia multiplicado por el número de nodos del grupo de nodos aún no se ha alcanzado. Por ejemplo, una instancia `m5.large` admite nueve interfaces de red de ramificación. Si el grupo

de nodos tiene cinco nodos, se puede crear un máximo de 45 interfaces de red de ramificación para el grupo de nodos. El pod 46 que intente implementar se establecerá en el estado `Pending` hasta que se elimine otro pod que tenga grupos de seguridad asociados.

Si ejecuta `kubectl describe pod` *`my-deployment-xxxxxxxxxx-xxxxx`* `-n` *`my-namespace`* y ve un mensaje similar al siguiente mensaje, puede ignorarlo de forma segura. Este mensaje puede aparecer cuando el complemento CNI de Amazon VPC para Kubernetes intenta configurar las redes de host y falla mientras se crea la interfaz de red. El complemento registra este evento hasta que se crea la interfaz de red.

```
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
```

No puede exceder el número máximo de pods que se pueden ejecutar en el tipo de instancia. Para obtener una lista del número máximo de pods que puede ejecutar en cada tipo de instancia, consulte eni-max-pods.txt en GitHub. Cuando elimina un pod que tiene grupos de seguridad asociados o elimina el nodo en el que se ejecuta el pod, el controlador de recursos de VPC elimina la interfaz de red de ramificación. Si elimina un clúster con pods mediante pods para grupos de seguridad, el controlador no elimina las interfaces de red de ramificación, por lo que deberá eliminarlas por su cuenta. A fin de obtener más información sobre las interfaces de red, consulte Eliminar una interfaz de red en la Guía del usuario de Amazon EC2.

5. En un terminal separado, se inserta en uno de los pods. En el resto de este tema, se hace referencia a este terminal como `TerminalB`. Reemplace *`5df6f7687b-4fbjm`* por el ID de uno de los pods devueltos en la salida del paso anterior.

```
kubectl exec -it -n my-namespace my-deployment-5df6f7687b-4fbjm -- /bin/bash
```

6. Desde el intérprete de comandos de `TerminalB`, confirme que la aplicación de ejemplo funciona.

```
curl my-app
```

Un ejemplo de salida sería el siguiente.

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
[...]
```

Recibió el resultado porque todos los pods que se ejecutan en la aplicación están asociados al grupo de seguridad que creó. Ese grupo contiene una regla que permite todo el tráfico entre todos los pods al que está asociado el grupo de seguridad. Se permite el tráfico DNS saliente de ese grupo de seguridad al grupo de seguridad del clúster, lo que está asociado a los nodos. Los nodos ejecutan los pods de CoreDNS, en los que sus pods hicieron una búsqueda de nombre.

7. Desde `TerminalA`, elimine las reglas del grupo de seguridad que permita la comunicación DNS al grupo de seguridad del clúster del grupo de seguridad. Si no agregó las reglas DNS al grupo de seguridad del clúster en un paso anterior, sustituya *`$my_cluster_security_group_id`* con el ID del grupo de seguridad en el que creó las reglas.

```
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. Desde `TerminalB`, intente acceder de nuevo a la aplicación.

```
curl my-app
```

Un ejemplo de salida sería el siguiente.

```
curl: (6) Could not resolve host: my-app
```

El intento falla porque el pod ya no puede acceder a los pods de CoreDNS, que tienen asociado el grupo de seguridad del clúster. El grupo de seguridad del clúster ya no tiene las reglas del grupo de seguridad que permiten la comunicación de DNS desde el grupo de seguridad asociado a su pod.

Si intenta acceder a la aplicación con las direcciones IP devueltas para uno de los pods en un paso anterior, sigue recibiendo una respuesta dado que todos los puertos están permitidos entre

pods que tienen el grupo de seguridad asociado a ellos y no es necesaria una búsqueda de nombres.

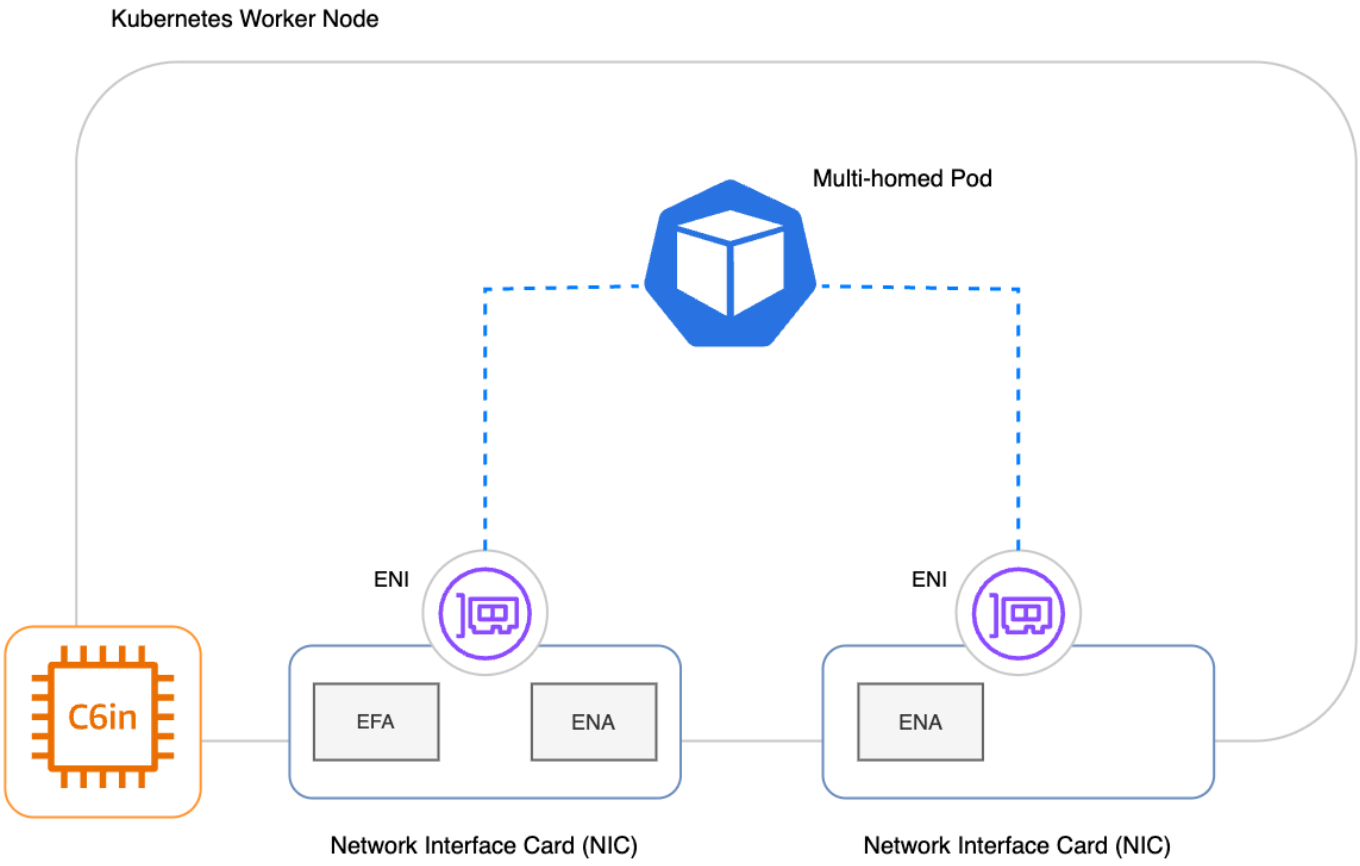
9. Una vez que haya terminado de experimentar, puede eliminar la política de grupo de seguridad, la aplicación y el grupo de seguridad de ejemplo que creó. Ejecute los siguientes comandos desde la `TerminalA`.

```
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
```

Conexión de múltiples interfaces de red a pods

De forma predeterminada, el complemento de la CNI de Amazon VPC asigna una dirección IP a cada pod. Esta dirección IP está conectada a una interfaz de red elástica que gestiona todo el tráfico entrante y saliente del pod. Para aumentar el ancho de banda y el rendimiento de la tasa de paquetes por segundo, puede utilizar la característica de NIC múltiple de la CNI de la VPC para configurar un pod con varios hosts. Un pod con múltiples hosts es un único pod de Kubernetes que utiliza varias interfaces de red (y varias direcciones IP). Al ejecutar un pod con varios hosts, puede distribuir el tráfico de sus aplicaciones entre varias interfaces de red mediante conexiones simultáneas. Esto resulta especialmente útil para los casos de uso de inteligencia artificial (IA), machine learning (ML) y computación de alto rendimiento (HPC).

En el siguiente diagrama, se muestra un pod con varios hosts que se ejecuta en un nodo de trabajo con varias tarjetas de interfaz de red (NIC) en uso.



Introducción

En Amazon EC2, una interfaz de red elástica es un componente de red lógico en una VPC que representa una tarjeta de red virtual. Para muchos tipos de instancias de EC2, las interfaces de red comparten una única tarjeta de interfaz de red (NIC) en el hardware. Esta NIC única tiene un ancho de banda máximo y una tasa máxima de paquetes por segundo.

Si la característica de NIC múltiple está habilitada, la CNI de la VPC no asigna direcciones IP en bloque, lo que hace de forma predeterminada. En su lugar, la CNI de la VPC asigna una dirección IP a una interfaz de red en cada tarjeta de red bajo demanda cuando se inicia un nuevo pod. Este comportamiento reduce la tasa de agotamiento de las direcciones IP, que aumenta con el uso de pods con varios hosts. Debido a que la CNI de la VPC asigna direcciones IP bajo demanda, los pods podrían tardar más en iniciarse en las instancias con la característica de NIC múltiple habilitada.

Consideraciones

- Asegúrese de que su clúster de Kubernetes esté ejecutando la versión `1.20.0`, o posterior, de la CNI de la VPC. La característica de NIC múltiple solo está disponible en la versión `1.20.0` de la CNI de la VPC o posterior.
- Habilite la variable de entorno `ENABLE_MULTI_NIC` en el complemento de la CNI de la VPC. Puede ejecutar el siguiente comando para establecer la variable e iniciar la implementación del DaemonSet.
  - `kubectl set env daemonset aws-node -n kube-system ENABLE_MULTI_NIC=true`
- Asegúrese de crear nodos de trabajo que tengan varias tarjetas de interfaz de red (NIC). Para obtener una lista de las instancias de EC2 que tienen varias tarjetas de interfaz de red, consulte Tarjetas de red en la Guía del usuario de Amazon EC2.
- Si la característica de NIC múltiple está habilitada, la CNI de la VPC no asigna direcciones IP en bloque, lo que hace de forma predeterminada. Debido a que la CNI de la VPC asigna direcciones IP bajo demanda, los pods podrían tardar más en iniciarse en las instancias con la característica de NIC múltiple habilitada. Para obtener más información, consulte la sección anterior the section called “Introducción”.
- Con la característica de NIC múltiple habilitada, los pods no tienen varias interfaces de red de forma predeterminada. Debe configurar cada carga de trabajo para usar varias NIC. Agregue la anotación `k8s.amazonaws.com/nicConfig: multi-nic-attachment` a las cargas de trabajo que deben tener varias interfaces de red.

`IPv6`Consideraciones de

- Política de IAM personalizada: en el caso de los clústeres de `IPv6`, cree y utilice la siguiente política de IAM personalizada para la CNI de la VPC. Esta política es específica de NIC múltiple. Para obtener más información general sobre el uso de la CNI de la VPC con clústeres de `IPv6`, consulte 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/*"
        }
    ]
}
```

- Mecanismo de transición de **`IPv6`** no disponible: si utiliza la característica de NIC múltiple, la CNI de la VPC no asigna una dirección `IPv4` a los pods de un clúster de `IPv6`. De lo contrario, la CNI de la VPC asigna una dirección `IPv4` local de host a cada pod para que puedan comunicarse con recursos de `IPv4` externos en otra Amazon VPC o en Internet.

Uso

Una vez que la característica de NIC múltiple esté habilitada en la CNI de la VPC y se hayan reiniciado los pods de `aws-node`, puede configurar cada carga de trabajo para que tenga varios hosts. El siguiente es un ejemplo de una configuración de YAML con la anotación requerida:

```
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:
...
```

Preguntas frecuentes

1. ¿Qué es una tarjeta de interfaz de red (NIC)?

Una tarjeta de interfaz de red (NIC), también denominada simplemente tarjeta de red, es un dispositivo físico que permite la conectividad de red para el hardware de computación en la nube subyacente. En los servidores de EC2 modernos, se refiere a la tarjeta de red Nitro. Una interfaz de red elástica (ENI) es una representación virtual de esta tarjeta de red subyacente.

Algunos tipos de instancias de EC2 tienen varias NIC para mayor ancho de banda y rendimiento de tasa de paquetes. Para tales instancias, puede asignar ENI secundarias a las tarjetas de red adicionales. Por ejemplo, la ENI n.° 1 puede funcionar como interfaz para la NIC conectada al índice de tarjeta de red 0, mientras que la ENI n.° 2 puede funcionar como interfaz para la NIC conectada a un índice de tarjeta de red separado.

2. ¿Qué es un pod con múltiples hosts?

Un pod con múltiples hosts es un único pod de Kubernetes que utiliza varias interfaces de red (y, en consecuencia, varias direcciones IP). Cada interfaz de red del pod está asociada a una interfaz de red elástica (ENI), y estas ENI son representaciones lógicas de NIC independientes en el nodo de trabajo subyacente. Con varias interfaces de red, un pod con varios hosts tiene una capacidad de transferencia de datos adicional, lo que también aumenta su tasa de transferencia de datos.

> **Important**
>
> La CNI de la VPC solo puede configurar pods con varios hosts en tipos de instancias que tengan varias NIC.

3. Por qué debo utilizar esta característica?

Si necesita escalar el rendimiento de la red en sus cargas de trabajo basadas en Kubernetes, puede utilizar la característica de NIC múltiple para ejecutar pods con varios hosts que interactúen con todas las NIC subyacentes que tengan un dispositivo ENA conectado. El uso de tarjetas de red adicionales aumenta la capacidad de ancho de banda y el rendimiento de la velocidad de paquetes de sus aplicaciones al distribuir el tráfico de las aplicaciones entre varias conexiones simultáneas. Esto resulta especialmente útil para los casos de uso de inteligencia artificial (IA), machine learning (ML) y computación de alto rendimiento (HPC).

4. ¿Cómo utilizo esta característica?

1. En primer lugar, debe asegurarse de que su clúster de Kubernetes utilice la versión 1.20 o posterior de la CNI de la VPC. Para conocer los pasos para actualizar la CNI de la VPC como un complemento de EKS, consulte the section called “Actualización (del complemento de EKS)”.
2. A continuación, debe habilitar la compatibilidad con varias NIC en la CNI de la VPC mediante la variable de entorno `ENABLE_MULTI_NIC`.
3. A continuación, debe asegurarse de crear y unir nodos que tengan varias tarjetas de red. Para obtener una lista de los tipos de instancias de EC2 que tienen varias tarjetas de red, consulte Tarjetas de red en la Guía del usuario de Amazon EC2.
4. Por último, debe configurar cada carga de trabajo para que utilice varias interfaces de red (pods con varios hosts) o una única interfaz de red.

5. ¿Cómo configuro mis cargas de trabajo para usar varias NIC en un nodo de trabajo compatible?

Para usar pods con varios hosts, debe agregar la siguiente anotación: `k8s.amazonaws.com/nicConfig: multi-nic-attachment`. Esto conectará una ENI de cada NIC de la instancia subyacente al pod (mapeo de uno a varios entre un pod y las NIC).

Si falta esta anotación, la CNI de la VPC asume que el pod solo requiere una interfaz de red y le asigna una IP desde una ENI en cualquier NIC disponible.

6. ¿Qué adaptadores de interfaz de red son compatibles con esta característica?

Puede utilizar cualquier adaptador de interfaz de red si tiene al menos un ENA conectado a la tarjeta de red subyacente para el tráfico IP. Para obtener más información sobre los ENA, consulte Elastic Network Adapter (ENA) en la Guía del usuario de Amazon EC2.

Configuraciones de dispositivos de red compatibles:

- Las interfaces ENA proporcionan todas las características de enrutamiento y redes IP tradicionales que se requieren para admitir las redes IP de una VPC. Para obtener más información, consulte Habilitar redes mejoradas con ENA en las instancias de EC2.
- Las interfaces EFA (EFA con ENA) proporcionan tanto el dispositivo de ENA para redes IP como el dispositivo EFA para comunicaciones de baja latencia y alto rendimiento.

> **Important**
>
> Si una tarjeta de red solo tiene conectado un adaptador exclusivo de EFA, la CNI de la VPC lo omitirá al aprovisionar la conectividad de red para un pod con varios hosts. Sin embargo, si combina un adaptador exclusivo de EFA con un adaptador para ENA en una tarjeta de red, la CNI de la VPC también administrará las ENI de este dispositivo. Para utilizar interfaces exclusivas de EFA con clústeres de EKS, consulte the section called “Configuración del entrenamiento de clústeres con EFA”.

7. ¿Puedo ver si un nodo de mi clúster es compatible con ENA?

Sí, puede usar la CLI de AWS o la API de EC2 para recuperar información de red sobre una instancia de EC2 de su clúster. Esto proporciona detalles sobre si la instancia es compatible con ENA o no. En el siguiente comando, sustituya `<your-instance-id>` por el ID de la instancia de EC2 de un nodo.

AWSEjemplo de la CLI de :

```
aws ec2 describe-instances --instance-ids <your-instance-id> --query
 "Reservations[].Instances[].EnaSupport"
```

Ejemplo de código de salida:

```
[ true ]
```

8. ¿Puedo ver las distintas direcciones IP asociadas a un pod?

No, no es fácil. Sin embargo, puede utilizar `nsenter` desde el nodo para ejecutar herramientas de red comunes, como `ip route show`, y ver las direcciones IP e interfaces adicionales.

9. ¿Puedo controlar el número de interfaces de red de mis pods?

No. Cuando la carga de trabajo está configurada para usar varias NIC en una instancia compatible, un único pod recibe automáticamente una dirección IP de cada tarjeta de red de la instancia. Como alternativa, los pods con un solo host tendrán una interfaz de red conectada a una NIC de la instancia.

> **Important**
>
> La CNI de la VPC omite las tarjetas de red que solo tienen conectado un dispositivo exclusivo de EFA.

10. ¿Puedo configurar mis pods para que usen una NIC específica?

No, no se puede. Si un pod tiene la anotación correspondiente, la CNI de la VPC lo configura automáticamente para usar todas las NIC con un adaptador de ENA en el nodo de trabajo.

11. ¿Esta característica funciona con las demás características de red de la CNI de la VPC?

Sí, la característica de NIC múltiple de la CNI de la VPC funciona tanto con redes personalizadas como con detección de subredes mejorada. Sin embargo, los pods con varios hosts no utilizan las subredes ni los grupos de seguridad personalizados. En su lugar, la CNI de la VPC asigna direcciones IP e interfaces de red a los pods de múltiples hosts con la misma configuración de subred y grupo de seguridad que el nodo. Para obtener más información sobre las redes personalizadas, consulte the section called “Redes personalizadas”.

La característica de NIC múltiple de la CNI de la VPC no funciona con los grupos de seguridad para los pods ni se puede combinar con ellos.

12. ¿Puedo usar políticas de red con esta característica?

Sí, puede usar las políticas de red de Kubernetes con varias NIC. Las políticas de red de Kubernetes restringen el tráfico de red que entra y sale de sus pods. Para obtener más información sobre cómo aplicar políticas de red con la CNI de la VPC, consulte the section called “Políticas de Kubernetes”.

13. ¿Está habilitada la compatibilidad con múltiples NIC en el modo automático de EKS?

La configuración de múltiples NIC no es compatible con los clústeres del modo automático de EKS.

Complementos de CNI alternativos para clústeres de Amazon EKS

El Complemento de la CNI de Amazon VPC para Kubernetes es el único complemento de la CNI admitido por Amazon EKS con nodos de Amazon EC2. Amazon EKS es compatible con las capacidades principales de Cilium y Calico para los Nodos híbridos de Amazon EKS. Amazon EKS ejecuta una versión anterior de Kubernetes, por lo que puede instalar complementos de CNI alternativos compatibles en los nodos de Amazon EC2 del clúster. Si tiene nodos de Fargate en el clúster, el complemento CNI de Amazon VPC para Kubernetes ya está en sus nodos de Fargate. Es el único complemento de CNI que puede utilizar con los nodos de Fargate. Se produce un error cuando intenta instalar un complemento de CNI alternativo en los nodos de Fargate.

Si planea usar un complemento de CNI alternativo en nodos de Amazon EC2, le recomendamos obtener compatibilidad comercial para el complemento o pedir al experto en plantilla que solucione los problemas y aporte correcciones al proyecto de complemento de CNI.

Amazon EKS mantiene relaciones con una red de socios que ofrecen soporte para complementos CNI compatibles alternativos. Consulte la siguiente documentación de socios para obtener información detallada sobre las versiones, las calificaciones y pruebas realizadas.

Socio	Producto	Documentación
Tigera	Calico	Instrucciones de instalación
Isovalent	Cilium	Instrucciones de instalación
Juniper	Redes en contrail nativas en la nube (CN2)	Instrucciones de instalación
VMware	Antrea	Instrucciones de instalación

Amazon EKS tiene como objetivo darle una amplia selección de opciones para cubrir todos los casos de uso.

Plugins de políticas de red compatibles alternativos

Calico es una solución ampliamente adoptada para seguridad y redes de contenedores. El uso de Calico en EKS proporciona una implementación de la política de red plenamente compatible para sus clústeres de EKS. Además, puede optar por utilizar la red de Calico, que conserva las

direcciones IP de la VPC subyacente. Calico Cloud mejora las características de Calico Open Source con capacidades avanzadas de seguridad y observabilidad.

El flujo de tráfico hacia y desde los pods con grupos de seguridad asociados no está sujeto a la política de red de Calico y solo se limita a la aplicación de grupos de seguridad de Amazon VPC.

Si utiliza la aplicación de políticas de red de Calico, le recomendamos que configure la variable de entorno `ANNOTATE_POD_IP` en `true` para evitar un problema conocido con Kubernetes. Para utilizar esta característica, debe agregar permisos de `patch` para los pods al ClusterRole `aws-node`. Tenga en cuenta que agregar permisos de parche al DaemonSet `aws-node` aumenta el alcance de seguridad del complemento. Para obtener más información, consulte ANNOTATE_POD_IP en el repositorio de CNI de la VPC en GitHub.

Consideraciones para el modo automático de Amazon EKS

El modo automático de Amazon EKS no admite complementos de la CNI alternativos ni complementos de políticas de red. Para obtener más información, consulte Modo automático de EKS.

Adjunción de múltiples interfaces de red a pods con Multus

Multus CNI es un complemento de interfaz de red de contenedor (CNI) para Amazon EKS que permite adjuntar varias interfaces de red a un pod. Para obtener más información, consulte la documentación de Multus-CNI en GitHub.

En Amazon EKS, cada pod tiene una interfaz de red asignada por el complemento CNI de Amazon VPC. Con Multus, puede crear un pod de varios alojamientos que tenga varias interfaces. Esto se logra mediante Multus al actuar como un “meta-complemento”; un complemento CNI que puede llamar a varios otros complementos CNI. El soporte de AWS para Multus viene configurado con el complemento CNI de Amazon VPC como complemento delegado predeterminado.

- Amazon EKS no creará ni publicará complementos CNI de virtualización de E/S de raíz única (SR-IOV) y del kit de desarrollo del plano de datos (DPDK). Sin embargo, puede lograr la aceleración de paquetes al conectarse directamente a Amazon EC2 Elastic Network Adapters (ENA) a través del dispositivo host administrado de Multus y los complementos `ipvlan`.
- Amazon EKS es compatible con Multus, que proporciona un proceso genérico que permite encadenar fácilmente complementos CNI adicionales. Se admite Multus y el proceso de encadenamiento, pero AWS no proporcionará soporte para todos los complementos CNI

compatibles que se pueden encadenar, o problemas que puedan surgir en esos complementos CNI que no están relacionados con la configuración de encadenamiento.

- Amazon EKS ofrece soporte y gestión del ciclo de vida para el complemento de Multus, pero no se hace responsable de ninguna dirección IP ni de la administración adicional asociada con las interfaces de red adicionales. La dirección IP y la gestión de la interfaz de red predeterminada que utiliza el complemento CNI de Amazon VPC permanece sin cambios.
- Solo se admite oficialmente el complemento CNI de Amazon VPC como complemento delegado predeterminado. Debe modificar el manifiesto de instalación de Multus publicado para volver a configurar el complemento delegado predeterminado en un CNI alternativo si decide no utilizar el complemento CNI de Amazon VPC para redes principales.
- Multus solo se admite cuando se utiliza el CNI de Amazon VPC como CNI principal. No admitimos el CNI de Amazon VPC cuando se utiliza para interfaces de orden superior, secundarias o de otro tipo.
- Para evitar que el complemento CNI de Amazon VPC intente administrar interfaces de red adicionales asignadas a los pods, agregue la etiqueta siguiente a la interfaz de red:

  clave

  : `node.k8s.amazonaws.com/no_manage`

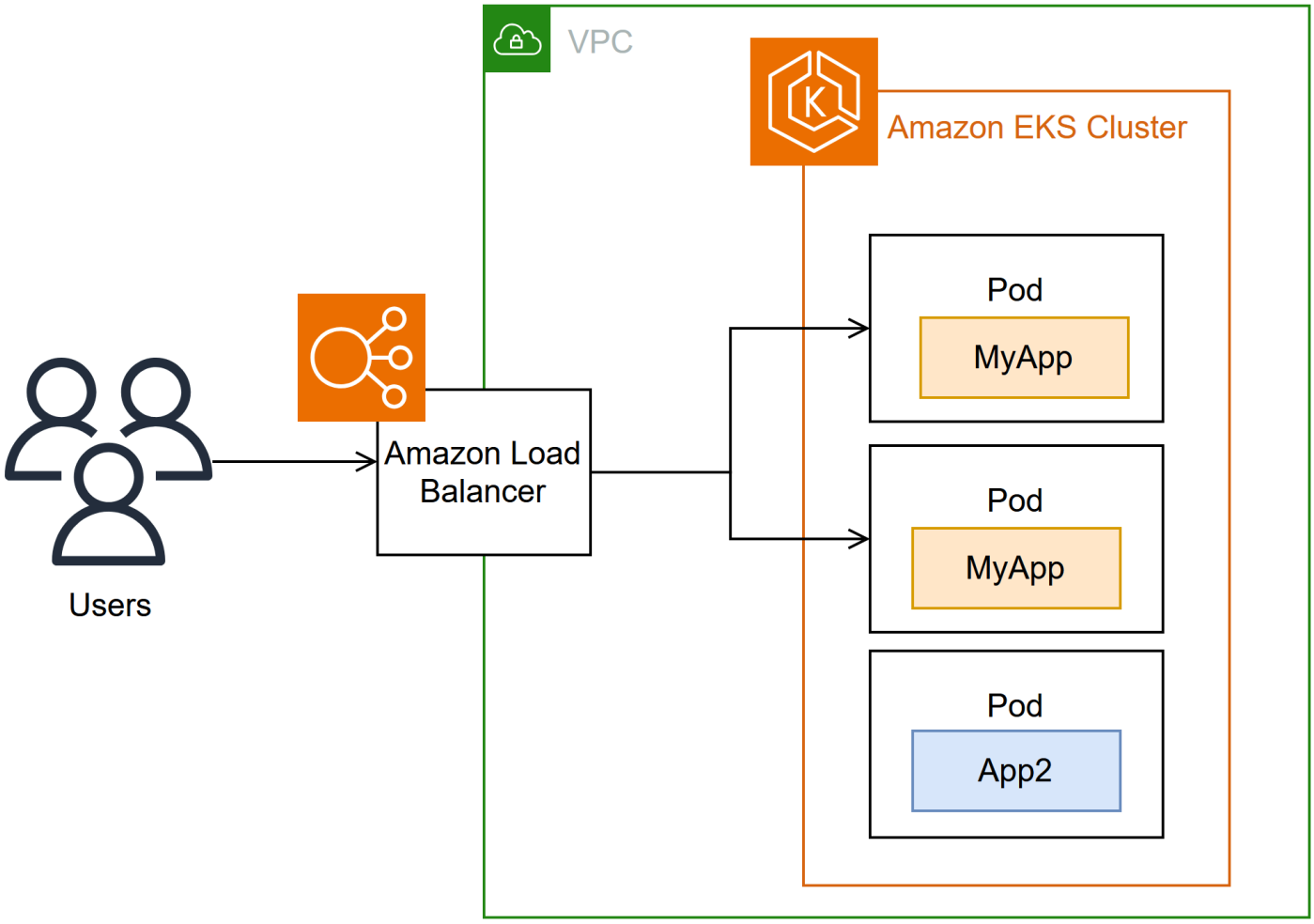
  value

  : `true`
- Multus es compatible con las políticas de red, pero la política tiene que enriquecerse para incluir puertos y direcciones IP que pueden formar parte de interfaces de red adicionales adjuntas a los pods.

Para ver la explicación de la implementación, consulte la Guía de configuración de Multus en GitHub.

Enrutamiento del tráfico de internet con el controlador del equilibrador de carga de AWS

El Controlador del equilibrador de carga de AWS administra los Elastic Load Balancer de AWS para un clúster de Kubernetes. Puede usar el controlador para exponer las aplicaciones del clúster a Internet. El controlador aprovisiona los equilibradores de carga de AWS que apuntan a los recursos de Service o Ingress del clúster. En otras palabras, el controlador crea una única dirección IP o nombre de DNS que apunta a varios pods del clúster.



El controlador observa los recursos de entrada o servicio de Kubernetes. En respuesta, crea los recursos adecuados de Elastic Load Balancing de AWS. Puede configurar el comportamiento específico de los equilibradores de carga mediante la aplicación de anotaciones a los recursos de Kubernetes. Por ejemplo, puede adjuntar grupos de seguridad de AWS a los equilibradores de carga mediante anotaciones.

El controlador aprovisiona los siguientes recursos:

`Ingress` de Kubernetes

El LBC crea un equilibrador de carga de aplicación (ALB) de AWS cuando se crea una `Ingress` de Kubernetes. Revise las anotaciones que puede aplicar a un recurso de Ingress.

Servicio de Kubernetes del tipo **`LoadBalancer`**

El LBC crea un equilibrador de carga de red (NLB) de AWS cuando crea un servicio de Kubernetes del tipo `LoadBalancer`. Revise las anotaciones que puede aplicar a un recurso de Service.

En el pasado, se utilizaba el equilibrador de carga de red de Kubernetes para destinos de instancia, pero se usaba el LBC para destinos de IP. Con la versión `2.3.0` o posterior del Controlador del equilibrador de carga de AWS, puede crear NLB con cualquiera de los tipos de destino. Para obtener más información acerca de los tipos de destinos del NLB, consulte Tipo de destino en la Guía del usuario para Network Load Balancers.

El controlador es un proyecto de código abierto administrado en GitHub.

Antes de implementar el controlador, recomendamos que revise los requisitos previos y las consideraciones que figuran en Enrutamiento de aplicaciones y tráfico HTTP con equilibradores de carga de aplicaciones y the section called “Equilibrio de carga de red”. En esos temas, implementará una aplicación de muestra que incluye un equilibrador de carga de AWS.

API de **`Gateway`** de Kubernetes

Con la versión `2.14.0` o posterior del Controlador del equilibrador de carga de AWS, el LBC crea un Equilibrador de carga de aplicación de AWS (ALB) al crear una `Gateway` de Kubernetes. La puerta de enlace de Kubernetes estandariza más configuraciones que Entrada, que necesitaba anotaciones personalizadas para muchas de las opciones habituales. Revise la configuración que puede aplicar a un recurso de puerta de enlace. Para obtener más información acerca de la API de `Gateway`, consulte Gateway API en la documentación de Kubernetes.

Instalación del controlador

Puede utilizar uno de los siguientes procedimientos para instalar el Controlador del equilibrador de carga de AWS:

- Si es la primera vez que utiliza Amazon EKS, le recomendamos que utilice Helm para la instalación, ya que simplifica la instalación del Controlador del equilibrador de carga de AWS. Para obtener más información, consulte the section called “Instalación con Helm”.

- Para configuraciones avanzadas, como clústeres con acceso de red restringido a los registros de contenedores públicos, utilice los manifiestos de Kubernetes. Para obtener más información, consulte the section called “Instalación de con manifiestos”.

Migración desde versiones de controlador obsoletas

- Si tiene versiones obsoletas del Controlador del equilibrador de carga de AWS instaladas, consulte the section called “Migración desde el Controlador obsoleto”.
- Las versiones obsoletas no se pueden actualizar. Deben eliminarse y se debe instalar una versión actual del Controlador del equilibrador de carga de AWS.
- Las versiones obsoleta incluyen lo siguiente:
  - Controlador de entradas de ALB de AWS para Kubernetes (“Controlador de entradas”), un predecesor del Controlador del equilibrador de carga de AWS.
  - Cualquier versión `0.1.x` del Controlador del equilibrador de carga de AWS

Proveedor de nube heredado

Kubernetes incluye un proveedor de nube heredado para AWS. El proveedor de nube heredado es capaz de aprovisionar equilibradores de carga de AWS, similares al Controlador del equilibrador de carga de AWS. El proveedor de nube heredado crea equilibradores de carga clásicos. Si no instala el Controlador del equilibrador de carga de AWS, Kubernetes utilizará de forma predeterminada el proveedor de nube heredado. Debe instalar el Controlador del equilibrador de carga de AWS y evitar utilizar el proveedor de nube heredado.

> **Important**
>
> En las versiones 2.5 y posteriores, el Controlador del equilibrador de carga de AWS se convierte en el controlador predeterminado para los recursos del servicio de Kubernetes con el `type: LoadBalancer` y hace un Equilibrador de carga de red (NLB) de AWS para cada servicio. Para ello, crea un webhook mutante para los servicios, que establece el campo `spec.loadBalancerClass` para `service.k8s.aws/nlb` para nuevos servicios de `type: LoadBalancer`. Puede desactivar esta característica y volver a utilizar el proveedor de nube tradicional como controlador predeterminado, estableciendo el valor del gráfico de Helm de `enableServiceMutatorWebhook` a `false`. El clúster no aprovisionará

nuevos equilibradores de carga clásicos para sus servicios a menos que desactive esta característica. Los equilibradores de carga clásicos existentes seguirán funcionando.

Instalación del Controlador del equilibrador de carga de AWS con Helm

> **Tip**
>
> Con el modo automático de Amazon EKS, no necesita instalar ni actualizar complementos de red. El modo automático ofrece capacidades para la conexión en red de pods y el equilibrio de carga.
> Para obtener más información, consulte Modo automático de EKS.

En este tema se describe cómo instalar el Controlador del equilibrador de carga AWS con Helm, un administrador de paquetes para Kubernetes y `eksctl`. El controlador se instala con las opciones predeterminadas. Para obtener más información sobre el controlador, incluidos los detalles sobre su configuración con anotaciones, consulte la documentación del Controlador del equilibrador de carga de AWS en GitHub.

En los siguientes pasos, reemplace los valores de ejemplo por sus propios valores.

Requisitos previos

Antes de comenzar este tutorial, debe completar los siguientes pasos:

- Cree un clúster de Amazon EKS. Para crear uno, consulte Introducción.
- Instale Helm en el equipo local.
- Asegúrese de que sus complementos CNI de Amazon VPC para Kubernetes, `kube-proxy` y CoreDNS tengan las versiones mínimas enumeradas en Service account tokens.
- Conozca los conceptos de AWS Elastic Load Balancing. Para obtener más información, consulte la Guía del usuario de Elastic Load Balancing.
- Obtenga información sobre los recursos del servicio y de entrada de Kubernetes.

Consideraciones

Antes de seguir con los pasos de configuración de esta página, tenga en cuenta lo siguiente:

- La política y el rol (`AmazonEKSLoadBalancerControllerRole`) de IAM se pueden reutilizar en varios clústeres de EKS de la misma cuenta de AWS.
- Si instalará el controlador en el mismo clúster en el que se creó originalmente el rol (`AmazonEKSLoadBalancerControllerRole`), vaya al Paso 2: Instalación del controlador del equilibrador de carga después de comprobar que el rol existe.
- Si utiliza roles de IAM para cuentas de servicio (IRSA), estos deben configurarse para cada clúster y el ARN del proveedor de OpenID Connect (OIDC) de la política de confianza del rol es específico de cada clúster de EKS. Además, si instalará el controlador en un clúster nuevo con un `AmazonEKSLoadBalancerControllerRole` existente, actualice la política de confianza del rol para incluir el proveedor de OIDC del nuevo clúster y cree una nueva cuenta de servicio con la anotación de rol adecuada. Para determinar si ya tiene un proveedor de OIDC o para crear uno, consulte the section called “Proveedor de OIDC de IAM”.

Paso 1: crear el rol de IAM usando **`eksctl`**

Los siguientes pasos corresponden a la versión 2.14.1 del controlador del equilibrador de carga de AWS. Para obtener más información sobre todas las versiones, consulte la página de versiones del Controlador del equilibrador de carga de AWS en GitHub.

1. Descargue una política de IAM para el Controlador del equilibrador de carga de AWS que le permita realizar llamadas a las API de AWS en su nombre.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-
controller/v2.14.1/docs/install/iam_policy.json
```

   - Si pertenece a una partición de AWS no estándar, como una región gubernamental o de China, revise las políticas en GitHub y descargue la política adecuada según la región.

2. Cree una política de IAM con la política descargada en el paso anterior.

```
aws iam create-policy \
    --policy-name AWSLoadBalancerControllerIAMPolicy \
    --policy-document file://iam_policy.json
```

> **Note**
>
> Si ve la política en la Consola de administración de AWS, la consola muestra advertencias para el servicio ELB, pero no para el servicio ELB v2. Esto ocurre porque algunas de

las acciones de la política existen para ELB v2, pero no para ELB. Puede obviar estas advertencias para ELB.

3. Sustituya los valores por el nombre del clúster, el código de la región y el ID de la cuenta.

```
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
```

Paso 2: instalación del Controlador del equilibrador de carga de AWS

1. Añada el repositorio de gráficos de Helm `eks-charts`. AWS mantiene este repositorio en GitHub.

```
helm repo add eks https://aws.github.io/eks-charts
```

2. Actualice el repositorio local para asegurarse de que cuenta con los gráficos más recientes.

```
helm repo update eks
```

3. Instale el Controlador del equilibrador de carga de AWS.

   Si va a implementar el controlador en nodos de Amazon EC2 que tienen acceso restringido al servicio de metadatos de instancias de Amazon EC2 (IMDS), o si va a implementar en Nodos híbridos de Amazon EKS o Fargate, agregue los siguientes indicadores al comando `helm` siguiente:

   - `--set region=`*`region-code`*
   - `--set vpcId=`*`vpc-xxxxxxxx`*

   Reemplace *`my-cluster`* por el nombre de su clúster. En el comando siguiente, `aws-load-balancer-controller` es la cuenta de servicio de Kubernetes que creó en un paso anterior.

   Para obtener más información acerca de la configuración del gráfico de Helm, consulte values.yaml en 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

El gráfico implementado no recibe actualizaciones de seguridad de forma automática. Debe actualizar de forma manual a un gráfico más reciente cuando se encuentre disponible. Al actualizar, cambie *`install`* por `upgrade` en el comando anterior.

El comando `helm install` instala automáticamente las definiciones de recursos personalizadas (CRD) para el controlador. El comando `helm upgrade` no lo hace. Si usa `helm upgrade`, , debe instalar manualmente las CRD. Ejecute el siguiente comando para instalar las CRD:

```
wget https://raw.githubusercontent.com/aws/eks-charts/master/stable/aws-load-balancer-
controller/crds/crds.yaml
kubectl apply -f crds.yaml
```

Paso 3: verificar que el controlador se encuentre instalado

1. Verifique que el controlador se encuentre instalado.

   ```
   kubectl get deployment -n kube-system aws-load-balancer-controller
   ```

   Un ejemplo de salida sería el siguiente.

   ```
   NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
   aws-load-balancer-controller   2/2     2            2           84s
   ```

   Recibe la salida anterior si ha implementado mediante Helm. Si ha implementado utilizando el manifiesto de Kubernetes, solo tiene una réplica.

2. Antes de utilizar el controlador para aprovisionar el recurso de AWS, el clúster debe cumplir requisitos específicos. Para obtener más información, consulte the section called “Equilibrio de carga de aplicaciones” y the section called “Equilibrio de carga de red”.

Instalación del Controlador del equilibrador de carga de AWS con manifiestos

> **Tip**
>
> Con el modo automático de Amazon EKS, no necesita instalar ni actualizar complementos de red. El modo automático ofrece capacidades para la conexión en red de pods y el equilibrio de carga.
> Para obtener más información, consulte Modo automático de EKS.

En este tema, se describe cómo instalar el controlador mediante la descarga y la aplicación de los manifiestos de Kubernetes. Puede ver la documentación completa para el controlador en GitHub.

En los siguientes pasos, reemplace los valores de ejemplo por sus propios valores.

Requisitos previos

Antes de comenzar este tutorial, debe completar los siguientes pasos:

- Cree un clúster de Amazon EKS. Para crear uno, consulte Introducción.
- Instale Helm en el equipo local.
- Asegúrese de que sus complementos CNI de Amazon VPC para Kubernetes, `kube-proxy` y CoreDNS tengan las versiones mínimas enumeradas en Service account tokens.
- Conozca los conceptos de AWS Elastic Load Balancing. Para obtener más información, consulte la Guía del usuario de Elastic Load Balancing.
- Obtenga información sobre los recursos del servicio y de entrada de Kubernetes.

Consideraciones

Antes de seguir con los pasos de configuración de esta página, tenga en cuenta lo siguiente:

- La política y el rol (`AmazonEKSLoadBalancerControllerRole`) de IAM se pueden reutilizar en varios clústeres de EKS de la misma cuenta de AWS.

- Si instalará el controlador en el mismo clúster en el que se creó originalmente el rol (`AmazonEKSLoadBalancerControllerRole`), vaya al Paso 2: Instalación de cert-manager después de comprobar que el rol existe.
- Si utiliza roles de IAM para cuentas de servicio (IRSA), estos deben configurarse para cada clúster y el ARN del proveedor de OpenID Connect (OIDC) de la política de confianza del rol es específico de cada clúster de EKS. Además, si instalará el controlador en un clúster nuevo con un `AmazonEKSLoadBalancerControllerRole` existente, actualice la política de confianza del rol para incluir el proveedor de OIDC del nuevo clúster y cree una nueva cuenta de servicio con la anotación de rol adecuada. Para determinar si ya tiene un proveedor de OIDC o para crear uno, consulte the section called “Proveedor de OIDC de IAM”.

Paso 1: configurar IAM

Los siguientes pasos corresponden a la versión 2.14.1 del controlador del equilibrador de carga de AWS. Para obtener más información sobre todas las versiones, consulte la página de versiones del Controlador del equilibrador de carga de AWS en GitHub.

1. Descargue una política de IAM para el Controlador del equilibrador de carga de AWS que le permita realizar llamadas a las API de AWS en su nombre.

   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. Cree una política de IAM con la política descargada en el paso anterior.

   ```
   aws iam create-policy \
       --policy-name AWSLoadBalancerControllerIAMPolicy \
   ```

```
    --policy-document file://iam_policy.json
```

> **Note**
>
> Si ve la política en la Consola de administración de AWS, la consola muestra advertencias para el servicio ELB, pero no para el servicio ELB v2. Esto ocurre porque algunas de las acciones de la política existen para ELB v2, pero no para ELB. Puede obviar estas advertencias para ELB.

Example

eksctl

a. Reemplace *my-cluster* por el nombre de su clúster y *111122223333* con el ID de su cuenta y luego ejecute el comando.

```
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. Recupere el ID del proveedor de OIDC de su clúster y almacénelo en una variable.

```
oidc_id=$(aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text | cut -d '/' -f 5)
```

b. Determine si ya hay un proveedor de OIDC de IAM con el ID de su clúster en su cuenta. Debe tener configurado OIDC tanto para el clúster como para IAM.

```
aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
```

Si no se devuelve ninguna salida, debe contar con un proveedor de OIDC de IAM para su clúster. Si no se devuelve ninguna salida, debe crear un proveedor de OIDC de IAM para el clúster. Para obtener más información, consulte the section called “Proveedor de OIDC de IAM”.

c. Copie los siguientes contenidos en su dispositivo. Reemplace *111122223333* por el ID de su cuenta. Reemplace *region-code* por la región de AWS en la que se encuentra el clúster. Reemplace *EXAMPLED539D4633E53DE1B71EXAMPLE* con la salida que obtuvo en el paso anterior.

```
{
    "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. Creación del rol de IAM.

```
aws iam create-role \
  --role-name AmazonEKSLoadBalancerControllerRole \
  --assume-role-policy-document file://"load-balancer-role-trust-policy.json"
```

e. Adjunte la política de IAM administrada por Amazon EKS requerida al rol de IAM. Reemplace *111122223333* por el ID de su cuenta.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::111122223333:policy/AWSLoadBalancerControllerIAMPolicy
 \
  --role-name AmazonEKSLoadBalancerControllerRole
```

f. Copie los siguientes contenidos en su dispositivo. Reemplace *111122223333* por el ID de su cuenta. Después de reemplazar el texto, ejecute el comando modificado para crear el archivo `aws-load-balancer-controller-service-account.yaml`.

```
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. Cree la cuenta de servicio de Kubernetes en el clúster. La cuenta de servicio de Kubernetes denominada `aws-load-balancer-controller` está anotada con el rol de IAM que creó con el nombre *AmazonEKSLoadBalancerControllerRole*.

```
kubectl apply -f aws-load-balancer-controller-service-account.yaml
```

Paso 2: instalar el **`cert-manager`**

Instale el `cert-manager` con uno de los siguientes métodos para ingresar la configuración del certificado en los webhooks. Para obtener más información, consulte Cómo empezar en la documentación de cert-manager.

Se recomienda utilizar el registro del contenedor `quay.io` para realizar la instalación de `cert-manager`. Si los nodos no tienen acceso al registro contenedor de `quay.io`, instale `cert-manager` mediante Amazon ECR (consulte a continuación).

Example

Quay.io

a. Si los nodos tienen acceso al registro de contenedores de `quay.io`, instale el `cert-manager` para ingresar la configuración del certificado en los webhooks.

```
kubectl apply \
    --validate=false \
    -f https://github.com/jetstack/cert-manager/releases/download/v1.13.5/cert-manager.yaml
```

Amazon ECR

a. Instale el `cert-manager` con uno de los siguientes métodos para ingresar la configuración del certificado en los webhooks. Para obtener más información, consulte Cómo empezar en la documentación de cert-manager.

b. Descargue el manifiesto.

```
curl -Lo cert-manager.yaml https://github.com/jetstack/cert-manager/releases/download/v1.13.5/cert-manager.yaml
```

c. Extraiga las siguientes imágenes y envíelas un repositorio al que tengan acceso sus nodos. Para obtener más información sobre cómo extraer, etiquetar y enviar las imágenes en su propio repositorio, consulte the section called “Copiar una imagen en un repositorio”.

```
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. Reemplace `quay.io` en el manifiesto de las tres imágenes por su propio nombre de registro. El siguiente comando supone que el nombre del repositorio privado es el mismo que el repositorio de origen. Sustituya *111122223333.dkr.ecr.region-code.amazonaws.com* por el registro privado.

```
sed -i.bak -e 's|quay.io|111122223333.dkr.ecr.region-code.amazonaws.com|' ./cert-manager.yaml
```

e. Aplique el manifiesto.

```
kubectl apply \
    --validate=false \
    -f ./cert-manager.yaml
```

Paso 3: instalación del Controlador del equilibrador de carga de AWS

1. Descargue la especificación del controlador. Para obtener más información sobre el controlador, consulte la documentación en 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. Lleve a cabo las siguientes modificaciones en el archivo.
   a. Si ha descargado el archivo `v2_14_1_full.yaml`, ejecute el siguiente comando para eliminar la sección `ServiceAccount` del manifiesto. Si no elimina esta sección, se sobrescribirá la anotación obligatoria que hizo en la cuenta de servicio en un paso anterior. Al eliminar esta sección también conserva la cuenta de servicio que creó en un paso anterior si elimina el controlador.

```
sed -i.bak -e '764,772d' ./v2_14_1_full.yaml
```

   Si ha descargado una versión de archivo diferente, abra el archivo en un editor y elimine las siguientes líneas.

```
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. Reemplace `your-cluster-name` en la sección `Deployment spec` del archivo por el nombre del clúster. Para ello, reemplace *`my-cluster`* por el nombre del clúster.

```
sed -i.bak -e 's|your-cluster-name|my-cluster|' ./v2_14_1_full.yaml
```

c. Si los nodos no tienen acceso a los repositorios de imágenes de Amazon ECR de Amazon EKS, tiene que extraer la siguiente imagen y enviarla a un repositorio al que tengan acceso los nodos. Para obtener más información sobre cómo extraer, etiquetar y enviar una imagen a su propio repositorio, consulte the section called “Copiar una imagen en un repositorio”.

```
public.ecr.aws/eks/aws-load-balancer-controller:v2.14.1
```

Agregue el nombre del registro al manifiesto. El siguiente comando supone que el nombre del repositorio privado es el mismo que el repositorio de origen y agrega el nombre del registro privado al archivo. Sustituya *111122223333.dkr.ecr.region-code.amazonaws.com* por el registro. En esta línea se supone que ha asignado el mismo nombre al repositorio privado que al repositorio de origen. Si no es así, cambie el texto `eks/aws-load-balancer-controller` después del nombre del registro privado al nombre del repositorio.

```
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. (Necesario solo para IMDS restringido o Fargate)

Si va a implementar el controlador en nodos de Amazon EC2 que tienen acceso restringido al servicio de metadatos de instancias de Amazon EC2 (IMDS), o si va a realizar una implementación en Nodos híbridos de Amazon EKS o Fargate, agregue los `following parameters` bajo `- args:`.

```
[...]
spec:
      containers:
        - args:
            - --cluster-name=your-cluster-name
            - --ingress-class=alb
            - --aws-vpc-id=vpc-xxxxxxxx
            - --aws-region=region-code


[...]
```

3. Aplique el archivo.

```
kubectl apply -f v2_14_1_full.yaml
```

4. Descargue el manifiesto `IngressClass` y `IngressClassParams` a su clúster.

```
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. Aplique el manifiesto al clúster.

```
kubectl apply -f v2_14_1_ingclass.yaml
```

Paso 4: verificar que el controlador se encuentre instalado

1. Verifique que el controlador se encuentre instalado.

```
kubectl get deployment -n kube-system aws-load-balancer-controller
```

   Un ejemplo de salida sería el siguiente.

```
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
aws-load-balancer-controller   2/2     2            2           84s
```

   Recibe la salida anterior si ha implementado mediante Helm. Si ha implementado utilizando el manifiesto de Kubernetes, solo tiene una réplica.

2. Antes de utilizar el controlador para aprovisionar el recurso de AWS, el clúster debe cumplir requisitos específicos. Para obtener más información, consulte the section called “Equilibrio de carga de aplicaciones” y the section called “Equilibrio de carga de red”.

Migración de aplicaciones desde el Controlador de entrada de ALB obsoleto

En este tema se explica cómo migrar desde versiones obsoletas de controladores. Más específicamente, se describe cómo eliminar las versiones obsoletas del Controlador del equilibrador de carga de AWS.

- Las versiones obsoletas no se pueden actualizar. Primero debe eliminarlas y, a continuación, instalar una versión actual.

- Las versiones obsoleta incluyen lo siguiente:
  - Controlador de entradas de ALB de AWS para Kubernetes (“Controlador de entradas”), un predecesor del Controlador del equilibrador de carga de AWS.
  - Cualquier versión `0.1.x` del Controlador del equilibrador de carga de AWS

Eliminación de la versión obsoleta del controlador

> **Note**
>
> Es posible que haya instalado la versión obsoleta con Helm o manualmente con manifiestos de Kubernetes. Realice el procedimiento utilizando la herramienta con la que la instaló originalmente.

1. Si ha instalado el gráfico de Helm `incubator/aws-alb-ingress-controller`, desinstálelo.

   ```
   helm delete aws-alb-ingress-controller -n kube-system
   ```

2. Si tiene la versión `0.1.x` del gráfico `eks-charts/aws-load-balancer-controller` instalado, desinstálelo. La actualización de `0.1.x` a la versión `1.0.0` no funciona debido a la incompatibilidad con la versión de la API webhook.

   ```
   helm delete aws-load-balancer-controller -n kube-system
   ```

3. Verifique si el controlador se encuentra instalado actualmente.

   ```
   kubectl get deployment -n kube-system alb-ingress-controller
   ```

   Esta es la salida si el controlador no está instalado.

   ```
   Error from server (NotFound): deployments.apps "alb-ingress-controller" not found
   ```

   Esta es la salida si el controlador está instalado.

   ```
   NAME                   READY UP-TO-DATE AVAILABLE AGE
   alb-ingress-controller 1/1   1          1         122d
   ```

4. Ingrese el siguiente comando para eliminar el controlador.

```
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
```

Migración al Controlador del equilibrador de carga de AWS

Para migrar del Controlador de entrada de ALB para Kubernetes al Controlador del equilibrador de carga de AWS, haga lo siguiente:

1. Retire el controlador de entrada de ALB (consulte más arriba).
2. Instale el controlador del equilibrador de carga de AWS.
3. Agregue una política adicional al rol de IAM utilizado por el Controlador del equilibrador de carga de AWS. Esta política permite al LBC administrar los recursos creados por el Controlador de entrada de ALB para Kubernetes.
4. Descargue la política de IAM. Esta política permite al Controlador del equilibrador de carga de AWS administrar los recursos creados por el Controlador de entrada de ALB para Kubernetes. También puede ver la política.

```
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. Si su clúster está en las regiones de AWS AWS GovCloud (este de EE. UU.) o AWS GovCloud (oeste de EE. UU.), reemplace `arn:aws:` con `arn:aws-us-gov:`.

```
sed -i.bak -e 's|arn:aws:|arn:aws-us-gov:|' iam_policy_v1_to_v2_additional.json
```

6. Cree la política de IAM y anote el ARN devuelto.

```
aws iam create-policy \
  --policy-name AWSLoadBalancerControllerAdditionalIAMPolicy \
  --policy-document file://iam_policy_v1_to_v2_additional.json
```

7. Adjunte la política de IAM al rol de IAM utilizado por el Controlador del equilibrador de carga de AWS. Sustituya *`your-role-name`* por el nombre del rol, así como `AmazonEKSLoadBalancerControllerRole`.

Si creó el rol con `eksctl`, para encontrar el nombre del rol que se creó, abra la consola de AWS CloudFormation y seleccione la pila eksctl-***my-cluster***-addon-iamserviceaccount-kube-system-aws-load-balancer-controller. Seleccione la pestaña Recursos. El nombre del rol se encuentra en la columna de ID físicos.

```
aws iam attach-role-policy \
  --role-name your-role-name \
  --policy-arn arn:aws:iam::111122223333:policy/
AWSLoadBalancerControllerAdditionalIAMPolicy
```

Administración de CoredNS para DNS en clústeres de Amazon EKS

> **Tip**
>
> Con el modo automático de Amazon EKS, no necesita instalar ni actualizar los complementos de red. El modo automático ofrece capacidades para la conexión en red de pods y el equilibrio de carga.
> Para obtener más información, consulte Modo automático de EKS.

CoreDNS es un servidor de DNS flexible y extensible que puede servir como el DNS del clúster de Kubernetes. Al lanzar un clúster de Amazon EKS con al menos un nodo, se implementan dos réplicas de la imagen de CoreDNS de forma predeterminada, independientemente del número de nodos implementados en el clúster. Los pods de CoreDNS proporcionan resolución de nombres para todos los pods del clúster. Los pods de CoreDNS se pueden implementar en los nodos de Fargate si su clúster incluye un perfil de Fargate con un espacio de nombres que coincida con el espacio de nombres para la `deployment` de CoreDNS. A fin de obtener más información sobre CoreDNS, consulte Uso de CoreDNS para la detección de servicios en la documentación de Kubernetes.

Versiones de CoreDNS

En la siguiente tabla se muestra la versión más reciente del tipo de complemento de Amazon EKS para cada versión de Kubernetes.

Versión de Kubernetes	Versión de CoreDNS
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
1.30	v1.11.4-eksbuild.24
1.29	v1.11.4-eksbuild.24
1.28	v1.10.1-eksbuild.38

Important

Si administra este complemento, es posible que las versiones de la tabla no sean las mismas que las versiones autoadministradas disponibles. Para obtener más información acerca de la actualización de complementos autoadministrados, consulte the section called “Actualización (autoadministrado)”.

Consideraciones importantes sobre la actualización de CoreDNS

- Las actualizaciones de CoreDNS utilizan un PodDisruptionBudget para ayudar a mantener la disponibilidad del servicio DNS durante el proceso de actualización.
- Para mejorar la estabilidad y la disponibilidad de la implementación de CoreDNS, la versión `v1.9.3-eksbuild.6` y posteriores y `v1.10.1-eksbuild.3` se implementan con `PodDisruptionBudget`. Si ha implementado un `PodDisruptionBudget` existente, la actualización a estas versiones podría fallar. Si se produce un error en la actualización, se solucionará el problema al completar una de las siguientes tareas:
  - Al actualizar el complemento Amazon EKS, elija anular la configuración existente como opción de resolución de conflictos. Si ha hechos otros ajustes personalizados en la implementación, asegúrese de hacer una copia de seguridad de los ajustes antes de efectuar la actualización para poder volver a aplicar los demás ajustes personalizados después de la actualización.

  - Elimine el PodDisruptionBudget que ya tiene y vuelva a intentar la actualización.
- En la versión v1.9.3-eksbuild.3 y posteriores del complemento de EKS y v1.10.1-eksbuild.6 y posteriores, la implementación de CoreDNS establece readinessProbe para usar el punto de conexión /ready. Este punto de conexión se habilita en el archivo de configuración Corefile de CoreDNS.

  Si utiliza un Corefile personalizado, debe agregar el complemento ready a la configuración, para que el punto de conexión /ready esté activo en CoreDNS de modo que lo pueda utilizar la sonda.
- En las versiones del complemento de EKS v1.9.3-eksbuild.7 y posteriores, y v1.10.1-eksbuild.4 y posteriores, puede cambiar el objeto PodDisruptionBudget. Puede editar el complemento y cambiar esta configuración en Valores de configuración opcionales mediante los campos del siguiente ejemplo. En este ejemplo se muestra el objeto PodDisruptionBudget predeterminado.

  ```
  {
      "podDisruptionBudget": {
          "enabled": true,
          "maxUnavailable": 1
          }
  }
  ```

  Puede establecer maxUnavailable o minAvailable, pero no puede establecer ambos en un solo PodDisruptionBudget. Para obtener más información sobre PodDisruptionBudgets, consulte Especificación de un PodDisruptionBudget en la documentación de Kubernetes.

  Tenga en cuenta que si establece enabled como false, no se elimina PodDisruptionBudget. Después de establecer este campo como false, debe eliminar el objeto PodDisruptionBudget. Del mismo modo, si edita el complemento para que utilice una versión anterior (es decir, desactualiza el complemento) después de actualizarlo a una versión con un objeto PodDisruptionBudget, no se elimina PodDisruptionBudget. Para eliminar el objeto PodDisruptionBudget, puede ejecutar el siguiente comando:

  ```
  kubectl delete poddisruptionbudget coredns -n kube-system
  ```
- En las versiones complementarias de EKS v1.10.1-eksbuild.5 y posteriores, cambie la tolerancia predeterminada de node-role.kubernetes.io/master:NoSchedule a node-role.kubernetes.io/control-plane:NoSchedule para cumplir con el KEP 2067. Para

obtener más información sobre KEP 2067, consulteKEP-2067: Rename the kubeadm "master" label and taint en Kubernetes Enhancement Proposals (KEPs) en GitHub.

En las versiones del complemento de EKS `v1.8.7-eksbuild.8` y posteriores y `v1.9.3-eksbuild.9` y posteriores, ambas tolerancias están configuradas para que sean compatibles con todas las versiones de Kubernetes.

- En las versiones del complemento de EKS `v1.9.3-eksbuild.11` y `v1.10.1-eksbuild.7` y posteriores, la implementación de CoreDNS establece un valor predeterminado para `topologySpreadConstraints`. El valor predeterminado garantiza que los pods de CoreDNS se distribuyan entre las zonas de disponibilidad si hay nodos en varias zonas de disponibilidad disponibles. Puede establecer un valor personalizado que se utilizará en lugar del valor predeterminado. El valor predeterminado es el siguiente:

```
topologySpreadConstraints:
  - maxSkew: 1
    topologyKey: topology.kubernetes.io/zone
    whenUnsatisfiable: ScheduleAnyway
    labelSelector:
      matchLabels:
        k8s-app: kube-dns
```

Consideraciones sobre la actualización de CoreDNS **v1.11**

- En las versiones complementarias de EKS `v1.11.1-eksbuild.4` y posteriores, la imagen del contenedor está basada en una imagen básica mínima mantenida por Amazon EKS Distro, el cual contiene paquetes mínimos y no contiene intérpretes de comandos. Para obtener más información, consulte ¿Qué es Amazon EKS Distro? El uso y la solución de problemas de la imagen de CoreDNS siguen siendo los mismos.

Creación del complemento CoreDNS para Amazon EKS

Cree el complemento CoreDNS para Amazon EKS. Debe tener un clúster antes de crear el complemento. Para obtener más información, consulte the section called “Creación de un clúster ”.

1. Consulte qué versión del complemento está instalada en el clúster.

```
kubectl describe deployment coredns --namespace kube-system | grep coredns: | cut -
d : -f 3
```

Un ejemplo de salida sería el siguiente.

```
v1.10.1-eksbuild.13
```

2. Consulte qué tipo del complemento está instalado en el clúster. Según la herramienta con la que haya creado el clúster, es posible que actualmente no tenga instalado el tipo de complemento Amazon EKS en el clúster. Reemplace *my-cluster* por el nombre de su clúster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 addon.addonVersion --output text
```

Si se devuelve el número de versión, tiene el tipo de complemento de Amazon EKS instalado en el clúster y no es necesario que complete los pasos restantes del procedimiento. Si se devuelve un error, no tiene el tipo de complemento de Amazon EKS instalado en el clúster. Complete los pasos restantes de este procedimiento para instalarlo.

3. Guarde la configuración del complemento instalado actualmente.

```
kubectl get deployment coredns -n kube-system -o yaml > aws-k8s-coredns-old.yaml
```

4. Cree el complemento mediante la AWS CLI. Si desea utilizar Consola de administración de AWS o `eksctl` para crear el complemento, consultethe section called “Cómo crear un complemento” y especifique`coredns` para el nombre del complemento. Copie el comando que sigue en su dispositivo. Realice las siguientes modificaciones en el comando según sea necesario y, a continuación, ejecute el comando modificado.

- Reemplace *my-cluster* por el nombre de su clúster.
- Sustituya *v1.11.3-eksbuild.1* por la versión más reciente que aparece en la tabla de versiones más recientes correspondiente a la versión del clúster.

```
aws eks create-addon --cluster-name my-cluster --addon-name coredns --addon-version
 v1.11.3-eksbuild.1
```

Si ha aplicado una configuración personalizada al complemento actual que entra en conflicto con la configuración predeterminada del complemento de Amazon EKS, es posible que se

produzca un error en la creación. Si se produce un error en la creación, recibe un error que puede serle de utilidad para resolver el problema. Como alternativa, puede añadir `--resolve-conflicts OVERWRITE` al comando anterior. Esto permite que el complemento sobrescriba cualquier configuración personalizada existente. Una vez que haya creado el complemento, puede actualizarlo con la configuración personalizada.

5. Confirme que la versión más reciente del complemento de la versión de su clúster de Kubernetes se haya agregado al clúster. Reemplace *my-cluster* por el nombre de su clúster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 addon.addonVersion --output text
```

   Es posible que la creación del complemento tarde varios segundos en completarse.

   Un ejemplo de salida sería el siguiente.

```
v1.11.3-eksbuild.1
```

6. Si ha realizado ajustes personalizados en el complemento original, antes de crear el complemento de Amazon EKS, utilice la configuración que guardó en el paso anterior para actualizar el complemento de Amazon EKS con su configuración personalizada. Para obtener instrucciones sobre cómo actualizar el complemento, consulte the section called “Actualización (del complemento de EKS)”.

Actualización del complemento CoreDNS de Amazon EKS

Actualización del tipo de Amazon EKS del complemento. Si no ha agregado el complemento de Amazon EKS al clúster, agréguelo o consulte the section called “Actualización (autoadministrado)”.

Antes de comenzar, revise las consideraciones para la actualización. Para obtener más información, consulte the section called “Consideraciones importantes sobre la actualización de CoreDNS”.

1. Consulte qué versión del complemento está instalada en el clúster. Sustituya *my-cluster* por el nombre del clúster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 "addon.addonVersion" --output text
```

   Un ejemplo de salida sería el siguiente.

```
v1.10.1-eksbuild.13
```

Si la versión devuelta es la misma que la versión del clúster de Kubernetes en la tabla de versiones más recientes, significa que ya tiene la última versión instalada en el clúster y no necesita completar el resto de este procedimiento. Si recibe un error, en lugar de un número de versión en la salida, significa que no tiene el tipo de versión de Amazon EKS en el clúster. Debe crear el complemento antes de poder actualizarlo mediante este procedimiento.

2. Guarde la configuración del complemento instalado actualmente.

```
kubectl get deployment coredns -n kube-system -o yaml > aws-k8s-coredns-old.yaml
```

3. Actualice el complemento con la AWS CLI. Si desea utilizar Consola de administración de AWS o `eksctl` para actualizar el complemento, consulte the section called “Cómo actualizar un complemento”. Copie el comando que sigue en su dispositivo. Realice las siguientes modificaciones en el comando según sea necesario y, a continuación, ejecute el comando modificado.
   - Reemplace *`my-cluster`* por el nombre de su clúster.
   - Sustituya *`v1.11.3-eksbuild.1`* por la versión más reciente que aparece en la tabla de versiones más recientes correspondiente a la versión del clúster.
   - La opción de `--resolve-conflicts`*`PRESERVE`* conserva los valores de configuración existentes del complemento. Si ha establecido valores personalizados para la configuración del complemento y no utiliza esta opción, Amazon EKS sobrescribe los valores con los valores predeterminados. Si utiliza esta opción, le recomendamos que pruebe cualquier cambio de campo y valor en un clúster que no sea de producción antes de actualizar el complemento del clúster de producción. Si cambia este valor a `OVERWRITE`, todas las configuraciones cambiarán a los valores predeterminados de Amazon EKS. Si ha establecido valores personalizados para cualquier configuración, es posible que se sobrescriban con los valores predeterminados de Amazon EKS. Si cambia este valor a `none`, Amazon EKS no cambia el valor de ninguna configuración, pero la actualización podría fallar. Si se produce un error en la actualización, recibe un mensaje de error que lo ayuda a resolver el conflicto.
   - Si no va a actualizar un ajuste de configuración, elimine `--configuration-values '{`*`"replicaCount":3`*`}'` del comando. Si está actualizando una configuración, reemplace *`"replicaCount":3`* por la configuración que desee establecer. En este ejemplo, el número de réplicas de CoreDNS se establece en `3`. El valor que especifique debe ser válido para el esquema de configuración. Si no conoce el esquema de configuración, ejecute `aws eks`

`describe-addon-configuration --addon-name coredns --addon-version` *`v1.11.3-eksbuild.1`* y reemplace *`v1.11.3-eksbuild.1`* por el número de versión del complemento cuya configuración desea ver. El esquema se devuelve en la salida. Si ya tiene alguna configuración personalizada, quiere eliminarla por completo y volver a establecer los valores de todos los ajustes a los valores predeterminados de Amazon EKS, elimine *`"replicaCount":3`* del comando para que quede vacío `{}`. Para obtener más información sobre la configuración de CoreDNS, consulte Customizing DNS Service en la documentación de Kubernetes.

```
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}'
```

La actualización puede tardar varios segundos en completarse.

4. Confirme que la versión del complemento se ha actualizado. Reemplace *`my-cluster`* por el nombre de su clúster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns
```

La actualización puede tardar varios segundos en completarse.

Un ejemplo de salida sería el siguiente.

```
{
    "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}"
    }
```

```
}
```

Actualización del complemento autoadministrado CoreDNS de Amazon EKS

> **Important**
>
> Recomendamos agregar el tipo de complemento de Amazon EKS al clúster en lugar de utilizar el tipo de complemento autoadministrado. Si no está familiarizado con la diferencia entre los tipos, consulte the section called “Complementos de Amazon EKS”. Para obtener más información acerca de cómo agregar un complemento de Amazon EKS al clúster, consulte the section called “Cómo crear un complemento”. Si no puede utilizar el complemento de Amazon EKS, le recomendamos que envíe una pregunta sobre los motivos por los que no puede hacerlo al repositorio de GitHub de la hoja de ruta de contenedores.

Antes de comenzar, revise las consideraciones para la actualización. Para obtener más información, consulte the section called “Consideraciones importantes sobre la actualización de CoreDNS”.

1. Confirme que tiene instalado en el clúster el tipo de complemento autoadministrado. Reemplace *my-cluster* por el nombre de su clúster.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
    addon.addonVersion --output text
   ```

   Si se devuelve un mensaje de error, tiene el tipo de complemento autoadministrado instalado en el clúster. Complete los pasos restantes de este procedimiento. Si se devuelve el número de versión, tiene el tipo de complemento de Amazon EKS instalado en el clúster. Para actualizar el tipo de Amazon EKS del complemento, siga el procedimiento que aparece en Actualizar el complemento CoreDNS de Amazon EKS, en lugar de realizar este procedimiento. Si no está familiarizado con las diferencias entre los tipos de complementos, consulte the section called “Complementos de Amazon EKS”.
2. Consulte qué versión de la imagen del contenedor está instalada actualmente en el clúster.

   ```
   kubectl describe deployment coredns -n kube-system | grep Image | cut -d ":" -f 3
   ```

   Un ejemplo de salida sería el siguiente.

```
v1.8.7-eksbuild.2
```

3. Si la versión actual de CoreDNS es `v1.5.0` o una posterior, pero anterior a la versión que aparece en la tabla de versiones de CoreDNS, omita este paso. Si su versión actual es anterior a `1.5.0`, debe modificar `ConfigMap` para que CoreDNS utilice el complemento de reenvío, en lugar del complemento proxy.

   a. Abra el `ConfigMap` con el siguiente comando.

   ```
   kubectl edit configmap coredns -n kube-system
   ```

   b. Sustituya el `proxy` en la línea siguiente por `forward`. Guarde el archivo y salga del editor.

   ```
   proxy . /etc/resolv.conf
   ```

4. Si implementó su clúster en Kubernetes `1.17` o una versión anterior inicialmente, es posible que deba eliminar una línea interrumpida de su manifiesto de CoreDNS.

   > **Important**
   >
   > Debe completar este paso antes de actualizar a la versión `1.7.0` de CoreDNS, pero se recomienda que complete este paso incluso si está actualizando a una versión anterior.

   a. Verifique si su manifiesto CoreDNS cuenta con la línea.

   ```
   kubectl get configmap coredns -n kube-system -o jsonpath='{$.data.Corefile}' |
    grep upstream
   ```

   Si no se devuelve un resultado, el manifiesto no cuenta con la línea y puede pasar al siguiente paso para actualizar CoreDNS. Si se devuelve el resultado, debe eliminar la línea.

   b. Edite el `ConfigMap` con el siguiente comando, al eliminar la línea en el archivo que tiene la palabra `upstream` en ella. No realice más cambios en el archivo. Una vez que elimine la línea, guarde los cambios.

   ```
   kubectl edit configmap coredns -n kube-system -o yaml
   ```

5. Recupere su versión de la imagen de CoreDNS actual:

```
kubectl describe deployment coredns -n kube-system | grep Image
```

Un ejemplo de salida sería el siguiente.

```
602401143452.dkr.ecr.region-code.amazonaws.com/eks/coredns:v1.8.7-eksbuild.2
```

6. Si está actualizando a CoreDNS `1.8.3` o una versión posterior, debe agregar el permiso `endpointslices` al `clusterrole` de Kubernetes `system:coredns`.

```
kubectl edit clusterrole system:coredns -n kube-system
```

Agregue las siguientes líneas en las líneas de permisos existentes en la sección `rules` del archivo.

```
[...]
- apiGroups:
  - discovery.k8s.io
  resources:
  - endpointslices
  verbs:
  - list
  - watch
[...]
```

7. Para actualizar el complemento CoreDNS, reemplace *602401143452* y *region-code* por los valores de la salida devuelta en un paso anterior. Reemplace *v1.11.3-eksbuild.1* por la versión de CoreDNS que aparece en la tabla de versiones más recientes correspondiente a la versión de Kubernetes.

```
kubectl set image deployment.apps/coredns -n kube-system
 coredns=602401143452.dkr.ecr.region-code.amazonaws.com/eks/coredns:v1.11.3-
eksbuild.1
```

Un ejemplo de salida sería el siguiente.

```
deployment.apps/coredns image updated
```

8. Vuelva a comprobar la versión de la imagen del contenedor para confirmar que se actualizó a la versión que especificó en el paso anterior.

```
kubectl describe deployment coredns -n kube-system | grep Image | cut -d ":" -f 3
```

Un ejemplo de salida sería el siguiente.

```
v1.11.3-eksbuild.1
```

Escalabilidad de los pods de CoredNS para un tráfico de DNS elevado

Al lanzar un clúster de Amazon EKS con al menos un nodo, se implementan dos réplicas de la imagen de CoreDNS de forma predeterminada, independientemente del número de nodos implementados en el clúster. Los pods de CoreDNS proporcionan resolución de nombres para todos los pods del clúster. Las aplicaciones utilizan la resolución de nombres para conectarse a los pods y los servicios del clúster, así como para conectarse a los servicios fuera del clúster. A medida que aumenta el número de solicitudes de resolución de nombres (consultas) procedentes de los pods, los pods de CoreDNS se sobrecargan y se ralentizan, y se rechazan las solicitudes que los pods no pueden gestionar.

Para gestionar la mayor carga de los pods de CoreDNS, considere la posibilidad de utilizar un sistema de escalado automático para CoreDNS. Amazon EKS puede administrar el escalado automático de la implementación de CoreDNS en la versión del complemento CoreDNS de EKS. Este escalador automático de CoreDNS monitorea continuamente el estado del clúster, incluida la cantidad de nodos y núcleos de CPU. En función de esa información, el controlador adaptará dinámicamente el número de réplicas de la implementación de CoreDNS en un clúster de Amazon EKS. Esta característica funciona para CoreDNS `v1.9` y versiones posteriores. Para obtener más información acerca de qué versiones son compatibles con el escalado automático de CoreDNS, consulte la siguiente sección.

El sistema administra automáticamente las réplicas de CoreDNS mediante una fórmula dinámica basada en el número de nodos y núcleos de CPU del clúster, calculada como el máximo de (numberOfNodes dividido por 16) y (numberOfCPUCores dividido por 256). Evalúa la demanda durante los periodos de pico de 10 minutos, escalando verticalmente de inmediato cuando es necesario para manejar el aumento de la carga de consultas de DNS, mientras que reduce verticalmente de forma gradual las réplicas en un 33 % cada 3 minutos para mantener la estabilidad del sistema y evitar interrupciones.

Recomendamos utilizar esta característica junto con otras prácticas recomendadas de escalado automático de clústeres de EKS para mejorar la disponibilidad general de las aplicaciones y la escalabilidad de los clústeres.

Requisitos previos

Para que Amazon EKS escale la implementación de CoreDNS, se deben cumplir tres requisitos previos:

- Debe utilizar la versión del complemento de EKS de CoreDNS.
- El clúster debe ejecutar al menos las versiones de clúster y plataforma mínimas.
- El clúster debe ejecutar al menos la versión mínima del complemento de EKS de CoreDNS.

Versión mínima del clúster

El escalado automático de CoreDNS se lleva a cabo mediante un nuevo componente en el plano de control del clúster, administrado por Amazon EKS. Por este motivo, debe actualizar el clúster a una versión de EKS que sea compatible con la versión de la plataforma mínima que incluya el nuevo componente.

Un nuevo clúster de Amazon EKS. Para implementar uno, consulte Introducción. El clúster debe estar ejecutando una de las versiones de Kubernetes y de la plataforma que se enumeran en la siguiente tabla o una versión posterior. Tenga en cuenta que también se admite cualquier versión de Kubernetes y de la plataforma posterior a las enumeradas. Para comprobar la versión actual de Kubernetes, reemplace *my-cluster* en el siguiente comando por el nombre del clúster y, a continuación, ejecute el comando modificado:

```
aws eks describe-cluster --name my-cluster --query cluster.version --output text
```

Versión de Kubernetes	Versión de la plataforma
No se muestra	Todas las versiones de la plataforma
1.29.3	eks.7
1.28.8	eks.13
1.27.12	eks.17

Versión de Kubernetes	Versión de la plataforma
1.26.15	eks.18

> **Note**
>
> También se admiten todas las versiones de la plataforma de Kubernetes posteriores; por ejemplo, la versión de Kubernetes `1.30` de `eks.1` y posteriores.

Versión mínima del complemento de EKS

Versión de Kubernetes	1.29	1.28
	v1.11.1-eksbuild.9	v1.10.1-eksbuild.11

Configuración del escalado automático de CoreDNS en la Consola de administración de AWS

1. Asegúrese de que el clúster sea igual o superior a la versión mínima del clúster.

   Amazon EKS actualiza automáticamente los clústeres entre versiones de la plataforma de la misma versión de Kubernetes y no puede iniciar este proceso. En su lugar, puede actualizar el clúster a la siguiente versión de Kubernetes, y el clúster se actualizará a esa versión de K8s y a la versión de la plataforma más reciente.

   Las nuevas versiones de Kubernetes suelen presentar cambios significativos. Por ende, recomendamos que pruebe el comportamiento de las aplicaciones con un clúster separado de la nueva versión de Kubernetes antes de efectuar la actualización en los clústeres de producción.

   Para actualizar un clúster a una nueva versión de Kubernetes, siga el procedimiento descrito en Actualización del clúster existente a la nueva versión de Kubernetes.
2. Asegúrese de tener el complemento de EKS para CoreDNS, no la implementación autoadministrada de CoreDNS.

   Según la herramienta con la que haya creado el clúster, es posible que actualmente no tenga instalado el tipo de complemento Amazon EKS en el clúster. Para ver qué tipo de complemento

está instalado en el clúster, puede ejecutar el siguiente comando. Reemplace `my-cluster` por el nombre del clúster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 addon.addonVersion --output text
```

Si se devuelve el número de versión, tiene el tipo de complemento de Amazon EKS instalado en el clúster y puede continuar con el siguiente paso. Si se devuelve un error, no tiene el tipo de complemento de Amazon EKS instalado en el clúster. Complete los pasos restantes del procedimiento Creación de un complemento CoreDNS de Amazon EKS para reemplazar la versión autoadministrada por el complemento de Amazon EKS.

3. Asegúrese de que la versión del complemento de EKS para CoreDNS sea igual o posterior a la versión mínima del complemento de EKS.

   Consulte qué versión del complemento está instalada en el clúster. Puede verificarlo en la Consola de administración de AWS o ejecutar el siguiente comando:

   ```
   kubectl describe deployment coredns --namespace kube-system | grep coredns: | cut -
   d : -f 3
   ```

   Un ejemplo de salida sería el siguiente.

   ```
   v1.10.1-eksbuild.13
   ```

   Compare esta versión con la versión mínima del complemento de EKS de la sección anterior. Si es necesario, actualice el complemento de EKS a una versión superior; para ello, siga el procedimiento Actualización el complemento CoreDNS de Amazon EKS.

4. Agregue la configuración de escalado automático a los ajustes de configuración opcionales del complemento de EKS.

   a. Abra la consola de Amazon EKS.

   b. En el panel de navegación izquierdo, seleccione Clusters (Clústeres) y, a continuación, seleccione el nombre del clúster para el que desea configurar el complemento.

   c. Elija la pestaña Complementos.

   d. Seleccione la casilla situada en la parte superior derecha del cuadro del complemento CoreDNS y, a continuación, elija Editar.

   e. En la página Configure CoreDNS:

i. Seleccione la Versión que desea utilizar. Le recomendamos que mantenga la misma versión que en el paso anterior y que actualice la versión y la configuración en acciones separadas.

ii. Seleccione Ajustes de configuración opcionales.

iii. Introduzca la clave JSON `"autoscaling":` y el valor de un objeto JSON anidado con una clave `"enabled":` y un valor `true` en Valores de configuración. El texto resultante debe ser un objeto JSON válido. Si esta clave y este valor son los únicos datos del cuadro de texto, rodee la clave y el valor entre corchetes `{ }`. En el siguiente ejemplo, se muestra que el escalado automático está activado:

```
{
  "autoScaling": {
    "enabled": true
  }
}
```

iv. (Opcional) Puede proporcionar valores mínimos y máximos a los que el escalado automático pueda escalar la cantidad de pods de CoreDNS.

En el siguiente ejemplo, se muestra que el escalado automático está activado y que todas las claves opcionales tienen valores. Recomendamos que la cantidad mínima de pods de CoreDNS sea siempre superior a 2 para proporcionar resiliencia al servicio de DNS del clúster.

```
{
  "autoScaling": {
    "enabled": true,
    "minReplicas": 2,
    "maxReplicas": 10
  }
}
```

f. Para aplicar la nueva configuración mediante la sustitución de los pods de CoreDNS, seleccione Guardar cambios.

Amazon EKS aplica los cambios a los complementos de EKS mediante el despliegue de la implementación de Kubernetes para CoreDNS. Puede hacer un seguimiento del estado del lanzamiento en el historial de actualizaciones en la Consola de administración de AWS y con `kubectl rollout status deployment/coredns --namespace kube-system`.

`kubectl rollout` tiene los siguientes comandos:

```
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
```

Si la implementación lleva demasiado tiempo, Amazon EKS la anulará y se agregará al historial de actualizaciones del complemento un mensaje con el tipo de actualización del complemento y el estado Fallido. Para investigar cualquier problema, comience por el historial del despliegue y ejecute `kubectl logs` en un pod de CoreDNS para ver los registros de CoreDNS.

5. Si el estado de la nueva entrada del Historial de actualizaciones es Correcto, significa que el despliegue se ha completado y que el complemento está utilizando la nueva configuración en todos los pods de CoreDNS. A medida que cambia la cantidad de nodos y los núcleos de CPU de los nodos del clúster, Amazon EKS escala la cantidad de réplicas de la implementación de CoreDNS.

Configuración del escalado automático de CoreDNS en la Interfaz de la línea de comandos de AWS

1. Asegúrese de que el clúster sea igual o superior a la versión mínima del clúster.

   Amazon EKS actualiza automáticamente los clústeres entre versiones de la plataforma de la misma versión de Kubernetes y no puede iniciar este proceso. En su lugar, puede actualizar el clúster a la siguiente versión de Kubernetes, y el clúster se actualizará a esa versión de K8s y a la versión de la plataforma más reciente.

   Las nuevas versiones de Kubernetes suelen presentar cambios significativos. Por ende, recomendamos que pruebe el comportamiento de las aplicaciones con un clúster separado de la nueva versión de Kubernetes antes de efectuar la actualización en los clústeres de producción.

   Para actualizar un clúster a una nueva versión de Kubernetes, siga el procedimiento descrito en Actualización del clúster existente a la nueva versión de Kubernetes.
2. Asegúrese de tener el complemento de EKS para CoreDNS, no la implementación autoadministrada de CoreDNS.

Según la herramienta con la que haya creado el clúster, es posible que actualmente no tenga instalado el tipo de complemento Amazon EKS en el clúster. Para ver qué tipo de complemento está instalado en el clúster, puede ejecutar el siguiente comando. Reemplace `my-cluster` por el nombre del clúster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 addon.addonVersion --output text
```

Si se devuelve el número de versión, tiene el tipo de complemento de Amazon EKS instalado en el clúster. Si se devuelve un error, no tiene el tipo de complemento de Amazon EKS instalado en el clúster. Complete los pasos restantes del procedimiento Creación de un complemento CoreDNS de Amazon EKS para reemplazar la versión autoadministrada por el complemento de Amazon EKS.

3. Asegúrese de que la versión del complemento de EKS para CoreDNS sea igual o posterior a la versión mínima del complemento de EKS.

   Consulte qué versión del complemento está instalada en el clúster. Puede verificarlo en la Consola de administración de AWS o ejecutar el siguiente comando:

   ```
   kubectl describe deployment coredns --namespace kube-system | grep coredns: | cut -
   d : -f 3
   ```

   Un ejemplo de salida sería el siguiente.

   ```
   v1.10.1-eksbuild.13
   ```

   Compare esta versión con la versión mínima del complemento de EKS de la sección anterior. Si es necesario, actualice el complemento de EKS a una versión superior; para ello, siga el procedimiento Actualización el complemento CoreDNS de Amazon EKS.

4. Agregue la configuración de escalado automático a los ajustes de configuración opcionales del complemento de EKS.

   Ejecute el siguiente comando de la CLI de AWS. Reemplace `my-cluster` por el nombre del clúster y el ARN del rol de IAM por el rol que va a usar.

   ```
   aws eks update-addon --cluster-name my-cluster --addon-name coredns \
   ```

```
    --resolve-conflicts PRESERVE --configuration-values '{"autoScaling":
{"enabled":true}}'
```

Amazon EKS aplica los cambios a los complementos de EKS mediante el despliegue de la implementación de Kubernetes para CoreDNS. Puede hacer un seguimiento del estado del lanzamiento en el historial de actualizaciones en la Consola de administración de AWS y con `kubectl rollout status deployment/coredns --namespace kube-system`.

`kubectl rollout` tiene los siguientes comandos:

```
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
```

Si la implementación lleva demasiado tiempo, Amazon EKS la anulará y se agregará al historial de actualizaciones del complemento un mensaje con el tipo de actualización del complemento y el estado Fallido. Para investigar cualquier problema, comience por el historial del despliegue y ejecute `kubectl logs` en un pod de CoreDNS para ver los registros de CoreDNS.

5. (Opcional) Puede proporcionar valores mínimos y máximos a los que el escalado automático pueda escalar la cantidad de pods de CoreDNS.

   En el siguiente ejemplo, se muestra que el escalado automático está activado y que todas las claves opcionales tienen valores. Recomendamos que la cantidad mínima de pods de CoreDNS sea siempre superior a 2 para proporcionar resiliencia al servicio de DNS del clúster.

   ```
   aws eks update-addon --cluster-name my-cluster --addon-name coredns \
       --resolve-conflicts PRESERVE --configuration-values '{"autoScaling":
   {"enabled":true,"minReplicas":2,"maxReplicas":10}}'
   ```

6. Consulte el estado de la actualización del complemento mediante la ejecución del siguiente comando:

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns
   ```

Si ve esta línea: `"status": "ACTIVE"`, significa que la implementación se ha completado y que el complemento está usando la nueva configuración en todos los pods de CoreDNS. A medida que cambia la cantidad de nodos y los núcleos de CPU de los nodos del clúster, Amazon EKS escala la cantidad de réplicas de la implementación de CoreDNS.

Supervisión de la resolución de DNS de Kubernetes con las métricas de CoreDNS

CoreDNS como un complemento de EKS expone las métricas de CoreDNS en el puerto `9153` con el formato Prometheus en el servicio de `kube-dns`. Puede utilizar Prometheus, el agente de Amazon CloudWatch o cualquier otro sistema compatible para raspar (recopilar) estas métricas.

Para ver un ejemplo de configuración de raspado que sea compatible tanto con Prometheus como con el agente de CloudWatch, consulte CloudWatch agent configuration for Prometheus en la Guía del usuario de Amazon CloudWatch.

Administración de **`kube-proxy`** en clústeres de Amazon EKS

> **Tip**
>
> Con el modo automático de Amazon EKS, no necesita instalar ni actualizar los complementos de red. El modo automático ofrece capacidades para la conexión en red de pods y el equilibrio de carga.
> Para obtener más información, consulte Modo automático de EKS.

Recomendamos agregar el tipo de complemento de Amazon EKS al clúster en lugar de utilizar el tipo de complemento autoadministrado. Si no está familiarizado con la diferencia entre los tipos, consulte the section called “Complementos de Amazon EKS”. Para obtener más información acerca de cómo agregar un complemento de Amazon EKS al clúster, consulte the section called “Cómo crear un complemento”. Si no puede usar el complemento de Amazon EKS, le recomendamos que envíe una pregunta sobre los motivos por los que no puede hacerlo al repositorio de GitHub de la hoja de ruta de contenedores.

El complemento `kube-proxy` se implementa en cada nodo de Amazon EC2 de su clúster de Amazon EKS. Mantiene las reglas de red en los nodos y permite la comunicación de red con los pods. El complemento no se implementa en los nodos de Fargate del clúster. Para obtener más información, consulte kube-proxy en la documentación de Kubernetes.

Instalar como complemento de Amazon EKS

kube-proxy Versiones de

En la siguiente tabla se muestra la versión más reciente del tipo de complemento de Amazon EKS para cada versión de Kubernetes.

Versión de Kubernetes	**kube-proxy** Versión
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**
>
> Una versión anterior de la documentación era incorrecta. Las versiones `v1.28.5`, `v1.27.9`, y `v1.26.12` de `kube-proxy` no están disponibles.
> Si administra este complemento, es posible que las versiones de la tabla no sean las mismas que las versiones autoadministradas disponibles.

kube-proxy Imagen de contenedor de

La imagen del contenedor `kube-proxy` está basada en una imagen básica mínima mantenida por Amazon EKS Distro, que contiene paquetes mínimos y no contiene intérpretes de comandos. Para obtener más información, consulte ¿Qué es Amazon EKS Distro?

En la siguiente tabla, se enumera la versión más reciente disponible de la imagen de contenedor `kube-proxy` autoadministrada para cada versión del clúster de Amazon EKS.

Versión	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

Cuando actualiza un tipo de complemento de Amazon EKS, especifica una versión de complemento de Amazon EKS válida, que puede ser una versión que no aparece en esta tabla. Esto se debe a que las versiones del complemento de Amazon EKS no siempre coinciden con las versiones de imágenes del contenedor especificadas al actualizar el tipo autoadministrado de este complemento. Al actualizar el tipo autoadministrado de este complemento, se especifica una versión válida de la imagen del contenedor que aparece en esta tabla.

Actualización del complemento autoadministrado **`kube-proxy`** de Kubernetes

> **Important**
>
> Recomendamos agregar el tipo de complemento de Amazon EKS al clúster en lugar de utilizar el tipo de complemento autoadministrado. Si no está familiarizado con la diferencia entre los tipos, consulte the section called “Complementos de Amazon EKS”. Para obtener más información acerca de cómo agregar un complemento de Amazon EKS al clúster, consulte the section called “Cómo crear un complemento”. Si no puede usar el complemento de Amazon EKS, le recomendamos que envíe una pregunta sobre los motivos por los que no puede hacerlo al repositorio de GitHub de la hoja de ruta de contenedores.

Requisitos previos

- Un clúster existente de Amazon EKS. Para implementar uno, consulte Introducción.

Consideraciones

- Kube-proxy en un clúster de Amazon EKS tiene la misma política de compatibilidad y sesgo que Kubernetes. Aprenda cómo realizar la Comprobación de la compatibilidad de la versión del complemento de Amazon EKS con un clúster.

  1. Confirme que tiene instalado en el clúster el tipo de complemento autoadministrado. Reemplace *my-cluster* por el nombre de su clúster.

     ```
     aws eks describe-addon --cluster-name my-cluster --addon-name kube-proxy --query
      addon.addonVersion --output text
     ```

     Si se devuelve un mensaje de error, tiene el tipo de complemento autoadministrado instalado en el clúster. Los pasos restantes de este tema son para actualizar el tipo de complemento autoadministrado. Si se devuelve el número de versión, tiene el tipo de complemento de Amazon EKS instalado en el clúster. Para actualizarlo, siga el procedimiento que aparece en Actualización del complemento de Amazon EKS, en lugar del procedimiento de este tema. Si no está familiarizado con las diferencias entre los tipos de complementos, consulte the section called “Complementos de Amazon EKS”.

  2. Consulte qué versión de la imagen del contenedor está instalada actualmente en el clúster.

     ```
     kubectl describe daemonset kube-proxy -n kube-system | grep Image
     ```

     Un ejemplo de salida sería el siguiente.

     ```
     Image:    602401143452.dkr.ecr.region-code.amazonaws.com/eks/kube-proxy:v1.29.1-
     eksbuild.2
     ```

     En el ejemplo de resultado, *v1.29.1-eksbuild.2* es la versión instalada en el clúster.

  3. Para actualizar el complemento de kube-proxy, sustituya *602401143452* y *region-code* por los valores obtenidos en el paso anterior. Sustituya *v1.30.6-eksbuild.3* por la versión de kube-proxy que aparece en la tabla Versión de imagen de contenedor kube-proxy autoadministrada más reciente disponible para cada versión de clúster de Amazon EKS.

> **Important**
>
> Los manifiestos de cada tipo de imagen son diferentes y no son compatibles entre los tipos de imagen predeterminados o mínimos. Debe utilizar el mismo tipo de imagen que la imagen anterior para que el punto de entrada y los argumentos coincidan.

```
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
```

Un ejemplo de salida sería el siguiente.

```
daemonset.apps/kube-proxy image updated
```

4. Confirme que la nueva versión ya esté instalada en el clúster.

```
kubectl describe daemonset kube-proxy -n kube-system | grep Image | cut -d ":" -f 3
```

Un ejemplo de salida sería el siguiente.

```
v1.30.0-eksbuild.3
```

5. Si utiliza nodos `x86` y `Arm` en el mismo clúster y su clúster se implementó antes del 17 de agosto de 2020. A continuación, edite el manifiesto de `kube-proxy` a fin de incluir un selector de nodos para varias arquitecturas de hardware con el siguiente comando. Esta es una operación que se realiza una vez. Después de agregar el selector al manifiesto, no es necesario que lo agregue cada vez que realiza una actualización del complemento. Si el clúster se implementó a partir del 17 de agosto de 2020, `kube-proxy` ya cuenta con capacidad de varias arquitecturas.

```
kubectl edit -n kube-system daemonset/kube-proxy
```

Agregue el siguiente selector de nodos al archivo en el editor y guárdelo. Para ver un ejemplo de dónde incluir este texto en el editor, consulte el archivo de manifiesto de CNI en GitHub. Esto permite a Kubernetes extraer la imagen de hardware correcta según la arquitectura de hardware del nodo.

```
- key: "kubernetes.io/arch"
  operator: In
  values:
  - amd64
  - arm64
```

6. Si su clúster se creó inicialmente con la versión de Kubernetes `1.14` o posterior, puede omitir este paso porque `kube-proxy` ya incluye esta `Affinity Rule`. Si creó inicialmente un clúster de Amazon EKS con versión `1.13` o anterior de Kubernetes y tiene la intención de utilizar nodos de Fargate en el clúster, edite su manifiesto de `kube-proxy` para incluir una regla `NodeAffinity` a fin de evitar que los pods de `kube-proxy` programen en los nodos de Fargate. Esta es una edición que se realiza una vez. Después de agregar el `Affinity Rule` al manifiesto, no es necesario que lo agregue cada vez que realiza una actualización del complemento. Edite el DaemonSet de `kube-proxy`.

   ```
   kubectl edit -n kube-system daemonset/kube-proxy
   ```

   Agregue la siguiente `Affinity Rule` a la sección `spec` del DaemonSet del archivo en el editor y guárdelo. Para ver un ejemplo de dónde incluir este texto en el editor, consulte el archivo de manifiesto de CNI en GitHub.

   ```
   - key: eks.amazonaws.com/compute-type
     operator: NotIn
     values:
     - fargate
   ```

Cómo implementar cargas de trabajo y complementos en Amazon EKS

Sus cargas de trabajo se implementan en contenedores, que se implementan en pods en Kubernetes. Un pod incluye uno o más contenedores. Normalmente, uno o más pods que proporcionan el mismo servicio se implementan en un servicio de Kubernetes. Una vez que haya implementado varios pods que proporcionan el mismo servicio, puede:

- Visualizar información acerca de las cargas de trabajo que se ejecutan en cada uno de los clústeres mediante la Consola de administración de AWS.
- Escalar o reducir verticalmente pods con el Escalador automático vertical de pods de Kubernetes.
- Escalar horizontalmente el número de pods necesarios para satisfacer la demanda hacia arriba o hacia abajo con el Escalador automático horizontal de pods de Kubernetes.
- Crear un equilibrador de carga de red externo (para pods accesibles a través de Internet) o interno (para pods privados) a fin de equilibrar el tráfico de red entre los pods. El balanceador de carga enruta el tráfico en la capa 4 del modelo OSI.
- Crear un Equilibrador de carga de aplicación para equilibrar el tráfico de aplicaciones entre los pods. El equilibrador de carga de aplicaciones enruta el tráfico en la capa 7 del modelo OSI.
- Si es la primera vez que utiliza Kubernetes, este tema lo ayudará a implementar una aplicación de muestra.
- Puede: Restringir las direcciones IP que se pueden asignar a un servicio por `externalIPs`.

Implementación de una aplicación de muestra en Linux

En este tema, implementará una aplicación de muestra a su clúster en nodos de Linux.

Requisitos previos

- Un clúster de Kubernetes existente que tenga como mínimo un nodo. Si no dispone de un clúster de Amazon EKS existente, puede implementar uno mediante una de las guías de Introducción.
- `Kubectl` instalado en su equipo. Para obtener más información, consulte the section called “Configure kubectl y eksctl”.
- `Kubectl` configurado para comunicarse con el clúster. Para obtener más información, consulte the section called “Acceso al clúster con kubectl”.

- Si planea implementar su carga de trabajo de muestra en Fargate, debe tener una perfil de Fargate que incluya el mismo espacio de nombres creado en este tutorial, que es `eks-sample-app`, a menos que cambie el nombre. Si utilizó una de las guías de Introducción para crear el clúster, tendrá que crear un nuevo perfil o agregar el espacio de nombres a su perfil existente, porque el perfil creado en las guías de introducción no especifica el espacio de nombres utilizado en este tutorial. La VPC debe tener también al menos una subred privada.

Aunque muchas variables se pueden cambiar en los siguientes pasos, recomendamos cambiar solo los valores de las variables si se especifican. Una vez que conozca mejor los pods, las implementaciones y los servicios de Kubernetes, puede experimentar mediante el cambio de otros valores.

Creación de un espacio de nombres de

Un espacio de nombres permite agrupar recursos en Kubernetes. Para obtener más información, consulte Espacios de nombres en la documentación de Kubernetes. Si planea implementar su aplicación de muestra para Simplificación de la administración de computación con AWS Fargate, asegúrese de que el valor de `namespace` en su Definición de qué pods usarán AWS Fargate cuando se lancen sea `eks-sample-app`.

```
kubectl create namespace eks-sample-app
```

Cree una implementación de Kubernetes

Cree una implementación de Kubernetes. Esta implementación de muestra extrae una imagen de contenedor de un repositorio público e implementa tres réplicas (pods individuales) de ella en su clúster. Para obtener más información, consulte Implementaciones en la documentación de Kubernetes.

1. Guarde los siguientes contenidos en un archivo llamado `eks-sample-deployment.yaml`. Los contenedores de la aplicación de muestra no utilizan almacenamiento en red, pero es posible que tenga aplicaciones que lo necesiten. Para obtener más información, consulte Almacenamiento de datos de aplicaciones.
   - Los `values` `amd64` o `arm64` en `kubernetes.io/arch` significan que la aplicación se puede implementar en cualquiera de las arquitecturas de hardware (si tiene ambos en el clúster). Esto es posible porque esta imagen es una imagen de arquitectura múltiple, pero no todas lo son. Puede determinar la arquitectura de hardware con la que se admite la imagen a partir de los

detalles de imagen en el repositorio del que lo está sacando. Al implementar imágenes que no admiten un tipo de arquitectura de hardware o en las que no desea que se implemente la imagen, elimine ese tipo del manifiesto. Para obtener más información, consulte Etiquetas, anotaciones y taints conocidas en la documentación de Kubernetes.

- `kubernetes.io/os: linux` `nodeSelector` significa que si tuviera nodos Linux y Windows (por ejemplo) en el clúster, la imagen solo se implementaría en nodos Linux. Para obtener más información, consulte Etiquetas, anotaciones y taints conocidas en la documentación de Kubernetes.

```
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. Aplique el manifiesto de implementación al clúster.

```
kubectl apply -f eks-sample-deployment.yaml
```

Crear un servicio

Un servicio le permite acceder a todas las réplicas a través de una única dirección IP o nombre. Para obtener más información, consulte Servicio en la documentación de Kubernetes. Aunque no se ha implementado en la aplicación de muestra, si tiene aplicaciones que necesitan interactuar con otros servicios de AWS, recomendamos que cree cuentas de servicio de Kubernetes para sus pods y las asocie a cuentas de AWS IAM. Al especificar cuentas de servicio, los pods tienen solo los permisos mínimos que especifica para interactuar con otros servicios. Para obtener más información, consulte the section called “Credenciales con IRSA”.

1. Guarde los siguientes contenidos en un archivo llamado `eks-sample-service.yaml`. Kubernetes asigna al servicio su propia dirección IP a la que se puede acceder solo desde dentro del clúster. Para acceder al servicio desde fuera del clúster, implemente el Controlador del equilibrador de carga de AWS para que equilibre la carga de la aplicación o del tráfico de red del servicio.

```
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. Aplique el manifiesto de servicio al clúster.

```
kubectl apply -f eks-sample-service.yaml
```

Revise los recursos creados

1. Consulte todos los recursos que existen en el espacio de nombres `eks-sample-app`.

```
kubectl get all -n eks-sample-app
```

Un ejemplo de salida sería el siguiente.

```
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
```

En la salida, verá el servicio y la implementación que se especificaron en los manifiestos de muestra implementados en pasos anteriores. También verá tres pods. Esto se debe a que especificó 3 `replicas` en el manifiesto de ejemplo. Para obtener más información sobre los pods, consulte Pods en la documentación de Kubernetes. Kubernetes crea automáticamente el recurso `replicaset`, aunque no se especifica en los manifiestos de ejemplo. Para obtener más información acerca de `ReplicaSets`, consulte ReplicaSet en la documentación de Kubernetes.

> **Note**
>
> Kubernetes mantiene el número de réplicas que se ha especificado en el manifiesto. Si se trata de una implementación de producción y desea que Kubernetes escale horizontalmente el número de réplicas o escale verticalmente los recursos de computación de los pods, utilice el Escalado de las implementaciones de pods con el Escalador automático horizontal de pods y Ajuste los recursos del pod con el Escalador automático vertical de pods para hacerlo.

2. Consulte los detalles del servicio implementado.

```
kubectl -n eks-sample-app describe service eks-sample-linux-service
```

Un ejemplo de salida sería el siguiente.

```
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>
```

En la salida anterior, el valor de `IP:` es una dirección IP única a la que se puede acceder desde cualquier nodo o pod del clúster, pero no se puede acceder a ella desde fuera del clúster. Los valores de `Endpoints` son direcciones IP asignadas desde la VPC a los pods que forman parte del servicio.

3. Vea los detalles de uno de los pods que aparecen en la salida de cuando vio el espacio de nombres en un paso anterior. Reemplace *`776d8f8fd8-78w66`* por el valor que devuelve uno de sus pods.

```
kubectl -n eks-sample-app describe pod eks-sample-linux-deployment-65b7669776-m6qxz
```

Ejemplo abreviado de salida

```
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
[...]
```

En la salida anterior, el valor de `IP:` es una IP única que se asigna al pod desde el bloque de CIDR asignado a la subred en la que se encuentra el nodo. Si prefiere asignar a los pods direcciones IP de bloques de CIDR distintos, puede cambiar el comportamiento predeterminado. Para obtener más información, consulte the section called “Redes personalizadas”. También puede ver que el programador de Kubernetes programó el pod en el `Node` con la dirección IP *`192.168.45.132`*.

> **Tip**
>
> En lugar de utilizar la línea de comandos, puede ver muchos detalles sobre los pods, los servicios, las implementaciones y otros recursos de Kubernetes en la Consola de administración de AWS. Para obtener más información, consulte the section called "Acceso a recursos del clúster".

Ejecute un intérprete de comandos en un pod

1. Ejecute un shell en el pod que describió en el paso anterior y reemplace *65b7669776-m6qxz* con el ID de uno de sus pods.

```
kubectl exec -it eks-sample-linux-deployment-65b7669776-m6qxz -n eks-sample-app -- /
bin/bash
```

2. Desde el intérprete de comandos del pod, vea la salida del servidor web que se instaló con la implementación en un paso anterior. Solo tiene que especificar el nombre del servicio. CoreDNS lo resuelve en la dirección IP del servicio, que se implementa con un clúster de Amazon EKS de forma predeterminada.

```
curl eks-sample-linux-service
```

   Un ejemplo de salida sería el siguiente.

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
[...]
```

3. Desde el shell del pod, vea el servidor de DNS del pod.

```
cat /etc/resolv.conf
```

   Un ejemplo de salida sería el siguiente.

```
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
```

En la salida anterior, `10.100.0.10` se asigna automáticamente como `nameserver` para todos los pods implementados en el clúster.

4. Puede desconectarse del pod al escribir `exit`.
5. Una vez que haya terminado de utilizar la aplicación de muestra, puede eliminar el espacio de nombres, el servicio y la implementación de muestra con el siguiente comando.

```
kubectl delete namespace eks-sample-app
```

Siguientes pasos

Después de implementar la aplicación de ejemplo, es posible que desee intentar alguno de los siguientes ejercicios:

- Dirección de la aplicación y el tráfico de HTTP con los equilibradores de carga de aplicación
- Dirija el tráfico de TCP y UDP con equilibradores de carga de red

Implementación de una aplicación de muestra en Windows

En este tema, implementará una aplicación de muestra a su clúster en nodos de Windows.

Requisitos previos

- Un clúster de Kubernetes existente que tenga como mínimo un nodo. Si no dispone de un clúster de Amazon EKS existente, puede implementar uno mediante una de las guías de Introducción. Debe tener la compatibilidad con Windows habilitada para el clúster y al menos en un nodo de Windows de Amazon EC2.
- `Kubectl` instalado en su equipo. Para obtener más información, consulte the section called “Configure kubectl y eksctl”.
- `Kubectl` configurado para comunicarse con el clúster. Para obtener más información, consulte the section called “Acceso al clúster con kubectl”.
- Si planea implementar su carga de trabajo de muestra en Fargate, debe tener una perfil de Fargate que incluya el mismo espacio de nombres creado en este tutorial, que es `eks-sample-app`, a

menos que cambie el nombre. Si utilizó una de las guías de Introducción para crear el clúster, tendrá que crear un nuevo perfil o agregar el espacio de nombres a su perfil existente, porque el perfil creado en las guías de introducción no especifica el espacio de nombres utilizado en este tutorial. La VPC debe tener también al menos una subred privada.

Aunque muchas variables se pueden cambiar en los siguientes pasos, recomendamos cambiar solo los valores de las variables si se especifican. Una vez que conozca mejor los pods, las implementaciones y los servicios de Kubernetes, puede experimentar mediante el cambio de otros valores.

Creación de un espacio de nombres de

Un espacio de nombres permite agrupar recursos en Kubernetes. Para obtener más información, consulte Espacios de nombres en la documentación de Kubernetes. Si planea implementar su aplicación de muestra para Simplificación de la administración de computación con AWS Fargate, asegúrese de que el valor de `namespace` en su Definición de qué pods usarán AWS Fargate cuando se lancen sea `eks-sample-app`.

```
kubectl create namespace eks-sample-app
```

Cree una implementación de Kubernetes

Esta implementación de muestra extrae una imagen de contenedor de un repositorio público e implementa tres réplicas (pods individuales) de ella en su clúster. Para obtener más información, consulte Implementaciones en la documentación de Kubernetes.

1. Guarde los siguientes contenidos en un archivo llamado `eks-sample-deployment.yaml`. Los contenedores de la aplicación de muestra no utilizan almacenamiento en red, pero es posible que tenga aplicaciones que lo necesiten. Para obtener más información, consulte Almacenamiento de datos de aplicaciones.
   - `kubernetes.io/os: windows` `nodeSelector` significa que si tuviera nodos Windows y Linux (por ejemplo) en el clúster, la imagen solo se implementaría en nodos Windows. Para obtener más información, consulte Etiquetas, anotaciones y taints conocidas en la documentación de Kubernetes.

   ```
   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. Aplique el manifiesto de implementación al clúster.

```
kubectl apply -f eks-sample-deployment.yaml
```

Crear un servicio

Un servicio le permite acceder a todas las réplicas a través de una única dirección IP o nombre. Para obtener más información, consulte Servicio en la documentación de Kubernetes. Aunque no se ha implementado en la aplicación de muestra, si tiene aplicaciones que necesitan interactuar con otros servicios de AWS, recomendamos que cree cuentas de servicio de Kubernetes para sus pods y las asocie a cuentas de AWS IAM. Al especificar cuentas de servicio, los pods tienen solo los permisos mínimos que especifica para interactuar con otros servicios. Para obtener más información, consulte the section called “Credenciales con IRSA”.

1. Guarde los siguientes contenidos en un archivo llamado `eks-sample-service.yaml`. Kubernetes asigna al servicio su propia dirección IP a la que se puede acceder solo desde dentro del clúster. Para acceder al servicio desde fuera del clúster, implemente el Controlador del equilibrador de carga de AWS para que equilibre la carga de la aplicación o del tráfico de red del servicio.

```
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. Aplique el manifiesto de servicio al clúster.

```
kubectl apply -f eks-sample-service.yaml
```

Revise los recursos creados

1. Consulte todos los recursos que existen en el espacio de nombres `eks-sample-app`.

```
kubectl get all -n eks-sample-app
```

Un ejemplo de salida sería el siguiente.

```
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
```

En la salida, verá el servicio y la implementación que se especificaron en los manifiestos de muestra implementados en pasos anteriores. También verá tres pods. Esto se debe a que especificó 3 `replicas` en el manifiesto de ejemplo. Para obtener más información sobre los pods, consulte Pods en la documentación de Kubernetes. Kubernetes crea automáticamente el recurso `replicaset`, aunque no se especifica en los manifiestos de ejemplo. Para obtener más información acerca de `ReplicaSets`, consulte ReplicaSet en la documentación de Kubernetes.

> **Note**
>
> Kubernetes mantiene el número de réplicas que se ha especificado en el manifiesto. Si se trata de una implementación de producción y desea que Kubernetes escale horizontalmente el número de réplicas o escale verticalmente los recursos de computación de los pods, utilice el Escalado de las implementaciones de pods con el Escalador

automático horizontal de pods y Ajuste los recursos del pod con el Escalador automático vertical de pods para hacerlo.

2. Consulte los detalles del servicio implementado.

```
kubectl -n eks-sample-app describe service eks-sample-windows-service
```

Un ejemplo de salida sería el siguiente.

```
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>
```

En la salida anterior, el valor de `IP:` es una dirección IP única a la que se puede acceder desde cualquier nodo o pod del clúster, pero no se puede acceder a ella desde fuera del clúster. Los valores de `Endpoints` son direcciones IP asignadas desde la VPC a los pods que forman parte del servicio.

3. Vea los detalles de uno de los pods que aparecen en la salida de cuando vio el espacio de nombres en un paso anterior. Reemplace *776d8f8fd8-78w66* por el valor que devuelve uno de sus pods.

```
kubectl -n eks-sample-app describe pod eks-sample-windows-deployment-65b7669776-m6qxz
```

Ejemplo abreviado de salida

```
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
[...]
```

En la salida anterior, el valor de `IP:` es una IP única que se asigna al pod desde el bloque de CIDR asignado a la subred en la que se encuentra el nodo. Si prefiere asignar a los pods direcciones IP de bloques de CIDR distintos, puede cambiar el comportamiento predeterminado. Para obtener más información, consulte the section called “Redes personalizadas”. También puede ver que el programador de Kubernetes programó el pod en el `Node` con la dirección IP *`192.168.45.132`*.

> **Tip**
>
> En lugar de utilizar la línea de comandos, puede ver muchos detalles sobre los pods, los servicios, las implementaciones y otros recursos de Kubernetes en la Consola de administración de AWS. Para obtener más información, consulte the section called “Acceso a recursos del clúster”.

Ejecute un intérprete de comandos en un pod

1. Ejecute un shell en el pod que describió en el paso anterior y reemplace *65b7669776-m6qxz* con el ID de uno de sus pods.

```
kubectl exec -it eks-sample-windows-deployment-65b7669776-m6qxz -n eks-sample-app --
 powershell.exe
```

2. Desde el intérprete de comandos del pod, vea la salida del servidor web que se instaló con la implementación en un paso anterior. Solo tiene que especificar el nombre del servicio. CoreDNS lo resuelve en la dirección IP del servicio, que se implementa con un clúster de Amazon EKS de forma predeterminada.

```
Invoke-WebRequest -uri eks-sample-windows-service/default.html -UseBasicParsing
```

   Un ejemplo de salida sería el siguiente.

```
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. Desde el shell del pod, vea el servidor de DNS del pod.

```
Get-NetIPConfiguration
```

   Salida abreviada

```
InterfaceAlias       : vEthernet
[...]
IPv4Address          : 192.168.63.14
[...]
DNSServer            : 10.100.0.10
```

   En la salida anterior, `10.100.0.10` se asigna automáticamente como servidor de DNS para todos los pods implementados en el clúster.

4. Puede desconectarse del pod al escribir `exit`.

5. Una vez que haya terminado de utilizar la aplicación de muestra, puede eliminar el espacio de nombres, el servicio y la implementación de muestra con el siguiente comando.

```
kubectl delete namespace eks-sample-app
```

Siguientes pasos

Después de implementar la aplicación de ejemplo, es posible que desee intentar alguno de los siguientes ejercicios:

- Dirección de la aplicación y el tráfico de HTTP con los equilibradores de carga de aplicación
- Dirija el tráfico de TCP y UDP con equilibradores de carga de red

Ajuste de los recursos del pod con el escalador automático vertical de pods

El Escalador automático vertical de pods de Kubernetes ajusta de forma automática las reservas de CPU y memoria de sus pods para ayudar a "ajustar el tamaño" de las aplicaciones. Este ajuste puede mejorar el uso de los recursos del clúster y liberar CPU y memoria para otros pods. Este tema lo ayuda a implementar el escalador automático vertical de pods en el clúster y a comprobar que funciona.

- Tiene un clúster de Amazon EKS existente. Si no lo tiene, consulte Introducción.
- Tiene instalado el servidor de métricas de Kubernetes. Para obtener más información, consulte the section called "Servidor de métricas".
- Utiliza un cliente `kubectl` que está configurado para comunicarse con el clúster de Amazon EKS.
- OpenSSL `1.1.1` o posterior instalado en su dispositivo.

Implementar el escalador automático vertical de pods

En esta sección, implementará el escalador automático vertical de pods en el clúster.

1. Abra una ventana de terminal y vaya al directorio en el que desee descargar el código fuente del escalador automático vertical de pods.
2. Clone el repositorio kubernetes/autoscaler de GitHub.

```
git clone https://github.com/kubernetes/autoscaler.git
```

3. Cambie al directorio de `vertical-pod-autoscaler`.

```
cd autoscaler/vertical-pod-autoscaler/
```

4. (Opcional) Si ya ha implementado otra versión del escalador automático vertical de pods, elimínela con el siguiente comando.

```
./hack/vpa-down.sh
```

5. Si los nodos no tienen acceso a Internet al registro de contenedores `registry.k8s.io`, entonces debe extraer las siguientes imágenes y enviarlas a su propio repositorio privado. Para obtener más información sobre cómo extraer y enviar las imágenes en su propio repositorio privado, consulte the section called “Copiar una imagen en un repositorio”.

```
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
```

   Si va a enviar las imágenes a un repositorio privado de Amazon ECR, sustituya `registry.k8s.io` en los manifiestos por su registro. Reemplace *`111122223333`* por el ID de su cuenta. Reemplace *`region-code`* por la región de AWS en la que se encuentra el clúster. El siguiente comando supone que el nombre del repositorio privado es el mismo que el repositorio de origen en el manifiesto. Si le ha dado un nombre diferente a su repositorio, tendrá que cambiarlo también.

```
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. Implemente el escalador automático vertical de pods en el clúster con el siguiente comando.

```
./hack/vpa-up.sh
```

7. Compruebe que se hayan creado correctamente los pods del Escalador automático vertical de pods.

```
kubectl get pods -n kube-system
```

Un ejemplo de salida sería el siguiente.

```
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
```

Comprobar la instalación del escalador automático vertical de pods

En esta sección, implementará una aplicación de ejemplo para verificar que el escalador automático vertical de pods funciona.

1. Implemente el ejemplo del escalador automático vertical de pods de `hamster.yaml` con el siguiente comando.

```
kubectl apply -f examples/hamster.yaml
```

2. Obtenga los pods de la aplicación de ejemplo de `hamster`.

```
kubectl get pods -l app=hamster
```

Un ejemplo de salida sería el siguiente.

```
hamster-c7d89d6db-rglf5   1/1     Running   0          48s
hamster-c7d89d6db-znvz5   1/1     Running   0          48s
```

3. Describa uno de los pods para ver la reserva de `cpu` y `memory`. Reemplace *`c7d89d6db-rglf5`* por uno de los ID obtenidos en la salida del paso anterior.

```
kubectl describe pod hamster-c7d89d6db-rglf5
```

Un ejemplo de salida sería el siguiente.

```
[...]
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
[...]
```

Puede ver que el pod original reserva 100 milicpu de CPU y 50 mebibytes de memoria. En esta aplicación de ejemplo, 100 milicpu es menos de lo que necesita el pod para ejecutarse, por lo que está limitada a la CPU. También reserva mucha menos memoria de la que necesita. La implementación del Escalador automático vertical de pods de `vpa-recommender` analiza los pods de hamster para ver si los requisitos de CPU y memoria son adecuados. Si es necesario hacer algún ajuste, `vpa-updater` vuelve a lanzar los pods con valores actualizados.

4. Espere a que `vpa-updater` lance un nuevo pod de hamster. Esto debería tardar uno o dos minutos. Puede supervisar los pods con el siguiente comando.

> **Note**
>
> Si no tiene la seguridad de que se haya lanzado un nuevo pod, compare los nombres de los pods con la lista anterior. Cuando se lance el nuevo pod, verá un nuevo nombre de pod.

```
kubectl get --watch Pods -l app=hamster
```

5. Cuando se inicie un nuevo pod de hamster, descríbalo y vea las reservas de CPU y memoria actualizadas.

```
kubectl describe pod hamster-c7d89d6db-jxgfv
```

Un ejemplo de salida sería el siguiente.

```
[...]
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
[...]
```

En la salida anterior puede ver que la reserva de `cpu` ha aumentado a 587 milicpu, que es más de cinco veces el valor original. La `memory` ha aumentado a 262 144 kilobytes, lo que equivale a 250 mebibytes, o a cinco veces el valor original. Este pod no tenía recursos suficientes, y el Escalador automático vertical de pods corrigió la estimación con un valor mucho más adecuado.

6. Describa el recurso de `hamster-vpa` para ver la nueva recomendación.

```
kubectl describe vpa/hamster-vpa
```

Un ejemplo de salida sería el siguiente.

```
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. Cuando termine de experimentar con la aplicación de ejemplo, elimínela con el siguiente comando.

```
kubectl delete -f examples/hamster.yaml
```

Escalado de las implementaciones de pods con el escalador automático de pods horizontales

El Escalador automático horizontal de pods de Kubernetes escala automáticamente el número de pods en una implementación, un controlador de reproducción o un conjunto de réplicas en función del uso de la CPU de ese recurso. Esto puede ayudar a las aplicaciones a escalar horizontalmente para satisfacer el aumento de la demanda o a reducir horizontalmente cuando no se necesitan recursos y, de esa forma, liberar los nodos para otras aplicaciones. Cuando se establece un porcentaje de utilización de CPU objetivo, el escalador automático de pods horizontales escala su aplicación de forma horizontal o la reduce horizontalmente para intentar alcanzar ese objetivo.

El escalador automático de pods horizontales es un recurso de la API estándar en Kubernetes que simplemente requiere que una fuente de métricas (como el servidor de métricas de Kubernetes) esté instalada en el clúster de Amazon EKS para funcionar. No es necesario implementar ni instalar el escalador automático de pods horizontales en el clúster para comenzar a escalar las aplicaciones. Para obtener más información, consulte Horizontal Pod Autoscaler (Escalador automático de pods horizontales) en la documentación de Kubernetes.

Utilice este tema para preparar el escalador automático de pods horizontales para el clúster de Amazon EKS y para verificar que funciona con una aplicación de muestra.

> **Note**
>
> Este tema se explica en Horizontal Pod autoscaler walkthrough en la documentación de Kubernetes.

- Tiene un clúster de Amazon EKS existente. Si no lo tiene, consulte Introducción.
- Tiene instalado el servidor de métricas de Kubernetes. Para obtener más información, consulte the section called “Servidor de métricas”.
- Utiliza un cliente `kubectl` que está configurado para comunicarse con el clúster de Amazon EKS.

Ejecutar una aplicación de prueba del escalador automático de pods horizontales

En esta sección, implementará una aplicación de muestra para verificar que el escalador automático de pods horizontales funciona.

> **Note**
>
> Este ejemplo se explica en Horizontal Pod autoscaler walkthrough en la documentación de Kubernetes.

1. Implemente una aplicación de servidor web Apache sencilla con el siguiente comando.

   ```
   kubectl apply -f https://k8s.io/examples/application/php-apache.yaml
   ```

   Este pod de servidor web Apache tiene un límite de CPU de 500 milicpu y sirve en el puerto 80.
2. Cree un recurso del escalador automático de pods horizontales para la implementación de `php-apache`.

   ```
   kubectl autoscale deployment php-apache --cpu-percent=50 --min=1 --max=10
   ```

   Este comando crea un escalador automático que tiene como objetivo el 50 % de uso de la CPU para la implementación, con un mínimo de un pod y un máximo de diez pods. Cuando la carga media de CPU es inferior al 50 %, el escalador automático intenta reducir el número de pods

en la implementación a un mínimo de uno. Cuando la carga es superior al 50 %, el escalador automático intenta aumentar el número de pods en la implementación, hasta un máximo de diez. Para obtener más información, consulte How does a HorizontalPodAutoscaler work? en la documentación de Kubernetes.

3. Describa el escalador automático con el siguiente comando para ver los detalles.

```
kubectl get hpa
```

Un ejemplo de salida sería el siguiente.

```
NAME         REFERENCE                TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
php-apache   Deployment/php-apache   0%/50%    1         10        1          51s
```

Como puede ver, la carga actual de la CPU es `0%`, porque todavía no hay carga en el servidor. El recuento de pods ya se encuentra en su límite más bajo (uno), por lo que no se puede reducir horizontalmente.

4. Cree una carga para el servidor web mediante la ejecución de un contenedor.

```
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. Para ver que la implementación escale horizontalmente, ejecute de manera periódica el siguiente comando en un terminal independiente desde el terminal en el que ejecutó el paso anterior.

```
kubectl get hpa php-apache
```

Un ejemplo de salida sería el siguiente.

```
NAME         REFERENCE                TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
php-apache   Deployment/php-apache   250%/50%   1         10        5          4m44s
```

El recuento de réplicas puede tardar más de un minuto en aumentar. Mientras el porcentaje actual de la CPU sea superior al porcentaje objetivo, el recuerdo de réplicas aumenta hasta 10. En este caso, es `250%`, para que el número de `REPLICAS` siga en aumento.

> **Note**
>
> Pueden pasar unos minutos antes de que vea que el recuento de réplicas alcanza su máximo. Si solo se necesitan 6 réplicas, por ejemplo, para que la carga de la CPU permanezca al o por debajo del 50 %, la carga no superará las 6 réplicas.

6. Detenga la carga. En la ventana del terminal en la que genera la carga, mantenga presionadas las teclas `Ctrl+C` para detener la carga. Puede ver cómo las réplicas vuelven a escalar a 1 al ejecutar de nuevo el siguiente comando en el terminal en el que ve la reducción horizontal.

```
kubectl get hpa
```

   Un ejemplo de salida sería el siguiente.

```
NAME         REFERENCE                TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
php-apache   Deployment/php-apache    0%/50%    1         10        1          25m
```

> **Note**
>
> El periodo predeterminado para reducir la escala es de cinco minutos, por lo que pasará algún tiempo antes de que vea que el recuento de réplicas es de 1 de nuevo, incluso cuando el porcentaje actual de la CPU es 0 %. El periodo de tiempo es modificable. Para obtener más información, consulte Horizontal Pod Autoscaler (Escalador automático de pods horizontales) en la documentación de Kubernetes.

7. Cuando haya terminado de experimentar con la aplicación de muestra, elimine los recursos `php-apache`.

```
kubectl delete deployment.apps/php-apache service/php-apache
 horizontalpodautoscaler.autoscaling/php-apache
```

Dirija el tráfico de TCP y UDP con equilibradores de carga de red

> **Note**
>
> Nuevo: el modo automático de Amazon EKS automatiza las tareas rutinarias para el equilibrio de carga. Para obtener más información, consulte:
>
> - the section called “Implementación del equilibrador de carga”
> - the section called “Crear un servicio”

Se equilibra la carga del tráfico de red en L4 del modelo OSI. Para equilibrar la carga del tráfico de aplicaciones en L7, implemente una `ingress` de Kubernetes, que aprovisiona un equilibrador de carga de aplicación de AWS. Para obtener más información, consulte the section called “Equilibrio de carga de aplicaciones”. Para obtener más información sobre las diferencias entre los dos tipos de equilibrio de carga, consulte Características de Elastic Load Balancing en el sitio web de AWS.

Cuando crea un `Service` de Kubernetes de tipo `LoadBalancer`, el Controlador del equilibrador de carga del proveedor de nube de AWS crea AWS Classic Load Balancers de forma predeterminada, pero también puede crear AWS Network Load Balancers. Este controlador solo recibirá correcciones de errores críticos en el futuro. Para obtener más información acerca del uso del equilibrador de carga del proveedor de nube de AWS, consulte Controlador del equilibrador de carga del proveedor de nube de AWS en la documentación de Kubernetes. Su uso no se aborda en este tema.

Le recomendamos usar la versión `2.7.2` o una posterior del Equilibrador de carga de AWS en lugar del controlador del equilibrador de carga del proveedor de la nube de AWS. El Controlador del equilibrador de carga de AWS crea equilibradores de carga de red de AWS, pero no equilibradores de carga clásicos de AWS. El resto de este tema trata sobre el uso del Controlador del equilibrador de carga de AWS.

Un equilibrador de carga de red de AWS puede equilibrar la carga del tráfico de red a pods implementados en destinos de instancia e IP de Amazon EC2, en destinos de IP de AWS Fargate o en los Nodos híbridos de Amazon EKS como destinos de IP. Para obtener más información, consulte Controlador del equilibrador de carga de AWS en GitHub.

Requisitos previos

Para poder equilibrar la carga del tráfico de red con el Controlador del equilibrador de carga de AWS, debe cumplir los siguientes requisitos.

- Tener un clúster existente. Si no tiene un clúster existente, consulte Introducción. Si necesita actualizar la versión de un clúster existente, consulte the section called “Actualización de una versión de Kubernetes”.
- Tenga el Controlador del equilibrador de carga de AWS implementado en el clúster. Para obtener más información, consulte the section called “ AWS Controlador del equilibrador de carga de ”. Recomendamos la versión `2.7.2` o posterior.
- Tener al menos una subred. Si varias subredes etiquetadas se encuentran en una zona de disponibilidad, el controlador elige la primera subred cuyo ID de subred va primero lexicográficamente. La subred debe tener al menos ocho direcciones IP disponibles.
- Si utiliza la versión `2.1.1` del Controlador del equilibrador de carga de AWS o versiones anteriores, las subredes deben etiquetarse de la siguiente manera. Si utiliza la versión `2.1.2` o posterior, esta etiqueta es opcional. Es posible que desee etiquetar una subred si tiene varios clústeres que se ejecutan en la misma VPC o varios servicios de AWS que comparten subredes en una VPC y desea tener más control sobre dónde se aprovisionan los equilibradores de carga para cada clúster. Si especifica de manera explícita los ID de subred como una anotación en un objeto de servicio, Kubernetes y el Controlador del equilibrador de carga de AWS utilizarán esas subredes directamente para crear el equilibrador de carga. El etiquetado de subred no es necesario si elige utilizar este método para aprovisionar los equilibradores de carga y puede omitir los siguientes requisitos de etiquetado de subred privada y pública. Sustituya *`my-cluster`* por el nombre del clúster.
  - Clave – `kubernetes.io/cluster/<my-cluster>`
  - Valor: `shared` o `owned`
- Las subredes públicas y privadas deben cumplir los siguientes requisitos, a menos que especifique de manera explícita los ID de subred como una anotación en un objeto de servicio o entrada. Si aprovisiona equilibradores de carga mediante la especificación explícita de los ID de subred como una anotación en un objeto de servicio o entrada, Kubernetes y el Controlador del equilibrador de carga de AWS utilizarán esas subredes directamente para crear el equilibrador de carga, y las siguientes etiquetas no serán necesarias.
  - Subredes privadas: deben etiquetarse con el siguiente formato. De esta manera Kubernetes y el Controlador del equilibrador de carga de AWS saben que las subredes se pueden utilizar para balanceadores de carga internos. Si utiliza `eksctl` o una plantilla de AWS de AWS Cloud Formation de Amazon EKS para crear la VPC después del 26 de marzo de 2020, las subredes se etiquetan correctamente cuando se crean. Para obtener más información sobre las plantillas de VPC de AWS CloudFormation de AWS Amazon EKS, consulte the section called “Creación de una VPC”.

- Clave – `kubernetes.io/role/internal-elb`
- Valor – 1

- Subredes públicas: deben etiquetarse con el siguiente formato. Es así para que Kubernetes sepa que debe utilizar solo esas subredes para los balanceadores de carga externos, en lugar de elegir una subred pública en cada zona de disponibilidad (en orden lexicográfico por ID de subred). Si utiliza `eksctl` o una plantilla de AWS Cloud Formation de Amazon EKS para crear la VPC después del 26 de marzo de 2020, las subredes se etiquetan correctamente cuando se crean. Para obtener más información sobre las plantillas de VPC de AWS CloudFormation de Amazon EKS, consulte the section called “Creación de una VPC”.
  - Clave – `kubernetes.io/role/elb`
  - Valor – 1

Si las etiquetas del rol de subred no se agregan de manera explícita, el controlador de servicio de Kubernetes examinará la tabla de enrutamiento de las subredes de la VPC del clúster para determinar si la subred es privada o pública. Recomendamos que no dependa de este comportamiento y que, en su lugar, agregue de manera explícita las etiquetas de rol públicas o privadas. El Controlador del equilibrador de carga de AWS no examina las tablas de enrutamiento y requiere que las etiquetas privadas y públicas estén presentes para la detección automática correcta.

Consideraciones

- La configuración del equilibradores de carga se controla mediante anotaciones que se agregan al manifiesto del servicio. Las anotaciones de servicio son diferentes cuando se utiliza el Controlador del equilibrador de carga de AWS en comparación a cuando se utiliza el controlador del equilibrador de carga del proveedor de nube de AWS. Asegúrese de revisar las anotaciones para el Controlador del equilibrador de carga de AWS antes de implementar los servicios.
- Cuando se utiliza el complemento CNI de Amazon VPC para Kubernetes, el Controlador del equilibrador de carga de AWS puede equilibrar la carga hacia destinos IP o de instancia de Amazon EC2 y destinos IP de Fargate. Cuando se utilizan complementos de CNI compatibles con Alternate, el controlador solo puede equilibrar la carga hacia destinos de instancias, a menos que se equilibre la carga hacia los Nodos híbridos de Amazon EKS. En el caso de los nodos híbridos, el controlador puede equilibrar la carga de los destinos IP. Para obtener más información acerca de los tipos de destinos de Network Load Balancer, consulte Tipo de destino en la Guía del usuario para Network Load Balancer.

- Si desea agregar etiquetas al equilibrador de carga durante su creación o después de ella, agregue la siguiente anotación en la especificación del servicio. Para obtener más información, consulte Etiquetas de recursos de AWS en la documentación del Controlador del equilibrador de carga de AWS.

```
service.beta.kubernetes.io/aws-load-balancer-additional-resource-tags
```

- Para asignar direcciones IP elásticas al Network Load Balancer, agregue la siguiente anotación. Sustituya los valores de ejemplo con los `Allocation IDs` de las direcciones IP elásticas. El número de `Allocation IDs` debe coincidir con el número de subredes utilizadas para el equilibrador de carga. Para obtener más información, consulte la documentación del Controlador del equilibrador de carga de AWS.

```
service.beta.kubernetes.io/aws-load-balancer-eip-allocations: eipalloc-
xxxxxxxxxxxxxxxxx,eipalloc-yyyyyyyyyyyyyyyyy
```

- Amazon EKS agrega una regla de entrada al grupo de seguridad del nodo para el tráfico del cliente y una regla para cada subred del equilibrador de carga de la VPC a fin de realizar comprobaciones de estado para cada equilibrador de carga de red que cree. La implementación de un servicio de tipo `LoadBalancer` puede fallar si Amazon EKS intenta crear reglas que superen la cuota del número máximo de reglas permitidas para un grupo de seguridad. Para obtener más información, consulte Grupos de seguridad en las cuotas de Amazon VPC en la Guía del usuario de Amazon VPC. Considere las siguientes opciones a fin de minimizar las posibilidades de exceder el número máximo de reglas para un grupo de seguridad:
  - Solicite un aumento de las reglas por cuota de grupo de seguridad. Para obtener más información, consulte Solicitar un aumento de cuota en la Guía del usuario de Service Quotas.
  - Utilice destinos de IP, en lugar de destinos de instancia. Con los destinos de IP, puede compartir reglas para los mismos puertos de destino. Puede especificar manualmente las subredes del balanceador de carga con una anotación. Para obtener más información, consulte Anotaciones en GitHub.
  - Utilice una entrada en lugar de un servicio de tipo `LoadBalancer` para enviar tráfico al servicio. El AWS Application Load Balancer requiere menos reglas que los Network Load Balancers. Puede compartir un ALB entre varias entradas. Para obtener más información, consulte the section called “Equilibrio de carga de aplicaciones”. No puede compartir un Equilibrador de carga de red entre varios servicios.
  - Implemente sus clústeres en varias cuentas.

- Si los pods se ejecutan en Windows en un clúster de Amazon EKS, un único servicio con un equilibrador de carga puede admitir hasta 1024 pods de backend. Cada pod tiene su propia dirección IP única.
- Recomendamos crear solo Network Load Balancers nuevos con el Controlador del equilibrador de carga de AWS. Intentar sustituir los Network Load Balancers existentes creados con el Controlador del equilibrador de carga del proveedor de nube de AWS puede dar lugar a varios Network Load Balancers que podrían causar tiempo de inactividad en la aplicación.

Crear un equilibrador de carga de red

Puede crear un equilibrador de carga de red con IP o destinos de instancia.

Cree un equilibrador de carga de red: destinos de IP

- Puede utilizar destinos de IP con pods implementados en nodos de Amazon EC2, Fargate o Nodos híbridos de Amazon EKS. El servicio de Kubernetes debe crearse como tipo `LoadBalancer`. Para obtener más información, consulte Tipo de balanceador de carga en la documentación de Kubernetes.

  Para crear un equilibrador de carga que utilice destinos IP, agregue las siguientes anotaciones a un manifiesto de servicio e implemente el servicio. El valor `external` para `aws-load-balancer-type` es lo que lleva al Controlador del equilibrador de carga de AWS, en lugar del controlador del equilibrador de carga del proveedor de nube de AWS, a crear el Network Load Balancer. Puede ver un manifiesto de servicio de ejemplo con los comentarios.

```
service.beta.kubernetes.io/aws-load-balancer-type: "external"
service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: "ip"
```

> **Note**
>
> Si equilibra la carga en pods `IPv6`, agregue la siguiente anotación. Solo puede equilibrar la carga a través de `IPv6` a destinos IP, no a destinos de instancias. Sin esta anotación, la tarea de equilibrar la carga se realiza a través de `IPv4`.

```
service.beta.kubernetes.io/aws-load-balancer-ip-address-type: dualstack
```

Los Network Load Balancers se crean con el `internal` `aws-load-balancer-scheme` de forma predeterminada. Puede lanzar equilibradores de carga de red en cualquier subred de la VPC del clúster, incluidas las subredes sin especificar cuando creó el clúster.

Kubernetes examina la tabla de enrutamiento de las subredes con el fin de identificar si son públicas o privadas. Las subredes públicas tienen una ruta directa a Internet mediante una puerta de enlace de Internet, pero no así las subredes privadas.

Si desea crear un Network Load Balancer en una subred pública para equilibrar la carga en nodos de Amazon EC2 (Fargate solo puede ser privado), especifique `internet-facing` con la siguiente anotación:

```
service.beta.kubernetes.io/aws-load-balancer-scheme: "internet-facing"
```

> **Note**
>
> La anotación `service.beta.kubernetes.io/aws-load-balancer-type: "nlb-ip"` todavía se admite para la compatibilidad con versiones anteriores. Sin embargo, le recomendamos que utilice las anotaciones anteriores para nuevos equilibradores de carga en lugar de `service.beta.kubernetes.io/aws-load-balancer-type: "nlb-ip"`.

> **Important**
>
> No edite los comentarios después de crear el servicio. Si necesita modificarlo, elimine el objeto de servicio y vuelva a crearlo con el valor deseado para este comentario.

Cree un equilibrador de carga de red: destinos de instancia

- El Controlador del equilibrador de carga del proveedor de nube de AWS crea Network Load Balancers solo con destinos de instancia. La versión `2.2.0` y posteriores del Controlador del equilibrador de carga de AWS también crean equilibradores de carga de red con destinos de instancia. Recomendamos utilizarlo, en lugar de utilizar el Controlador del equilibrador de carga del proveedor de nube de AWS, para crear nuevos Network Load Balancers. Puede utilizar destinos de instancia del equilibrador de carga de red con pods implementados en nodos de Amazon EC2,

pero no en Fargate. Para equilibrar la carga del tráfico de red a través de los pods implementados en Fargate, debe utilizar destinos de IP.

Para implementar un Network Load Balancer en una subred privada, la especificación del servicio debe tener las siguientes anotaciones. Puede ver un manifiesto de servicio de ejemplo con los comentarios. El valor `external` para `aws-load-balancer-type` es lo que lleva al Controlador del equilibrador de carga de AWS, en lugar del controlador del equilibrador de carga del proveedor de nube de AWS, a crear el Network Load Balancer.

```
service.beta.kubernetes.io/aws-load-balancer-type: "external"
service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: "instance"
```

Los Network Load Balancers se crean con el `internal` `aws-load-balancer-scheme` de forma predeterminada. Para los Network Load Balancers internos, el clúster de Amazon EKS debe estar configurado para utilizar al menos una subred privada en la VPC. Kubernetes examina la tabla de enrutamiento de las subredes con el fin de identificar si son públicas o privadas. Las subredes públicas tienen una ruta directa a Internet mediante una puerta de enlace de Internet, pero no así las subredes privadas.

Si desea crear un Network Load Balancer en una subred pública para equilibrar la carga en nodos de Amazon EC2, especifique `internet-facing` con la siguiente anotación:

```
service.beta.kubernetes.io/aws-load-balancer-scheme: "internet-facing"
```

> **Important**
>
> No edite los comentarios después de crear el servicio. Si necesita modificarlo, elimine el objeto de servicio y vuelva a crearlo con el valor deseado para este comentario.

(Opcional) Implementación de una aplicación de muestra

- Al menos una subred privada o pública en la VPC del clúster.
- Tenga el Controlador del equilibrador de carga de AWS implementado en el clúster. Para obtener más información, consulte the section called “ AWS Controlador del equilibrador de carga de ”. Recomendamos la versión `2.7.2` o posterior.

1. Si va a implementar en Fargate, asegúrese de tener una subred privada disponible en la VPC y cree un perfil de Fargate. Si no implementa en Fargate, omita este paso. Puede crear el perfil con la ejecución del siguiente comando o en la Consola de administración de AWS con los mismos valores para `name` y `namespace` que los del comando. Sustituya los valores de ejemplo por sus propios valores.

```
eksctl create fargateprofile \
    --cluster my-cluster \
    --region region-code \
    --name nlb-sample-app \
    --namespace nlb-sample-app
```

2. Implemente una aplicación de muestra.

   a. Cree un espacio de nombres para la aplicación.

```
kubectl create namespace nlb-sample-app
```

   b. Guarde el siguiente contenido en un archivo denominado `sample-deployment.yaml` en el equipo.

```
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. Aplique el manifiesto al clúster.

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

3. Cree un servicio con un equilibrador de carga de red interno que equilibre la carga en destinos IP.

a. Guarde el siguiente contenido en un archivo denominado `sample-service.yaml` en el equipo. Si va a implementar en nodos Fargate, elimine la línea `service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing`.

```
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. Aplique el manifiesto al clúster.

```
kubectl apply -f sample-service.yaml
```

4. Compruebe que el servicio se haya implementado.

```
kubectl get svc nlb-sample-service -n nlb-sample-app
```

Un ejemplo de salida sería el siguiente.

```
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**
>
> Los valores de *10.100.240.137* y *xxxxxxxxxx*-*xxxxxxxxxxxxxxxx* serán diferentes a la salida de ejemplo (serán exclusivos de su equilibrador de carga) y *us-west-2* puede ser diferente según la región de AWS en la que se encuentre el clúster.

5. Abra la Consola de administración de AWS de Amazon EC2. Seleccione Target Groups (Grupos de destino) (en Load Balancing (Equilibrador de carga) en el panel de navegación izquierdo. En la columna Nombre, seleccione el nombre del grupo de destino donde el valor de la columna Equilibrador de carga coincida con el nombre de la columna `EXTERNAL-IP` de la salida en el paso anterior. Por ejemplo, debe seleccionar el grupo de destino denominado `k8s-default-samplese-`*xxxxxxxxxx* si la salida es la misma que la salida anterior. El Target type (Tipo de destino) es `IP` porque así se especificó en el manifiesto del servicio de muestra.
6. Seleccione el Grupo de destino y elija la pestaña Targets (Destinos). En Registered targets (Destinos registrados), debería ver tres direcciones IP de las tres réplicas implementadas en un paso anterior. Espere hasta que el estado de todos los destinos sea healthy (bueno) antes de continuar. Pueden pasar varios minutos hasta que todos los destinos sean `healthy`. Los destinos pueden estar en estado `unhealthy` antes de cambiar al estado `healthy`.
7. Envíe tráfico al servicio mediante el reemplazo de *xxxxxxxxxx-xxxxxxxxxxxxxxxx* y *us-west-2* por los valores devueltos en la salida de un paso anterior para `EXTERNAL-IP`. Si ha realizado la implementación en una subred privada, tendrá que ver la página desde un dispositivo dentro de la VPC, como un host bastión. Para obtener más información, consulte Hosts bastión de Linux en AWS.

   ```
   curl k8s-default-samplese-xxxxxxxxxx-xxxxxxxxxxxxxxxx.elb.region-code.amazonaws.com
   ```

   Un ejemplo de salida sería el siguiente.

   ```
   <!DOCTYPE html>
   <html>
   <head>
   <title>Welcome to nginx!</title>
   [...]
   ```

8. Cuando haya terminado de utilizar la implementación, el servicio y el espacio de nombres de muestra, elimínelos.

```
kubectl delete namespace nlb-sample-app
```

Redirección de tráfico de aplicaciones y HTTP con los equilibradores de carga de aplicaciones

> **Note**
>
> Nuevo: el modo automático de Amazon EKS automatiza las tareas rutinarias para el equilibrio de carga. Para obtener más información, consulte:
>
> - the section called “Implementación del equilibrador de carga”
> - the section called “Creación de una clase de ingreso”

Cuando crea una `ingress` de Kubernetes, se aprovisiona un Application Load Balancer (ALB) de AWS que equilibra la carga del tráfico de aplicaciones. Para obtener más información, consulte ¿Qué es un Application Load Balancer? en la Guía del usuario deApplication Load Balancers y Entrada en la documentación de Kubernetes. Los ALB se pueden utilizar con pods que se implementan en nodos o en AWS Fargate. Puede implementar un ALB en subredes públicas o privadas.

El tráfico de aplicaciones se equilibra en `L7` del modelo OSI. Para equilibrar la carga del tráfico de red en `L4`, implemente un `service` de Kubernetes del tipo `LoadBalancer`. Este tipo aprovisiona un Network Load Balancer de AWS. Para obtener más información, consulte the section called “Equilibrio de carga de red”. Para obtener más información sobre las diferencias entre los dos tipos de equilibrio de carga, consulte Características de Elastic Load Balancing en el sitio web de AWS.

Requisitos previos

Para poder equilibrar la carga del tráfico de aplicaciones a una aplicación, debe cumplir los siguientes requisitos.

- Tener un clúster existente. Si no tiene un clúster existente, consulte Introducción. Si necesita actualizar la versión de un clúster existente, consulte the section called “Actualización de una versión de Kubernetes”.

- Tenga el Controlador del equilibrador de carga de AWS implementado en el clúster. Para obtener más información, consulte the section called " AWS Controlador del equilibrador de carga de ". Recomendamos la versión `2.7.2` o posterior.
- Al menos dos subredes en diferentes zonas de disponibilidad. El Controlador del equilibrador de carga de AWS elige una subred de cada zona de disponibilidad. Cuando varias subredes etiquetadas se encuentran en una zona de disponibilidad, el controlador elige la subred cuyo ID de subred vaya primero lexicográficamente. Cada subred debe tener al menos ocho direcciones IP disponibles.

  Si utiliza varios grupos de seguridad adjuntos al nodo de trabajo, debe etiquetarse exactamente un grupo de seguridad de la siguiente manera. Sustituya *my-cluster* por el nombre del clúster.

  - Clave – `kubernetes.io/cluster/<my-cluster>`
  - Valor: `shared` o `owned`
- Si utiliza la versión `2.1.1` o anterior del Controlador del equilibrador de carga de AWS, las subredes deben etiquetarse con el formato siguiente. Si utiliza la versión `2.1.2` o posterior, el etiquetado es opcional. No obstante, recomendamos que etiquete una subred si se da cualquiera de las siguientes situaciones. Tiene varios clústeres que se ejecutan en la misma VPC o que tienen varios servicios de AWS que comparten subredes en una VPC. O bien, desea tener más control sobre dónde se aprovisionan los equilibradores de carga para cada clúster. Sustituya *my-cluster* por el nombre del clúster.
  - Clave – `kubernetes.io/cluster/<my-cluster>`
  - Valor: `shared` o `owned`
- Las subredes públicas y privadas deben cumplir los siguientes requisitos. Esto es así a menos que especifique de manera explícita los ID de subred como una anotación en un objeto de entrada o servicio. Supongamos que aprovisiona equilibradores de carga mediante la especificación explícita de los ID de subred como una anotación en un objeto de entrada o servicio. En esta situación, Kubernetes y el Controlador del equilibrador de carga de AWS utilizan esas subredes directamente para crear el equilibrador de carga y no se requieren las siguientes etiquetas.
  - Subredes privadas: deben etiquetarse con el siguiente formato. De esta manera Kubernetes y el Controlador del equilibrador de carga de AWS saben que las subredes se pueden utilizar para balanceadores de carga internos. Si utiliza `eksctl` o una plantilla de AWS CloudFormation de Amazon EKS para crear la VPC después del 26 de marzo de 2020, las subredes se etiquetan correctamente cuando se crean. Para obtener más información sobre las plantillas de VPC de AWS CloudFormation de Amazon EKS, consulte the section called "Creación de una VPC".
    - Clave – `kubernetes.io/role/internal-elb`

    - Valor – 1
- Subredes públicas: deben etiquetarse con el siguiente formato. Esto es para que Kubernetes sepa que debe utilizar solo las subredes que se especificaron para los balanceadores de carga externos. De esta forma, Kubernetes no elige una subred pública en cada zona de disponibilidad (lexicográficamente en función de su ID de subred). Si utiliza `eksctl` o una plantilla de AWS CloudFormation de Amazon EKS para crear la VPC después del 26 de marzo de 2020, las subredes se etiquetan correctamente cuando se crean. Para obtener más información sobre las plantillas de VPC de AWS CloudFormation de Amazon EKS, consulte the section called "Creación de una VPC".
    - Clave – `kubernetes.io/role/elb`
    - Valor – 1

  Si las etiquetas de rol de subred no se agregan de manera explícita, el controlador de servicio de Kubernetes examina la tabla de enrutamiento de las subredes de la VPC del clúster. Esto sirve para determinar si la subred es privada o pública. Recomendamos que no dependa de este comportamiento. En su lugar, agregue de manera explícita las etiquetas de rol públicas o privadas. El Controlador del equilibrador de carga de AWS no examina las tablas de enrutamiento. También requiere que las etiquetas privadas y públicas estén presentes para la detección automática correcta.

- El Controlador del equilibrador de carga de AWS crea los ALB y los recursos de apoyo de AWS necesarios cada vez que se crea un recurso de entrada de Kubernetes en el clúster con la anotación `kubernetes.io/ingress.class: alb`. El recurso de entrada configura el ALB para dirigir el tráfico HTTP o HTTPS a diferentes pods dentro del clúster. Para asegurarse de que los objetos de entrada utilizan el Controlador del equilibrador de carga de AWS, agregue la siguiente anotación a su especificación de entrada de Kubernetes. Para obtener más información, consulte Especificación de entrada en GitHub.

```
annotations:
    kubernetes.io/ingress.class: alb
```

> **Note**
>
> Si equilibra la carga en pods `IPv6`, agregue la siguiente anotación a su especificación de entrada. Solo puede equilibrar la carga a través de `IPv6` a destinos IP, no a destinos de instancias. Sin esta anotación, la tarea de equilibrar la carga se realiza a través de `IPv4`.

```
alb.ingress.kubernetes.io/ip-address-type: dualstack
```

- El Controlador del equilibrador de carga de AWS admite los siguientes modos de tráfico:
  - Instancia: registra los nodos dentro del clúster como destinos para el ALB. El tráfico que llega al ALB se dirige a `NodePort` para su servicio y, a continuación, se transfiere por proxy a sus pods. Este es el modo de tráfico predeterminado. También puede especificarlo de manera explícita con la anotación `alb.ingress.kubernetes.io/target-type: instance`.

    > **Note**
    >
    > Su servicio de Kubernetes debe especificar el tipo de `NodePort` o `LoadBalancer` necesario para utilizar este modo de tráfico.

  - IP: registra los pods como destinos para el ALB. El tráfico que llega al ALB se dirige directamente a los pods para su servicio. Debe especificar la anotación `alb.ingress.kubernetes.io/target-type: ip` necesaria para utilizar este modo de tráfico. El tipo de destino de IP es necesario cuando los pods de destino se ejecutan en Fargate o en los Nodos híbridos de Amazon EKS.
- Para etiquetar ALB creados por el controlador, agregue el siguiente comentario al controlador: `alb.ingress.kubernetes.io/tags`. Para ver una lista de todos los comentarios disponibles compatibles con el Controlador del equilibrador de carga de AWS, consulte Comentarios de entrada en GitHub.
- La actualización o degradación de la versión del controlador de ALB puede presentar cambios importantes en las características que dependen de él. Para obtener más información sobre los cambios importantes que se presentan en cada versión, consulte las Notas de la versión del controlador de ALB en GitHub.

Reutilice los ALB con grupos de entrada

Puede compartir un equilibrador de carga de aplicación entre varios recursos de servicio mediante `IngressGroups`.

Para unir una entrada a un grupo, agregue la siguiente anotación a la especificación de un recurso de entrada de Kubernetes.

```
alb.ingress.kubernetes.io/group.name: my-group
```

El nombre del grupo debe:

- Tener 63 caracteres o menos de longitud.
- Constar de letras minúsculas, números, `-` y `.`
- Comenzar y terminar por un número o una letra.

El controlador fusiona de manera automática las reglas de entrada para todas las entradas del mismo grupo de entradas. Las soporta con un solo ALB. La mayoría de las anotaciones definidas en una entrada solo se aplican a las rutas definidas por esa entrada. De forma predeterminada, los recursos de entrada no pertenecen a ningún grupo de entradas.

> **Warning**
>
> Riesgo potencial de seguridad
> Especifique un grupo de entradas para una entrada solo cuando todos los usuarios de Kubernetes que tienen el permiso RBAC para crear o modificar recursos de entrada tienen el mismo límite de confianza. Si agrega la anotación con un nombre de grupo, otros usuarios de Kubernetes podrán crear o modificar sus entradas para que pertenezcan al mismo grupo de entradas. Si lo hace, puede provocar comportamientos indeseables, como sobrescribir reglas existentes con reglas de mayor prioridad.

Puede agregar un número de orden para el recurso de entrada.

```
alb.ingress.kubernetes.io/group.order: '10'
```

El número puede ser entre 1 y 1000. El número más bajo para todas las entradas del mismo grupo de entradas se evalúa primero. Todas las entradas sin esta anotación se evalúan con un valor de cero. Las reglas duplicadas con un número superior pueden sobrescribir reglas con un número inferior. De forma predeterminada, el orden de las reglas entre entradas dentro del mismo grupo de entradas se determina lexicográficamente en función del espacio de nombres y del nombre.

> **Important**
>
> Asegúrese de que cada entrada del mismo grupo de entradas tenga un número de prioridad único. No puede tener números de orden duplicados entre entradas.

(Opcional) Implementación de una aplicación de muestra

- Al menos una subred privada o pública en la VPC del clúster.
- Tenga el Controlador del equilibrador de carga de AWS implementado en el clúster. Para obtener más información, consulte the section called “ AWS Controlador del equilibrador de carga de ”. Recomendamos la versión `2.7.2` o posterior.

Puede ejecutar la aplicación de muestra en un clúster que tenga nodos de Amazon EC2, pods de Fargate o ambos.

1. Si no va a implementar en Fargate, omita este paso. Si va a implementar en Fargate, cree un perfil de Fargate. Puede crear el perfil con la ejecución del siguiente comando o en la Consola de administración de AWS con los mismos valores para `name` y `namespace` que los del comando. Sustituya los valores de ejemplo por sus propios valores.

   ```
   eksctl create fargateprofile \
       --cluster my-cluster \
       --region region-code \
       --name alb-sample-app \
       --namespace game-2048
   ```

2. Implemente el videojuego 2048 como una aplicación de muestra para comprobar que el Controlador del equilibrador de carga de AWS crea un ALB de AWS como resultado del objeto de entrada. Complete los pasos del tipo de subred en la que va a realizar la implementación.

   a. Si va a implementar en pods de un clúster que creó con la familia `IPv6`, vaya al siguiente paso.

   - Público:

     ```
     kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.14.1/docs/examples/2048/2048_full.yaml
     ```

   - Privado:

A. Descargue el manifiesto.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.14.1/docs/examples/2048/2048_full.yaml
```

B. Edite el archivo y busque la línea que indica `alb.ingress.kubernetes.io/scheme: internet-facing`.

C. Cambie *`internet-facing`* por `internal` y guarde el archivo.

D. Aplique el manifiesto al clúster.

```
kubectl apply -f 2048_full.yaml
```

b. Si va a implementar en pods de un clúster que creó con la familia IPv6, lleve a cabo los pasos siguientes.

i. Descargue el manifiesto.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.14.1/docs/examples/2048/2048_full.yaml
```

ii. Abra el archivo en un editor y agregue la siguiente línea a las anotaciones de la especificación de entrada.

```
alb.ingress.kubernetes.io/ip-address-type: dualstack
```

iii. Si equilibra la carga en pods internos, en lugar de en pods orientados a Internet, cambie la línea que dice `alb.ingress.kubernetes.io/scheme:` *`internet-facing`* por `alb.ingress.kubernetes.io/scheme: internal`.

iv. Guarde el archivo.

v. Aplique el manifiesto al clúster.

```
kubectl apply -f 2048_full.yaml
```

3. Al cabo de unos minutos, verifique que el recurso de entrada se haya creado con el comando siguiente.

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

Un ejemplo de salida sería el siguiente.

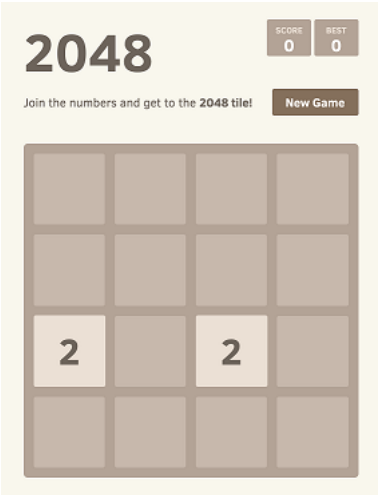
```
NAME           CLASS    HOSTS   ADDRESS
                      PORTS   AGE
ingress-2048   <none>   *       k8s-game2048-ingress2-xxxxxxxxxx-yyyyyyyyyy.region-
code.elb.amazonaws.com   80      2m32s
```

> **Note**
>
> Si ha creado el equilibrador de carga en una subred privada, el valor de `ADDRESS` en la salida anterior aparece precedido por `internal-`.

Si la entrada no se creó correctamente después de varios minutos, ejecute el siguiente comando para ver los registros del Controlador del equilibrador de carga de AWS. Estos registros pueden contener mensajes de error que pueden ayudarlo a diagnosticar cualquier problema con la implementación.

```
kubectl logs -f -n kube-system -l app.kubernetes.io/instance=aws-load-balancer-
controller
```

1. Si ha realizado la implementación en una subred pública, abra un navegador y navegue hasta la URL `ADDRESS` de la salida del comando anterior para ver la aplicación de muestra. Si no ve nada, actualice el navegador e inténtelo de nuevo. Si ha realizado la implementación en una subred privada, tendrá que ver la página desde un dispositivo dentro de la VPC, como un host bastión. Para obtener más información, consulte Hosts bastión de Linux en AWS.



2. Cuando termine de experimentar con la aplicación de muestra, elimínela ejecutando alguno de los siguientes comandos.
   - Si aplicó el manifiesto, en lugar de aplicar una copia que haya descargado, utilice el siguiente comando.

     ```
     kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.14.1/docs/examples/2048/2048_full.yaml
     ```

   - Si descargó y editó el manifiesto, utilice el siguiente comando.

     ```
     kubectl delete -f 2048_full.yaml
     ```

Cómo restringir las direcciones IP externas que se pueden asignar a los servicios

Se puede acceder a los servicios de Kubernetes desde el interior de un clúster a través de:

- Una dirección IP de clúster asignada automáticamente por Kubernetes.
- Cualquier dirección IP que especifique para la propiedad `externalIPs` en una especificación de servicio. Las direcciones IP externas no son administradas por Kubernetes y son responsabilidad del administrador del clúster. Las direcciones IP externas especificadas con `externalIPs` son diferentes de la dirección IP externa asignada a un servicio de tipo `LoadBalancer` por un proveedor de nube.

Para obtener más información sobre los servicios de Kubernetes, consulte Servicio en la documentación de Kubernetes. Puede restringir las direcciones IP que se pueden especificar para `externalIPs` en una especificación de servicio.

1. Implemente `cert-manager` para administrar certificados de webhook. Para obtener más información, consulte la documentación de cert-manager.

   ```
   kubectl apply -f https://github.com/jetstack/cert-manager/releases/download/v1.5.4/cert-manager.yaml
   ```

2. Compruebe que se están ejecutando los pods de `cert-manager`.

```
kubectl get pods -n cert-manager
```

Un ejemplo de salida sería el siguiente.

```
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. Revise sus servicios existentes para asegurarse de que ninguno de ellos tenga asignadas direcciones IP externas que no estén incluidas en el bloque de CIDR al que desea limitar las direcciones.

```
kubectl get services -A
```

Un ejemplo de salida sería el siguiente.

```
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
```

Si alguno de los valores son direcciones IP que no están dentro del bloque al que desea restringir el acceso, deberá cambiar las direcciones para que estén dentro del bloque y volver a implementar los servicios. Por ejemplo, el servicio `my-service` de la salida anterior tiene asignada una dirección IP externa que no está dentro del ejemplo de bloque de CIDR en el paso 5.

4. Descargue el manifiesto de webhook de la IP externa. También puede ver el código fuente para el webhook en GitHub.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/docs/externalip-webhook.yaml
```

5. Especifica los bloques de CIDR. Abra el archivo descargado en el editor y elimine el \# al comienzo de las siguientes líneas.

```
#args:
#- --allowed-external-ip-cidrs=10.0.0.0/8
```

   Reemplace `10.0.0.0/8` por su propio bloque de CIDR. Puede especificar tantos bloques como desee. Si especifica varios bloques, agregue una coma entre bloques.

6. Si su clúster no está en la región de AWS `us-west-2`, reemplace `us-west-2`, `602401143452` y `amazonaws.com` en el archivo con los siguientes comandos. Antes de ejecutar los comandos, sustituya el *`region-code`* y *`111122223333`* por el valor de su región de AWS de la lista que se encuentra en Visualización de los registros de imágenes de contenedores de Amazon para los complementos de Amazon EKS.

```
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. Aplique el manifiesto al clúster.

```
kubectl apply -f externalip-webhook.yaml
```

   Se producirá un error al intentar implementar un servicio en el clúster con una dirección IP especificada para `externalIPs` que no está incluida en los bloques que especificó en el paso Especificar bloques CIDR.

Copiar una imagen de contenedor de un repositorio en otro repositorio

En este tema se describe cómo extraer una imagen del contenedor de un repositorio al que los nodos no tienen acceso y enviar la imagen a un repositorio al que tienen acceso los nodos. Puede enviar la imagen a Amazon ECR o a un repositorio alternativo al que tengan acceso sus nodos.

- El motor de Docker instalado y configurado en su computadora. Para ver instrucciones, consulte Install Docker Engine (Instalar motor de Docker) en la documentación de Docker.
- La versión `2.12.3` o posterior, o bien, la versión `1.27.160` o posterior de la AWS interfaz de la línea de comandos (AWS CLI) instalada y configurada en su dispositivo o AWS CloudShell. Para comprobar su versión actual, utilice `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente. Para actualizarla, consulte Instalación de la CLI de AWS en su directorio de usuarios principal en la Guía del usuario de AWS CloudShell.
- Un punto de conexión de VPC de interfaz para Amazon ECR, si desea que los nodos extraigan imágenes de contenedores o envíen imágenes de contenedores a un repositorio privado de Amazon ECR a través de la red de Amazon. Para obtener más información, consulte Crear los puntos de conexión de VPC para Amazon ECR en la Guía del usuario de Amazon Elastic Container Registry.

Siga los siguientes pasos para extraer una imagen de contenedor de un repositorio y enviarla a su propio repositorio. En los siguientes ejemplos que se proporcionan en este tema, se extrae la imagen del complemento CNI de Amazon VPC para el auxiliar de métricas de Kubernetes. Cuando siga estos pasos, asegúrese de reemplazar los valores de ejemplo por sus propios valores.

1. Si todavía no tiene un repositorio de Amazon ECR u otro repositorio, cree uno al que tengan acceso sus nodos. El siguiente comando crea un repositorio privado de Amazon ECR. El nombre de un repositorio privado de Amazon ECR debe comenzar por una letra. Solo puede contener letras minúsculas, números, guiones (-), guiones bajos (_) y barras inclinadas (/). Para obtener más información, consulte Creación de un repositorio privado en la Guía del usuario de Amazon Elastic Container Registry.

   Puede reemplazar *`cni-metrics-helper`* por lo que elija. Como práctica recomendada, cree un repositorio independiente para cada imagen. Recomendamos esto porque las etiquetas de imagen deben ser únicas dentro de un repositorio. Reemplace *`region-code`* por una Región de AWS compatible con Amazon ECR.

   ```
   aws ecr create-repository --region region-code --repository-name cni-metrics-helper
   ```

2. Determine el registro, el repositorio y la etiqueta (opcional) de la imagen que deben extraer los nodos. Esta información se encuentra en el formato `registry/repository[:tag]`.

   Muchos de los temas de Amazon EKS sobre la instalación de imágenes requieren aplicar un archivo de manifiesto o instalar la imagen mediante un gráfico de Helm. Sin embargo, antes de aplicar un archivo de manifiesto o instalar un gráfico de Helm, consulte primero el contenido del manifiesto o el archivo `values.yaml` del gráfico. Así podrá determinar el registro, el repositorio y la etiqueta que desea extraer.

   Por ejemplo, puede encontrar la siguiente línea en el archivo de manifiesto para el complemento CNI de Amazon VPC para el auxiliar de métricas de Kubernetes. El registro es `602401143452.dkr.ecr.us-west-2.amazonaws.com`, que es un registro privado de Amazon ECR. El repositorio es `cni-metrics-helper`.

   ```
   image: "602401143452.dkr.ecr.us-west-2.amazonaws.com/cni-metrics-helper:v1.12.6"
   ```

   Podría ver las siguientes variaciones para la ubicación de una imagen:

   - Solo `repository-name:tag`. En este caso, `docker.io` suele ser el registro, pero no se especifica ya que Kubernetes lo antepone a un nombre de repositorio de forma predeterminada si no se especifica ningún registro.
   - `repository-name/repository-namespace/repository:tag`. Es opcional definir un espacio de nombres para el repositorio, pero a veces lo especifica el propietario del repositorio para clasificar las imágenes. Por ejemplo, todas las imágenes de Amazon EC2 de la galería pública de Amazon ECR usan el espacio de nombres `aws-ec2`.

     Antes de instalar una imagen con Helm, consulte el archivo `values.yaml` de Helm para determinar la ubicación de la imagen. Por ejemplo, el archivo values.yaml para el complemento CNI de Amazon VPC para el auxiliar de métricas de Kubernetes incluye las siguientes líneas.

     ```
     image:
       region: us-west-2
       tag: v1.12.6
       account: "602401143452"
       domain: "amazonaws.com"
     ```

3. Extraiga la imagen del contenedor especificada en el archivo de manifiesto.

   a. Si la extracción es de un registro público, como la Galería pública de Amazon ECR, puede pasar al siguiente subpaso porque no es necesaria la autenticación. En este ejemplo,

se autentica en un registro privado de Amazon ECR que contiene el repositorio de la imagen auxiliar de métricas de CNI. Amazon EKS mantiene la imagen en cada registro que aparece en Visualización de los registros de imágenes de contenedores de Amazon para los complementos de Amazon EKS. Puede autenticarse en cualquiera de los registros reemplazando *602401143452* y *region-code* por la información de otro registro. Existe un registro independiente para cada Región de AWS compatible con Amazon EKS.

```
aws ecr get-login-password --region region-code | docker login --username AWS --
password-stdin 602401143452.dkr.ecr.region-code.amazonaws.com
```

b. Extraiga la imagen. En este ejemplo, extrae del registro en el que se autenticó en el subpaso anterior. Reemplace *602401143452* y *region-code* por la información proporcionada en el subpaso anterior.

```
docker pull 602401143452.dkr.ecr.region-code.amazonaws.com/cni-metrics-
helper:v1.12.6
```

4. Etiquete la imagen extraída con su registro, repositorio y etiqueta. En el siguiente ejemplo se supone que ha extraído la imagen del archivo de manifiesto y la va a enviar al repositorio privado de Amazon ECR que creó en el primer paso. Reemplace *111122223333* por el ID de su cuenta. Reemplace *region-code* por la región de AWS que creó en su repositorio privado de Amazon ECR.

```
docker tag cni-metrics-helper:v1.12.6 111122223333.dkr.ecr.region-code.amazonaws.com/
cni-metrics-helper:v1.12.6
```

5. Realice la autenticación del registro. En este ejemplo, se autenticará en el registro privado de Amazon ECR que creó en el primer paso. Para obtener más información, consulte Autenticación de registros en la Guía del usuario de 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. Inserte la imagen en el repositorio. En este ejemplo, envía la imagen en el repositorio privado de Amazon ECR que creó en el primer paso. Para obtener más información, consulte Inserción de una imagen de Docker en la Guía del usuario de Amazon Elastic Container Registry.

```
docker push 111122223333.dkr.ecr.region-code.amazonaws.com/cni-metrics-helper:v1.12.6
```

7. Actualice el archivo de manifiesto que utilizó para determinar la imagen en un paso anterior con el `registry/repository:tag` para la imagen que envió. Si hace la instalación con un gráfico de Helm, a menudo hay una opción para especificar el `registry/repository:tag`. Al instalar el gráfico, especifique el `registry/repository:tag` para la imagen que envió a su repositorio.

Visualización de los registros de imágenes de contenedores de Amazon para los complementos de Amazon EKS

Al implementar complementos de Amazon EKS de AWS en su clúster, sus nodos extraen las imágenes del contenedor necesarias del registro especificado en el mecanismo de instalación del complemento, como un manifiesto de instalación o un archivo Helm `values.yaml`. Las imágenes se extraen de un repositorio privado de Amazon ECR de Amazon EKS. Amazon EKS replica las imágenes en un repositorio de cada región de AWS compatible con Amazon EKS. Los nodos pueden extraer la imagen del contenedor a través de Internet desde cualquiera de los siguientes registros. Como alternativa, sus nodos pueden extraer la imagen a través de la red de Amazon si ha creado un punto de conexión de VPC de interfaz para Amazon ECR (AWS PrivateLink) en la VPC. Los registros requieren autenticación con una cuenta de AWS IAM. Los nodos se autentican mediante el rol de IAM del nodo de Amazon EKS, que tiene asociados a él los permisos de la política de IAM administrada AmazonEC2ContainerRegistryReadOnly.

Región de AWS	Registro
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
ap-southeast-3	296578399912.dkr.ecr.ap-southeast-3.amazonaws.com
ap-south-2	900889452093.dkr.ecr.ap-south-2.amazonaws.com

Región de AWS	Registro
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
eu-central-1	602401143452.dkr.ecr.eu-central-1.amazonaws.com
eu-west-1	602401143452.dkr.ecr.eu-west-1.amazonaws.com

Región de AWS	Registro
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
us-east-2	602401143452.dkr.ecr.us-east-2.amazonaws.com
us-west-1	602401143452.dkr.ecr.us-west-1.amazonaws.com

Región de AWS	Registro
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

Complementos de Amazon EKS

Un complemento es un software que proporciona capacidades operativas de soporte para aplicaciones de Kubernetes, pero no es específico de la aplicación. Esto incluye software como agentes de observabilidad o controladores de Kubernetes que permiten al clúster interactuar con recursos subyacentes de AWS para redes, informática y almacenamiento. La comunidad de Kubernetes, proveedores de nube como AWS o proveedores de terceros suelen crear y mantener este tipo de software. Amazon EKS instala de forma automática complementos autoadministrados como CNI de Amazon VPC para Kubernetes, `kube-proxy` y CoreDNS para cada clúster. Tenga en cuenta que el complemento de CNI de la VPC no es compatible con los Nodos híbridos de Amazon EKS y no se implementa en nodos híbridos. Puede cambiar la configuración predeterminada de los complementos y actualizarlos cuando lo desee.

Los complementos de Amazon EKS proporcionan la instalación y administración de un conjunto seleccionado de complementos para clústeres de Amazon EKS. Todos los complementos de Amazon EKS incluyen los parches de seguridad más recientes, correcciones de errores y están validados por AWS para trabajar con Amazon EKS. Los complementos de Amazon EKS permiten garantizar de forma coherente que los clústeres de Amazon EKS sean seguros y estables, así como

reducir la cantidad de trabajo necesario para instalar, configurar y actualizar complementos. Si un complemento autoadministrado, como `kube-proxy`, ya se ejecuta en su clúster y está disponible como complemento de Amazon EKS, a continuación, puede instalar el complemento `kube-proxy` de Amazon EKS para comenzar a beneficiarse de las capacidades de los complementos de Amazon EKS.

Puede actualizar campos de configuración administrados de Amazon EKS específicos para complementos de Amazon EKS a través de la API de Amazon EKS. Además, puede modificar los campos de configuración no administrados por Amazon EKS directamente en el clúster de Kubernetes una vez que se inicie el complemento. Esto incluye la definición de campos de configuración específicos para un complemento cuando corresponda. Amazon EKS no anulará estos cambios una vez realizados. Esto es posible mediante la característica de aplicación del lado del servidor de Kubernetes. Para obtener más información, consulte the section called “Campos que puede personalizar”.

Puede usar complementos de Amazon EKS con cualquier tipo de nodos de Amazon EKS. Para obtener más información, consulte Administración de la computación.

Puede agregar, actualizar o eliminar complementos de Amazon EKS mediante la API de Amazon EKS, la Consola de administración de AWS, la CLI de AWS y `eksctl`. También puede crear complementos de Amazon EKS mediante AWS CloudFormation.

Consideraciones

Tenga en cuenta lo siguiente cuando utilice los complementos de Amazon EKS:

- Para configurar complementos para el clúster, la entidad principal de IAM debe tener permisos de IAM para trabajar con complementos. Para obtener más información, consulte las acciones con `Addon` en su nombre en Acciones definidas por Amazon Elastic Kubernetes Service.
- Los complementos de Amazon EKS se ejecutan en los nodos que aprovisiona o configura para el clúster. Los tipos de nodo incluyen instancias de Amazon EC2, Fargate y nodos híbridos.
- Puede modificar campos que no estén administrados por Amazon EKS para personalizar la instalación de un complemento de Amazon EKS. Para obtener más información, consulte the section called “Campos que puede personalizar”.
- Si crea un clúster con la Consola de administración de AWS, el `kube-proxy` de Amazon EKS, los complementos CNI de Amazon VPC para Kubernetes y CoreDNS de Amazon EKS se agregan automáticamente al clúster. Si utiliza `eksctl` para crear el clúster con un archivo de `config`, `eksctl` también puede crear el clúster con complementos de Amazon EKS. Si crea el

clúster con `eksctl` sin un archivo de `config` o con cualquier otra herramienta, el `kube-proxy` autoadministrado y los complementos CNI de Amazon VPC para Kubernetes y CoreDNS están instalados, en lugar de los complementos de Amazon EKS. Puede administrarlos o agregar los complementos de Amazon EKS de forma manual después de crear el clúster. Independientemente del método que utilice para crear el clúster, el complemento de CNI de la VPC no se instala en los nodos híbridos.

- El `ClusterRoleBinding` `eks:addon-cluster-admin` une el `ClusterRole` `cluster-admin` a la identidad de Kubernetes `eks:addon-manager`. El rol tiene los permisos necesarios para que la identidad de `eks:addon-manager` cree espacios de nombres de Kubernetes e instale complementos en los espacios de nombres. Si el `ClusterRoleBinding` `eks:addon-cluster-admin` se elimina, el clúster de Amazon EKS seguirá funcionando; sin embargo, Amazon EKS ya no podrá administrar ningún complemento. Todos los clústeres que comiencen con las siguientes versiones de plataforma utilizan el nuevo `ClusterRoleBinding`.
- Se ha validado la compatibilidad de un subconjunto de complementos de EKS de AWS con los Nodos híbridos de Amazon EKS. Para obtener más información, consulte la tabla de compatibilidad que se encuentra en the section called “ Complementos de AWS”.

Espacio de nombres personalizado para complementos

Para los complementos de AWS y de la comunidad, tiene la opción de especificar un espacio de nombres personalizado durante la creación del complemento. Una vez que instale un complemento en un espacio de nombres específico, debe eliminarlo y volver a crearlo para cambiar su espacio de nombres.

Si no especifica ningún espacio de nombres, se utilizará el espacio de nombres predefinido para el complemento.

Utilice espacios de nombres personalizados para una mejor organización y aislamiento de los objetos de complementos dentro del clúster de EKS. Esta flexibilidad le ayuda a alinear los complementos con sus necesidades operativas y la estrategia de espacios de nombres existente.

Puede configurar un espacio de nombres personalizado al crear un complemento. Para obtener más información, consulte the section called “Cómo crear un complemento”.

Obtención de un espacio de nombres predefinido para un complemento

El espacio de nombres predefinido para un complemento es el espacio de nombres en el que se instalará si no se especifica ninguno.

Para obtener el espacio de nombres predefinido para un complemento, utilice el siguiente comando:

```
aws eks describe-addon-versions --addon-name <addon-name> --query
 "addons[].defaultNamespace"
```

Ejemplo de código de salida:

```
[
    "kube-system"
]
```

Consideraciones para el modo automático de Amazon EKS

El modo automático de Amazon EKS incluye capacidades que ofrecen funcionalidades esenciales del clúster, entre las que se incluyen:

- Conexión de pods en red
- Conexión de servicios en red
- DNS de clúster
- Escalado automático
- Almacenamiento en bloque
- Controlador del equilibrador de carga
- Agente de Pod Identity
- Agente de supervisión de nodos

Con la computación del modo automático, muchos complementos de EKS de uso común se vuelven redundantes, como:

- CNI de Amazon VPC
- kube-proxy
- CoreDNS
- Controlador de CSI de Amazon EBS
- Agente de Pod Identity de EKS

Sin embargo, si el clúster combina el modo automático con otras opciones de computación, como instancias de EC2 autoadministradas, grupos de nodos administrados o AWS Fargate, aún se necesitarán estos complementos. AWS ha mejorado los complementos de EKS con reglas antiafinidad que garantizan automáticamente que los pods complementarios se programen solo en los tipos de computación compatibles. Además, ahora los usuarios pueden aprovechar la API `DescribeAddonVersions` de complementos de EKS para verificar los computeTypes admitidos para cada complemento y sus versiones específicas. Además, con el modo automático de EKS, los controladores mencionados anteriormente se ejecutan en la infraestructura propiedad de AWS. Por lo tanto, es posible que ni siquiera los vea en las cuentas, a menos que utilice el modo automático de EKS con otros tipos de computación, en cuyo caso verá los controladores instalados en el clúster.

Si tiene previsto habilitar el modo automático de EKS en un clúster existente, es posible que deba actualizar la versión de determinados complementos. Para obtener más información, consulte the section called “Versiones del complemento requeridas” para el modo automático de EKS.

Soporte

AWS publica varios tipos de complementos con diferentes niveles de soporte.

- Complementos de AWS: estos complementos son creados por AWS y cuentan con pleno soporte.
  - Utilice un complemento de AWS para trabajar con otros servicios de AWS, como Amazon EFS.
  - Para obtener más información, consulte the section called “ Complementos de AWS”.
- Complementos de AWS Marketplace: estos complementos son analizados por AWS y cuentan con el soporte de un socio de AWS independiente.
  - Utilice un complemento del Marketplace para aportar características valiosas y sofisticadas al clúster, como la supervisión con Splunk.
  - Para obtener más información, consulte the section called “Complementos del Marketplace”.
- Complementos de la comunidad: AWS analiza estos complementos, pero el soporte está a cargo de la comunidad de código abierto.
  - Utilice un complemento de la comunidad para reducir la complejidad que supone instalar software común de código abierto, como el Servidor de métricas de Kubernetes.
  - El origen de los paquetes de complementos de la comunidad es AWS. AWS solo valida los complementos de la comunidad para la compatibilidad de versiones.
  - Para obtener más información, consulte the section called “Complementos de la comunidad”.

En la tabla siguiente se indica el alcance del soporte para cada tipo de complemento:

Categoría	Característica	Complementos de AWS	Complementos de AWS Marketplace	Complementos de la comunidad
Desarrollo	Creado por AWS	Sí	No	Sí
Desarrollo	Validado por AWS	Sí	No	Sí*
Desarrollo	Validado por un socio de AWS	No	Sí	No
Mantenimiento	Analizado por AWS	Sí	Sí	Sí
Mantenimiento	Revisado por AWS	Sí	No	Sí
Mantenimiento	Revisado por un socio de AWS	No	Sí	No
Distribución	Publicado por AWS	Sí	No	Sí
Distribución	Publicado por un socio de AWS	No	Sí	No
Soporte	Soporte de instalación básico proporcionado por AWS	Sí	Sí	Sí
Soporte	Soporte pleno por parte de AWS	Sí	No	No
Soporte	Soporte pleno por parte de un socio de AWS	No	Sí	No

*: la validación de los complementos de la comunidad solo incluye la compatibilidad de las versiones de Kubernetes. Por ejemplo, si instala un complemento comunitario en un clúster, AWS comprueba si es compatible con la versión de Kubernetes del clúster.

Los complementos de AWS Marketplace pueden descargar dependencias de software adicionales desde fuentes de terceros fuera de AWS. AWS no analiza ni válida estas dependencias de terceros. Tenga en cuenta los requisitos de seguridad al implementar complementos de AWS Marketplace que obtienen dependencias externas.

Complementos de AWS

Los siguientes complementos de Amazon EKS están disponibles para crearlos en el clúster. Puede ver la lista más actualizada de complementos disponibles mediante `eksctl`, la Consola de administración de AWS o la AWS CLI. Para ver todos los complementos disponibles o instalar un complemento, consulte the section called “Cómo crear un complemento”. Si un complemento requiere permisos de IAM, debe tener un proveedor de OpenID Connect (OIDC) de IAM para el clúster. Para determinar si ya tiene uno o si debe crearlo, consulte the section called “Proveedor de OIDC de IAM”. Puede crear o eliminar un complemento una vez que lo haya instalado. Para obtener más información, consulte the section called “Cómo actualizar un complemento” o the section called “Cómo eliminar un complemento”. Para obtener más información sobre las consideraciones específicas al ejecutar complementos de EKS con Nodos híbridos de Amazon EKS, consulte the section called “Configuración de complementos”.

Puede utilizar cualquiera de los siguientes complementos de Amazon EKS.

Descripción	Más información	Tipos de computación compatibles
Proporcione redes de VPC nativas para su clúster.	the section called “Complemento CNI de Amazon VPC para Kubernetes”	EC2
Un servidor de DNS flexible y extensible que puede servir como el DNS del clúster de Kubernetes	the section called “CoreDNS”	EC2, Fargate, Modo automático de EKS, Nodos híbridos de EKS

Descripción	Más información	Tipos de computación compatibles
Mantenga las reglas de red en cada nodo de Amazon EC2.	the section called “Kube-proxy ”	EC2, Nodos híbridos de EKS
Proporcione almacenamiento de Amazon EBS para su clúster.	the section called “Controlador CSI de Amazon EBS”	EC2
Proporcione almacenamiento de Amazon EFS para el clúster.	the section called “Controlador CSI de Amazon EFS”	EC2, Modo automático de EKS
Proporcione almacenamiento de Amazon FSx para Lustre para su clúster	the section called “Controlador CSI de Amazon FSx”	EC2, Modo automático de EKS
Proporcione almacenamiento de Amazon S3 para el clúster.	the section called “Controlador CSI del Mountpoint para Amazon S3”	EC2, Modo automático de EKS
Detecte otros problemas de estado de los nodos	the section called “Agente de supervisión de nodos”	EC2, Nodos híbridos de EKS
Habilite el uso de la funcionalidad de captura de instantáneas en controladores de CSI compatibles, como el controlador de CSI de Amazon EBS.	the section called “Controlador de instantáneas CSI”	EC2, Fargate, Modo automático de EKS, Nodos híbridos de EKS

Descripción	Más información	Tipos de computación compatibles
La gobernanza de tareas de SageMaker HyperPod optimiza la asignación y el uso de los recursos de computación entre los equipos de los clústeres de Amazon EKS, abordando así las ineficiencias en la priorización de tareas y el uso compartido de recursos.	the section called “Gobernanza de tareas de Amazon SageMaker HyperPod”	EC2, Modo automático de EKS,
El complemento de observabilidad de Amazon SageMaker HyperPod proporciona capacidades integrales de supervisión y observabilidad para los clústeres de HyperPod.	the section called “Complemento de observabilidad para Amazon SageMaker HyperPod”	EC2, Modo automático de EKS,
El operador de entrenamiento de Amazon SageMaker HyperPod permite un entrenamiento distribuido eficiente en los clústeres de Amazon EKS con capacidades avanzadas de administración de recursos y programación.	the section called “Operador de entrenamiento de Amazon SageMaker HyperPod”	EC2, Modo automático de EKS

Descripción	Más información	Tipos de computación compatibles
Un agente de Kubernetes que recopila datos de flujo de red y los reporta a Amazon CloudWatch, permitiendo una supervisión integral de las conexiones TCP en los nodos del clúster.	the section called “Agente del monitor de flujo de red de AWS”	EC2, Modo automático de EKS
Distribución segura, lista para la producción y compatible con AWS del proyecto OpenTelemetry.	the section called “AWS Distro para OpenTelemetry”	EC2, Fargate, Modo automático de EKS, Nodos híbridos de EKS
Servicio de supervisión de seguridad que analiza y procesa los orígenes de datos fundacionales, incluidos los eventos de administración de AWS CloudTrail y los registros de flujo de Amazon VPC. Amazon GuardDuty también procesa características, como los registros de auditoría de Kubernetes y la supervisión del tiempo de ejecución.	the section called “Agente Amazon GuardDuty”	EC2, Modo automático de EKS

Descripción	Más información	Tipos de computación compatibles
Servicio de supervisión y observabilidad proporcionado por AWS. Este complemento instala el agente de CloudWatch y habilita tanto CloudWatch Application Signals como Información de contenedores de Amazon CloudWatch con una observabilidad mejorada para Amazon EKS	the section called “Agente de Amazon CloudWatch Observability”	EC2, Modo automático de EKS, Nodos híbridos de EKS
Ofrece la posibilidad de administrar las credenciales de las aplicaciones, de un modo similar a cómo los perfiles de instancia de EC2 proporcionan credenciales a instancias de EC2.	the section called “Agente de Pod Identity de EKS”	EC2, Nodos híbridos de EKS
Active cert-manager para emitir certificados X.509 desde AWS Private CA. Requiere cert-manager.	the section called “Conector de AWS Private CA para Kubernetes”	EC2, Fargate, Modo automático de EKS, Nodos híbridos de EKS
Generación de métricas de Prometheus sobre el rendimiento de los dispositivos de red SR-IOV	the section called “Exportador de métricas de red SR-IOV”	EC2

Descripción	Más información	Tipos de computación compatibles
Recupere los secretos desde AWS Secrets Manager y los parámetros del Almacén de parámetros de AWS Systems Manager y móntelos como archivos en los pods de Kubernetes.	the section called “Proveedor de AWS para el controlador CSI del Almacén de secretos”	EC2, Modo automático de EKS, Nodos híbridos de EKS
Con Espacios, puede crear y administrar aplicaciones de JupyterLab y el Editor de código para ejecutar cargas de trabajo de ML interactivas.	the section called “Espacios de Amazon SageMaker”	HyperPod

Complemento CNI de Amazon VPC para Kubernetes

El complemento CNI de Amazon VPC para Kubernetes de Amazon EKS es un complemento de interfaz de red de contenedores (CNI) de Kubernetes que proporciona redes de VPC nativas para el clúster. El tipo autoadministrado o administrado de este complemento se instala en cada nodo de Amazon EC2 de forma predeterminada. Para obtener más información, consulte Complemento de Interfaz de red de contenedores (CNI) de Kubernetes.

> **Note**
>
> No es necesario instalar este complemento en los clústeres del modo automático de Amazon EKS. Para obtener más información, consulte the section called “Consideraciones para el modo automático de Amazon EKS”.

El nombre del complemento de Amazon EKS es `vpc-cni`.

Permisos de IAM necesarios

Este complemento utiliza la capacidad de roles de IAM para cuentas de servicio de Amazon EKS. Para obtener más información, consulte the section called “Credenciales con IRSA”.

Si el clúster usa la familia `IPv4`, se requieren los permisos de la AmazonEKS_CNI_Policy. Si el clúster utiliza la familia `IPv6`, entonces debe crear una política de IAM con los permisos del modo IPv6. Puede crear un rol de IAM, adjuntarle una de las políticas y anotar la cuenta de servicio de Kubernetes utilizada por el complemento con el siguiente comando.

Reemplace *`my-cluster`* por el nombre del clúster y *`AmazonEKSVPCCNIRole`* por el nombre del rol. Si el clúster usa la familia `IPv6`, reemplace *`AmazonEKS_CNI_Policy`* por el nombre de la política que creó. Este comando requiere que tenga eksctl instalado en su dispositivo. Si necesita usar una herramienta diferente para crear el rol, adjuntarle la política y anotar la cuenta de servicio de Kubernetes, consulte the section called “Asignación del rol de IAM”.

```
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
```

Actualización de la información

Solo puede actualizar una versión secundaria a la vez. Por ejemplo, si su versión actual es la `1.28.`*`x`*`-eksbuild.`*`y`* y desea actualizarla a la `1.30.`*`x`*`-eksbuild.`*`y`*, primero debe actualizarla a la `1.29.`*`x`*`-eksbuild.`*`y`* y luego a la `1.30.`*`x`*`-eksbuild.`*`y`*. Para obtener más información acerca de la actualización de complementos, consulte the section called “Actualización (del complemento de EKS)”.

CoreDNS

El complemento de CoreDNS de Amazon EKS es un servidor de DNS flexible y extensible que puede servir como el DNS del clúster de Kubernetes. El tipo autoadministrado o administrado de este complemento se instaló de forma predeterminada cuando se creó el clúster. Al lanzar un clúster de Amazon EKS con al menos un nodo, se implementan dos réplicas de la imagen de CoreDNS de forma predeterminada, independientemente del número de nodos implementados en el clúster. Los pods de CoreDNS proporcionan resolución de nombres para todos los pods del clúster. Los pods de CoreDNS se pueden implementar en los nodos de Fargate si el clúster incluye un perfil de Fargate con un espacio de nombres que coincida con el espacio de nombres para la implementación de CoreDNS. Para obtener más información, consulte the section called “Definición de perfiles”

> **Note**
>
> No es necesario instalar este complemento en los clústeres del modo automático de Amazon EKS. Para obtener más información, consulte the section called “Consideraciones para el modo automático de Amazon EKS”.

El nombre del complemento de Amazon EKS es `coredns`.

Permisos de IAM necesarios

Este complemento no requiere ningún permiso.

Información adicional

Para obtener más información sobre CoreDNS, consulte Using CoreDNS for Service Discovery y Customizing DNS Service en la documentación de Kubernetes.

`Kube-proxy`

El complemento de `Kube-proxy` de Amazon EKS mantiene las reglas de red en cada nodo de Amazon EC2. Permite la comunicación de red con sus pods. De forma predeterminada, el tipo autoadministrado o administrado de este complemento se instala en cada nodo de Amazon EC2 en el clúster.

> **Note**
>
> No es necesario instalar este complemento en los clústeres del modo automático de Amazon EKS. Para obtener más información, consulte the section called “Consideraciones para el modo automático de Amazon EKS”.

El nombre del complemento de Amazon EKS es `kube-proxy`.

Permisos de IAM necesarios

Este complemento no requiere ningún permiso.

Actualización de la información

Antes de actualizar la versión actual, tenga en cuenta los siguientes requisitos:

- `Kube-proxy` en un clúster de Amazon EKS tiene la misma política de compatibilidad y sesgo que Kubernetes.

Información adicional

Para obtener más información sobre `kube-proxy`, consulte kube-proxy en la documentación de Kubernetes.

Controlador CSI de Amazon EBS

El complemento del controlador de CSI de Amazon EBS para Amazon EKS es un complemento de interfaz de almacenamiento de contenedores (CSI) de Kubernetes que proporciona almacenamiento de Amazon EBS para el clúster.

> **Note**
>
> No es necesario instalar este complemento en los clústeres del modo automático de Amazon EKS. El modo automático incluye una función de almacenamiento en bloque. Para obtener más información, consulte the section called “Implementación de una carga de trabajo con estado”.

El nombre del complemento de Amazon EKS es `aws-ebs-csi-driver`.

Permisos de IAM necesarios

Este complemento utiliza la capacidad de roles de IAM para cuentas de servicio de Amazon EKS. Para obtener más información, consulte the section called “Credenciales con IRSA”. Los permisos de la política administrada AmazonEBSCSIDriverPolicy de AWS son obligatorios. Cree un rol de IAM y adjunte la política administrada del rol de IAM con el siguiente comando. Reemplace *`my-cluster`* por el nombre del clúster y *`AmazonEKS_EBS_CSI_DriverRole`* por el nombre del rol. Este comando requiere que tenga eksctl instalado en su dispositivo. Si necesita usar una herramienta diferente o necesita usar una clave de KMS personalizada para el cifrado, consultethe section called “Paso 1: creación de un rol de IAM”.

```
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
```

Información adicional

Para obtener más información sobre el complemento, consulte the section called “Amazon EBS”.

Controlador CSI de Amazon EFS

El complemento del controlador de CSI de Amazon EFS para Amazon EKS es un complemento de interfaz de almacenamiento de contenedores (CSI) de Kubernetes que proporciona almacenamiento de Amazon EFS para el clúster.

El nombre del complemento de Amazon EKS es `aws-efs-csi-driver`.

Permisos de IAM necesarios

Permisos de IAM requeridos: este complemento utiliza la capacidad roles de IAM para cuentas de servicio de Amazon EKS. Para obtener más información, consulte the section called “Credenciales con IRSA”. Los permisos de la política administrada AmazonEFSCSIDriverPolicy de AWS son obligatorios. Cree un rol de IAM y adjunte la política administrada del rol de IAM con los siguientes comandos. Reemplace *`my-cluster`* por el nombre del clúster y *`AmazonEKS_EFS_CSI_DriverRole`* por el nombre del rol. Estos comandos requieren que tenga eksctl instalado en su dispositivo. Si necesita utilizar una herramienta diferente, consulte the section called “Paso 1: creación de un rol de IAM”.

```
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"
```

Información adicional

Para obtener más información sobre el complemento, consulte the section called “Amazon EFS”.

Controlador CSI de Amazon FSx

El complemento del controlador de CSI de Amazon ELS para Amazon FSx es un complemento de interfaz de almacenamiento de contenedores (CSI) de Kubernetes que proporciona almacenamiento de Amazon FSx para Lustre para el clúster.

El nombre del complemento de Amazon EKS es `aws-fsx-csi-driver`.

> **Note**
>
> - Las instalaciones preexistentes del controlador de CSI de Amazon FSx en el clúster pueden provocar errores en la instalación de los complementos. Si intenta instalar la versión del complemento de Amazon EKS mientras existe un controlador de CSI de FSx que no es de EKS, se producirá un error en la instalación debido a conflictos de recursos. Utilice la marca `OVERWRITE` durante la instalación para resolver este problema:
>
>   ```
>   aws eks create-addon --addon-name aws-fsx-csi-driver --cluster-name my-cluster
>    --resolve-conflicts OVERWRITE
>   ```
>
> - El complemento de EKS del controlador de CSI de Amazon FSx requiere el agente de EKS Pod Identity para la autenticación. Sin este componente, se producirá el error `Amazon EKS Pod Identity agent is not installed in the cluster` en el complemento, que impedirá las operaciones de volumen. Instale el agente de Pod Identity antes o después de implementar el complemento del controlador de CSI de FSx. Para obtener más información, consulte the section called “Configuración del agente”.

Permisos de IAM necesarios

Este complemento utiliza la capacidad de roles de IAM para cuentas de servicio de Amazon EKS. Para obtener más información, consulte the section called “Credenciales con IRSA”. Los permisos de la política administrada AmazonFSxFullAccess de AWS son obligatorios. Cree un rol de IAM y

adjunte la política administrada del rol de IAM con el siguiente comando. Reemplace *my-cluster* por el nombre del clúster y *AmazonEKS_FSx_CSI_DriverRole* por el nombre del rol. Este comando requiere que tenga eksctl instalado en su dispositivo.

```
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
```

Información adicional

Para obtener más información sobre el complemento, consulte the section called “Amazon FSx para Lustre”.

Controlador CSI del Mountpoint para Amazon S3

El complemento del controlador de CSI de Mountpoint para Amazon S3 de Amazon EKS es un complemento de interfaz de almacenamiento de contenedores (CSI) de Kubernetes que proporciona almacenamiento de Amazon S3 para el clúster.

El nombre del complemento de Amazon EKS es `aws-mountpoint-s3-csi-driver`.

Permisos de IAM necesarios

Este complemento utiliza la capacidad de roles de IAM para cuentas de servicio de Amazon EKS. Para obtener más información, consulte the section called “Credenciales con IRSA”.

El rol de IAM que se cree requerirá una política que dé acceso a S3. Siga las recomendaciones de permisos de Mountpoint IAM al crear la política. Como alternativa, puede utilizar la política administrada de AWS AmazonS3FullAccess, pero esta política administrada concede más permisos de los necesarios para Mountpoint.

Cree un rol de IAM y adjunte la política a él con los siguientes comandos. Sustituya *my-cluster* por el nombre del clúster, *region-code* por el código de región de AWS correcto, *AmazonEKS_S3_CSI_DriverRole* por el nombre del rol y *AmazonEKS_S3_CSI_DriverRole_ARN* por el ARN del rol. Estos comandos requieren que tenga

eksctl instalado en su dispositivo. Para obtener instrucciones sobre cómo utilizar la consola de IAM o AWS CLI, consulte the section called “Paso 2: creación de un rol de IAM”.

```
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
```

Información adicional

Para obtener más información sobre el complemento, consulte the section called “Mountpoint para Amazon S3”.

Controlador de instantáneas CSI

El controlador de instantáneas de la interfaz de almacenamiento de contenedores (CSI) permite el uso de la funcionalidad de captura de instantáneas en controladores de CSI compatibles, como el controlador de CSI de Amazon EBS.

El nombre del complemento de Amazon EKS es `snapshot-controller`.

Permisos de IAM necesarios

Este complemento no requiere ningún permiso.

Información adicional

Para obtener más información sobre el complemento, consulte the section called “Controlador de instantáneas CSI”.

Gobernanza de tareas de Amazon SageMaker HyperPod

La gobernanza de tareas de SageMaker HyperPod es un sistema de administración sólido, diseñado para optimizar la asignación de recursos y garantizar un uso eficiente de los recursos de

computación en todos los equipos y proyectos dentro de los clústeres de Amazon EKS. Esto brinda a los administradores la capacidad de establecer:

- Niveles de prioridad para diversas tareas
- Asignación de computación para cada equipo
- El modo en que cada equipo presta y utiliza computación inactiva
- Si un equipo interrumpe o reasigna sus propias tareas

Además, la gobernanza de tareas de HyperPod brinda la capacidad de observabilidad de clústeres de Amazon EKS, que ofrece visibilidad en tiempo real de la capacidad de los clústeres. Esto incluye la disponibilidad y el uso de la computación, la asignación y utilización de los equipos y la información sobre el tiempo de ejecución y espera de las tareas, lo que le permite tomar decisiones fundamentadas y administrar los recursos de forma proactiva.

El nombre del complemento de Amazon EKS es `amazon-sagemaker-hyperpod-taskgovernance`.

Permisos de IAM necesarios

Este complemento no requiere ningún permiso.

Información adicional

Para obtener más información sobre el complemento, consulte SageMaker HyperPod task governance

Complemento de observabilidad para Amazon SageMaker HyperPod

El complemento de·observabilidad·de Amazon SageMaker HyperPod proporciona capacidades integrales de supervisión y observabilidad para los clústeres de HyperPod. Este complemento implementa y administra automáticamente los componentes de supervisión esenciales, incluidos el exportador de nodos, el exportador DCGM, kube-state-metrics y el exportador EFA. Recopila y reenvía métricas a una instancia de Amazon Managed Prometheus (AMP) designada por el cliente y expone un punto de conexión de OTLP para métricas personalizadas y la ingesta de eventos de los trabajos de entrenamiento del cliente.

El complemento se integra en el ecosistema más amplio de HyperPod mediante la recopilación de métricas de varios componentes, como el complemento HyperPod Task Governance, HyperPod

Training Operator, Kubeflow y KEDA. Todas las métricas recopiladas se centralizan en Amazon Managed Prometheus, lo que permite a los clientes obtener una vista de observabilidad unificada a través de los paneles de Amazon Managed Grafana. Esto proporciona una visibilidad integral del estado del clúster, la utilización de los recursos y el rendimiento de los trabajos de entrenamiento en todo el entorno de HyperPod.

El nombre del complemento de Amazon EKS es `amazon-sagemaker-hyperpod-observability`.

Permisos de IAM necesarios

Este complemento utiliza la capacidad de roles de IAM para cuentas de servicio de Amazon EKS. Para obtener más información, consulte the section called “Credenciales con IRSA”. Se requieren las siguientes políticas administradas:

- `AmazonPrometheusRemoteWriteAccess`: para escribir de forma remota las métricas del clúster en AMP
- `CloudWatchAgentServerPolicy`: para escribir de forma remota los registros del clúster en CloudWatch

Información adicional

Para obtener más información sobre el complemento y sus capacidades, consulte Observabilidad de SageMaker HyperPod.

Operador de entrenamiento de Amazon SageMaker HyperPod

El operador de entrenamiento de Amazon SageMaker HyperPod le ayuda a acelerar el desarrollo de modelos de IA generativa mediante la administración eficiente del entrenamiento distribuido en grandes clústeres de GPU. Presenta capacidades inteligentes de recuperación de errores, detección de tareas pendientes y administración de procesos que minimizan las interrupciones en el entrenamiento y reducen los costos. A diferencia de la infraestructura de entrenamiento tradicional, que requiere que el trabajo se reinicie por completo cuando se producen fallos, este operador implementa la recuperación del proceso quirúrgico para que sus trabajos de entrenamiento se desarrollen sin problemas.

El operador también trabaja con las funciones de supervisión y observabilidad del estado de HyperPod, lo que proporciona visibilidad en tiempo real de la ejecución del entrenamiento y supervisa automáticamente métricas críticas, como los picos de pérdidas y la degradación del

rendimiento. Puede definir las políticas de recuperación mediante sencillas configuraciones de YAML sin cambios en el código, lo que le permitirá responder rápidamente a estados de entrenamiento irrecuperables y recuperarse de ellos. Estas capacidades de supervisión y recuperación funcionan en conjunto para mantener un rendimiento de entrenamiento óptimo y, al mismo tiempo, minimizar los gastos operativos.

El nombre del complemento de Amazon EKS es `amazon-sagemaker-hyperpod-training-operator`.

Para obtener más información, consulte Uso del operador de entrenamiento HyperPod en la Guía para desarrolladores de Amazon SageMaker.

Permisos de IAM necesarios

Este complemento requiere permisos de IAM y utiliza Pod Identity de EKS.

AWS sugiere la política administrada `AmazonSageMakerHyperPodTrainingOperatorAccess`.

Para obtener más información, consulte Installing the training operator en la Guía para desarrolladores de Amazon SageMaker.

Información adicional

Para obtener más información sobre el complemento, consulte SageMaker HyperPod training operator.

Agente del monitor de flujo de red de AWS

El Agente del monitor de flujo de red de Amazon CloudWatch es una aplicación de Kubernetes que recopila estadísticas de conexión TCP de todos los nodos de un clúster y publica informes de flujo de red en las API de ingesta del Monitor de flujo de red de Amazon CloudWatch.

El nombre del complemento de Amazon EKS es `aws-network-flow-monitoring-agent`.

Permisos de IAM necesarios

Este complemento requiere permisos de IAM.

Debe asociar la política administrada de `CloudWatchNetworkFlowMonitorAgentPublishPolicy` al complemento.

Para obtener más información sobre la configuración de IAM requerida, consulte la Política de IAM en el repositorio de GitHub del Agente del Monitor de flujo de red de Amazon CloudWatch.

Para obtener más información sobre la política administrada, consulte CloudWatchNetworkFlowMonitorAgentPublishPolicy en la Guía del usuario de Amazon CloudWatch.

Información adicional

Para obtener más información sobre el complemento, consulte el repositorio de GitHub del Agente del Monitor de flujo de red de Amazon CloudWatch.

Agente de supervisión de nodos

El agente de supervisión de nodos del complemento de Amazon EKS puede detectar otros problemas de estado de los nodos. Estas señales de estado adicionales también se pueden aprovechar mediante la característica opcional de reparación automática de nodos para reemplazar automáticamente los nodos según sea necesario.

> **Note**
>
> No es necesario instalar este complemento en los clústeres del modo automático de Amazon EKS. Para obtener más información, consulte the section called “Consideraciones para el modo automático de Amazon EKS”.

El nombre del complemento de Amazon EKS es `eks-node-monitoring-agent`.

Permisos de IAM necesarios

Este complemento no requiere permisos adicionales.

Información adicional

Para obtener más información, consulte the section called “Estado de los nodos”.

AWS Distro para OpenTelemetry

El complemento de AWS Distro para OpenTelemetry de Amazon EKS es una distribución segura, lista para la producción y compatible con AWS del proyecto OpenTelemetry. Para obtener más información, consulte AWS Distro for OpenTelemetry en GitHub.

El nombre del complemento de Amazon EKS es `adot`.

Permisos de IAM necesarios

Este complemento solo requiere permisos de IAM si utiliza uno de los recursos personalizados configurados previamente que se pueden utilizar mediante una configuración avanzada.

Información adicional

Para obtener más información, consulte Getting Started with AWS Distro for OpenTelemetry using EKS Add-Ons en la documentación de AWS Distro para OpenTelemetry.

ADOT requiere que el complemento `cert-manager` esté implementado en el clúster como requisito previo; de lo contrario, este complemento no funcionará si se implementa directamente mediante la propiedad `cluster_addons` https://registry.terraform.io/modules/terraform-aws-modules/eks/aws/latest. Para obtener más información sobre los requisitos, consulte Requirements for Getting Started with AWS Distro for OpenTelemetry using EKS Add-Ons en la documentación de AWS Distro para OpenTelemetry.

Agente Amazon GuardDuty

El complemento de Amazon EKS del agente de Amazon GuardDuty recopila eventos de tiempo de ejecución (acceso a archivos, ejecución de procesos, conexiones de red) de los nodos del clúster de EKS para que Supervisión del tiempo de ejecución de GuardDuty los analice. El propio GuardDuty (no el agente) es el servicio de supervisión de seguridad que analiza y procesa orígenes de datos fundamentales, incluidos eventos de administración de AWS CloudTrail y registros de flujo de Amazon VPC, así como características, como los registros de auditoría de Kubernetes y la supervisión del tiempo de ejecución.

El nombre del complemento de Amazon EKS es `aws-guardduty-agent`.

Permisos de IAM necesarios

Este complemento no requiere ningún permiso.

Información adicional

Para obtener más información, consulte Runtime Monitoring for Amazon EKS clusters in Amazon GuardDuty.

- Para detectar posibles amenazas de seguridad en sus clústeres de Amazon EKS, habilite el monitoreo del tiempo de ejecución de Amazon GuardDuty e implemente el agente de seguridad de GuardDuty en sus clústeres de Amazon EKS.

Agente de Amazon CloudWatch Observability

El complemento del agente de observabilidad de Amazon CloudWatch de Amazon EKS es el servicio de supervisión y observabilidad que ofrece AWS. Este complemento instala el CloudWatch Agent y habilita tanto CloudWatch Application Signals como CloudWatch Container Insights con una observabilidad mejorada para Amazon EKS. Para obtener más información, consulte Agente de Amazon CloudWatch.

El nombre del complemento de Amazon EKS es `amazon-cloudwatch-observability`.

Permisos de IAM necesarios

Este complemento utiliza la capacidad de roles de IAM para cuentas de servicio de Amazon EKS. Para obtener más información, consulte the section called “Credenciales con IRSA”. Los permisos de las políticas administradas AWSXrayWriteOnlyAccess y CloudWatchAgentServerPolicy de AWS son obligatorios. Puede crear un rol de IAM, asociarle políticas administradas y anotar la cuenta de servicio de Kubernetes utilizada por el complemento con el siguiente comando. Reemplace *`my-cluster`* por el nombre del clúster y *`AmazonEKS_Observability_role`* por el nombre del rol. Este comando requiere que tenga eksctl instalado en su dispositivo. Si necesita usar una herramienta diferente para crear el rol, adjuntarle la política y anotar la cuenta de servicio de Kubernetes, consulte the section called “Asignación del rol de IAM”.

```
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
```

Información adicional

Para obtener más información, consulte Instale el agente de CloudWatch.

Conector de AWS Private CA para Kubernetes

El Conector de AWS Private CA para Kubernetes es un complemento para cert-manager que permite a los usuarios obtener certificados de AWS Private Certificate Authority (AWS Private CA).

- El nombre del complemento de Amazon EKS es `aws-privateca-connector-for-kubernetes`.
- El espacio de nombres del complemento es `aws-privateca-issuer`.

Este complemento requiere `cert-manager`. `cert-manager` está disponible en Amazon EKS como complemento de la comunidad. Para obtener más información acerca de este complemento, consulte the section called “Cert Manager”. Para obtener más información sobre la instalación de complementos, consulte the section called “Cómo crear un complemento”.

Permisos de IAM necesarios

Este complemento requiere permisos de IAM.

Utilice EKS Pod Identities para adjuntar la política de IAM `AWSPrivateCAConnectorForKubernetesPolicy` a la cuenta de servicio de Kubernetes `aws-privateca-issuer`. Para obtener más información, consulte the section called “Uso de Pod Identities”.

Para obtener información sobre los permisos necesarios, consulte AWSPrivateCAConnectorForKubernetesPolicy en la referencia de la política administrada de AWS.

Información adicional

Para obtener más información, consulte el repositorio de GitHub de AWS Private CA Issuer para Kubernetes.

Para obtener más información sobre la configuración del complemento, consulte values.yaml en el repositorio de GitHub `aws-privateca-issuer`. Confirme que la versión de values.yaml coincida con la versión del complemento instalado en el clúster.

Este complemento tolera la taint `CriticalAddonsOnly` utilizada por el NodePool `system` del modo automático de EKS. Para obtener más información, consulte the section called “Ejecución de complementos esenciales”.

Agente de Pod Identity de EKS

El complemento del agente de Pod Identity de EKS de Amazon EKS ofrece la posibilidad de administrar las credenciales de sus aplicaciones, de un modo similar a cómo los perfiles de instancias de EC2 proporcionan credenciales a instancias de EC2.

> **Note**
>
> No es necesario instalar este complemento en los clústeres del modo automático de Amazon EKS. El modo automático de Amazon EKS se integra con EKS Pod Identity. Para obtener más información, consulte the section called “Consideraciones para el modo automático de Amazon EKS”.

El nombre del complemento de Amazon EKS es `eks-pod-identity-agent`.

Permisos de IAM necesarios

El complemento del agente de Pod Identity no requiere un rol de IAM. Utiliza los permisos del rol de IAM del nodo de Amazon EKS para funcionar, pero no necesita un rol de IAM dedicado para el complemento.

Actualización de la información

Solo puede actualizar una versión secundaria a la vez. Por ejemplo, si su versión actual es la `1.28.x-eksbuild.y` y desea actualizarla a la `1.30.x-eksbuild.y`, primero debe actualizarla a la `1.29.x-eksbuild.y` y luego a la `1.30.x-eksbuild.y`. Para obtener más información acerca de la actualización de complementos, consulte the section called “Cómo actualizar un complemento”.

Exportador de métricas de red SR-IOV

El complemento de Amazon EKS para el exportador de métricas de red SR-IOV recopila y expone métricas sobre los dispositivos de red SR-IOV en formato Prometheus. Permite supervisar el rendimiento de la red SR-IOV en los nodos bare metal de EKS. El exportador funciona como un DaemonSet en nodos con interfaces de red compatibles con SR-IOV y exporta métricas que Prometheus puede recopilar.

> **Note**
>
> Este complemento requiere nodos con interfaces de red compatibles con SR-IOV.

Propiedad	Valor
Nombre del complemento	`sriov-network-metrics-exporter`

Propiedad	Valor
Espacio de nombres	`monitoring`
Documentación	Repositorio de GitHub del exportador de métricas de red SR-IOV
Nombre de la cuenta de servicio	Ninguno
Política de IAM administrada	Ninguno
Permisos de IAM personalizados	Ninguno

Proveedor de AWS para el controlador CSI del Almacén de secretos

El proveedor de AWS del controlador CSI del Almacén de secretos es un complemento que permite recuperar secretos de AWS Secrets Manager y parámetros del Almacén de parámetros de AWS Systems Manager y montarlos como archivos en pods de Kubernetes.

Permisos de IAM necesarios

El complemento no requiere permisos de IAM. Sin embargo, los pods de aplicaciones requerirán permisos de IAM para recuperar los secretos de AWS Secrets Manager y los parámetros del Almacén de parámetros de AWS Systems Manager. Tras instalar el complemento, el acceso debe configurarse mediante roles de IAM para cuentas de servicio (IRSA) o Pod Identity de EKS. Para usar IRSA, consulte la documentación de configuración de IRSA de Secrets Manager. Para utilizar Pod Identity de EKS, consulte la documentación de configuración de Pod Identity de Secrets Manager.

AWS sugiere la política administrada `AWSSecretsManagerClientReadOnlyAccess`.

Para obtener más información sobre los permisos necesarios, consulte `AWSSecretsManagerClientReadOnlyAccess` en la referencia de la política administrada de AWS.

Información adicional

Para obtener más información, consulte el repositorio de GitHub secrets-store-csi-driver-provider-aws.

Para obtener más información sobre el complemento, consulte la documentación de AWS Secrets Manager del complemento.

Espacios de Amazon SageMaker

El complemento Espacios de Amazon SageMaker permite ejecutar IDE y cuadernos en clústeres de HyperPod-EKS o EKS. Los administradores pueden utilizar la consola de EKS para instalar el complemento en el clúster y definir las configuraciones de espacio predeterminadas, como las imágenes, los recursos de computación, el almacenamiento local para la configuración del cuaderno (se adjunta almacenamiento adicional a sus espacios), los sistemas de archivos y los scripts de inicialización.

Los desarrolladores de IA pueden utilizar kubectl para crear, actualizar y eliminar espacios. Tienen la flexibilidad de usar las configuraciones predeterminadas proporcionadas por los administradores o personalizar las configuraciones. Los desarrolladores de IA pueden acceder a sus espacios en EKS o HyperPod-EKS mediante sus IDE de VS Code locales o su navegador web que aloja su IDE de JupyterLab o CodeEditor en un dominio de DNS personalizado configurado por sus administradores. También pueden usar la característica de reenvío de puertos de Kubernetes para acceder a los espacios de sus navegadores web.

El nombre del complemento de Amazon EKS es `amazon-sagemaker-spaces`.

Permisos de IAM necesarios

Este complemento requiere permisos de IAM. Para obtener más información sobre la configuración de IAM requerida, consulte Configuración de permisos de IAM en la Guía para desarrolladores de Amazon SageMaker.

Información adicional

Para obtener más información sobre el complemento y sus capacidades, consulte Cuadernos SageMaker AI para HyperPod en la Guía para desarrolladores de Amazon SageMaker.

Complementos de la comunidad

Puede utilizar las API de AWS para instalar complementos de la comunidad, como el Servidor de métricas de Kubernetes. Puede optar por instalar complementos de la comunidad como complementos de Amazon EKS para reducir la complejidad que supone mantener el software en varios clústeres.

Por ejemplo, puede utilizar la API de AWS, la CLI o la consola de administración para instalar complementos de la comunidad. Puede instalar un complemento de la comunidad durante la creación del clúster.

Los complementos de la comunidad se administran del mismo modo que los complementos existentes de Amazon EKS. Los complementos de la comunidad se diferencian de los complementos existentes en que tienen un ámbito de soporte único.

> **Note**
>
> El uso de complementos comunitarios queda a su entera discreción. Como parte del modelo de responsabilidad compartida con AWS, es fundamental comprender con claridad qué se instala en el clúster al incorporar estos complementos de terceros. También tiene la responsabilidad de asegurar que los complementos de la comunidad cumplan con los requisitos de seguridad del clúster. Para obtener más información, consulte the section called "Soporte del software implementado en EKS".

AWS no crea los complementos de la comunidad, sino que solo los valida para la compatibilidad de versiones. Por ejemplo, si instala un complemento comunitario en un clúster, AWS comprueba si es compatible con la versión de Kubernetes del clúster.

Es importante destacar que AWS no ofrece soporte completo para los complementos de la comunidad. AWS solo ofrece soporte para las operaciones del ciclo de vida realizadas mediante la API de AWS, como la instalación o eliminación de complementos.

Si necesita soporte para un complemento de la comunidad, utilice los recursos existentes en el proyecto. Por ejemplo, puede crear una incidencia de GitHub en el repositorio del proyecto.

Cómo determinar el tipo de complemento

Puede utilizar AWS CLI para determinar de qué tipo es un complemento de Amazon EKS.

Utilice el siguiente comando de la CLI para recuperar información sobre un complemento. Puede sustituir `metrics-server` por el nombre de cualquier complemento.

```
aws eks describe-addon-versions --addon-name metrics-server
```

Revise el resultado de la CLI correspondiente al campo `owner`.

```
{
    "addons": [
        {
            "addonName": "metrics-server",
            "type": "observability",
            "owner": "community",
            "addonVersions": [
```

Si el valor de `owner` es `community`, significa que se trata de un complemento de la comunidad. AWS solo proporciona soporte para instalar, actualizar y eliminar el complemento. Si tiene dudas sobre la funcionalidad y el funcionamiento del complemento en sí, utilice los recursos de la comunidad, como las incidencias de GitHub.

Cómo instalar o actualizar un complemento de la comunidad

Los complementos de la comunidad se instalan y actualizan del mismo modo que otros complementos de Amazon EKS.

- the section called “Cómo crear un complemento”
- the section called “Cómo actualizar un complemento”
- the section called “Cómo eliminar un complemento”

Complementos de la comunidad disponibles

Los siguientes complementos de la comunidad están disponibles en Amazon EKS.

- the section called “Servidor de métricas de Kubernetes”
- the section called “kube-state-metrics”
- the section called “Exportador de nodos de Prometheus”
- the section called “Cert Manager”
- the section called “DNS externo”
- the section called “Fluent Bit”

Servidor de métricas de Kubernetes

El Servidor de métricas de Kubernetes es una fuente escalable y eficiente de métricas de recursos de contenedores para las canalizaciones de escalado automático integradas en Kubernetes.

Recopila métricas de recursos de los Kubelets y las expone en el apiserver de Kubernetes a través de la API Metrics para que las utilicen el escalador automático horizontal de pods y el escalador automático vertical de pods.

Propiedad	Valor
Nombre del complemento	`metrics-server`
Espacio de nombres	`kube-system`
Documentación	Readme de GitHub
Nombre de la cuenta de servicio	Ninguno
Política de IAM administrada	Ninguno
Permisos de IAM personalizados	Ninguno

kube-state-metrics

Agente del complemento para generar y exponer métricas de clúster.

El estado de los objetos de Kubernetes en la API de Kubernetes se puede exponer como métricas. Un agente del complemento llamado kube-state-metrics puede conectarse al servidor de la API de Kubernetes y mostrar un punto de conexión HTTP con métricas generadas a partir del estado de los objetos individuales del clúster. Expone información diversa sobre el estado de los objetos, como etiquetas y anotaciones, las horas de inicio y finalización, el estado o la fase en la que se encuentra el objeto actualmente.

Propiedad	Valor
Nombre del complemento	`kube-state-metrics`
Espacio de nombres	`kube-state-metrics`
Documentación	Metrics for Kubernetes Object States en la documentación de Kubernetes
Nombre de la cuenta de servicio	Ninguno

Propiedad	Valor
Política de IAM administrada	Ninguno
Permisos de IAM personalizados	Ninguno

Exportador de nodos de Prometheus

Exportador de Prometheus para métricas de sistema operativo y hardware expuestas por kernels *NIX, escritas en Go con recopiladores de métricas conectables. El Exportador de nodos de Prometheus expone una amplia variedad de métricas relacionadas con el kernel y el hardware.

Propiedad	Valor
Nombre del complemento	`prometheus-node-exporter`
Espacio de nombres	`prometheus-node-exporter`
Documentación	Monitoring Linux host metrics with the Node Exporter en la documentación de Prometheus
Nombre de la cuenta de servicio	Ninguno
Política de IAM administrada	Ninguno
Permisos de IAM personalizados	Ninguno

Cert Manager

Cert Manager se puede utilizar para administrar la creación y renovación de certificados.

Propiedad	Valor
Nombre del complemento	`cert-manager`
Espacio de nombres	`cert-manager`
Documentación	Documentación de Cert Manager

Propiedad	Valor
Nombre de la cuenta de servicio	Ninguno
Política de IAM administrada	Ninguno
Permisos de IAM personalizados	Ninguno

DNS externo

El complemento de DNS externo para EKS permite administrar los registros de DNS de Route53 mediante recursos de Kubernetes.

Los permisos de DNS externo se pueden reducir a `route53:ChangeResourceRecordSets`, `route53:ListHostedZones` y `route53:ListResourceRecordSets` en las zonas alojadas que desee administrar.

Propiedad	Valor
Nombre del complemento	`external-dns`
Espacio de nombres	`external-dns`
Documentación	Readme de GitHub
Nombre de la cuenta de servicio	`external-dns`
Política de IAM administrada	`arn:aws:iam::aws:policy/AmazonRoute53FullAccess`
Permisos de IAM personalizados	Ninguno

Fluent Bit

Fluent Bit es un procesador y reenviador de registros ligero y de alto rendimiento. Le permite recopilar datos y registros de diferentes orígenes, unificarlos y enviarlos a varios destinos, incluidos Registros de Amazon CloudWatch, Amazon S3 y Amazon Kinesis Data Firehose. Fluent Bit está diseñado con el rendimiento y la eficiencia de los recursos en mente, por lo que es ideal para entornos de Kubernetes.

Este complemento no requiere permisos de IAM en la configuración predeterminada. Sin embargo, es posible que tenga que conceder permisos de IAM a este complemento si configura una ubicación de salida de AWS. Para obtener más información, consulte the section called “Uso de Pod Identities”.

Propiedad	Valor
Nombre del complemento	`fluent-bit`
Espacio de nombres	`fluent-bit`
Documentación	Documentación de Fluent Bit
Nombre de la cuenta de servicio	`fluent-bit`
Política de IAM administrada	Ninguno
Permisos de IAM personalizados	Ninguno

Ver atribuciones

Es posible descargar las atribuciones de código abierto y la información de licencias de los complementos de la comunidad.

1. Determine el nombre y la versión del complemento del que quiere descargar las atribuciones.
2. Actualiza el siguiente comando con el nombre y la versión:

```
curl -O https://amazon-eks-docs.s3.amazonaws.com/attributions/<add-on-name>/<add-on-version>/attributions.zip
```

   Por ejemplo:

```
curl -O https://amazon-eks-docs.s3.amazonaws.com/attributions/kube-state-metrics/v2.14.0-eksbuild.1/attributions.zip
```

3. Utilice el comando para descargar el archivo.

Utilice este archivo comprimido para ver información sobre las atribuciones de la licencia.

Complementos de AWS Marketplace

Además de la lista anterior de complementos de Amazon EKS, también puede agregar una amplia selección de complementos de Amazon EKS de software operativo de proveedores de software independientes. Elija un complemento para obtener más información sobre él y sus requisitos de instalación.

Accuknox

El nombre del complemento es `accuknox_kubearmor` y el espacio de nombres es `kubearmor`. Accuknox publica el complemento.

Para obtener más información sobre el complemento, consulte Getting Started with KubeArmor en la documentación de KubeArmor.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Akuity

El nombre del complemento es `akuity_agent` y el espacio de nombres es `akuity`. Akuity publica el complemento.

Para obtener información sobre cómo funciona el complemento, consulte Installing the Akuity Agent on Amazon EKS with the Akuity EKS add-on en la documentación de Akuity Platform.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Calyptia

El nombre del complemento es `calyptia_fluent-bit` y el espacio de nombres es `calytia-fluentbit`. Calyptia publica el complemento.

Para obtener más información sobre el complemento, consulte Getting Started with Calyptia Core Agent en el sitio web de documentación de Calyptia.

Nombre de la cuenta de servicio

El nombre de la cuenta de servicio es `clyptia-fluentbit`.

Política de IAM administrada de AWS

Este complemento utiliza la política administrada por `AWSMarketplaceMeteringRegisterUsage`. Para obtener más información, consulte AWSMarketplaceMeteringRegisterUsage en la guía de referencia de políticas administradas por AWS.

Comando para crear el rol de IAM necesario

El siguiente comando requiere que tenga un proveedor de OpenID Connect (OIDC) de IAM para su clúster. Para determinar si ya tiene uno o si debe crearlo, consulte the section called “Proveedor de OIDC de IAM”. Reemplace *`my-cluster`* con el nombre del clúster y reemplace *`my-calyptia-role`* con el nombre que desea para el rol. Este comando requiere que tenga eksctl instalado en su dispositivo. Si necesita usar una herramienta diferente para crear el rol y anotar la cuenta de servicio de Kubernetes, consulte the section called “Asignación del rol de IAM”.

```
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 Observability Collector

El nombre del complemento es `cisco_cisco-cloud-observability-collectors` y el espacio de nombres es `appdynamics`. Cisco publica el complemento.

Para obtener más información sobre el complemento, consulte Use the Cisco Cloud Observability AWS Marketplace Add-Ons en la documentación de Cisco AppDynamics.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Cisco Observability Operator

El nombre del complemento es `cisco_cisco-cloud-observability-operators` y el espacio de nombres es `appdynamics`. Cisco publica el complemento.

Para obtener más información sobre el complemento, consulte Use the Cisco Cloud Observability AWS Marketplace Add-Ons en la documentación de Cisco AppDynamics.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

CLOUDSOFT

El nombre del complemento es `cloudsoft_cloudsoft-amp` y el espacio de nombres es `cloudsoft-amp`. CLOUDSOFT publica el complemento.

Para obtener más información sobre el complemento, consulte Amazon EKS ADDON en la documentación de CLOUDSOFT.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Cribl

El nombre del complemento es `cribl_cribledge` y el espacio de nombres es `cribledge`. Cribl publica el complemento.

Para obtener más información sobre el complemento, consulte Installing the Cribl Amazon EKS Add-on for Edge en la documentación de Cribl.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Dynatrace

El nombre del complemento es `dynatrace_dynatrace-operator` y el espacio de nombres es `dynatrace`. Dynatrace publica el complemento.

Para obtener más información sobre el complemento, consulte Kubernetes monitoring en la documentación de Dynatrace.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Datree

El nombre del complemento es `datree_engine-pro` y el espacio de nombres es `datree`. Datree publica el complemento.

Para obtener más información sobre el complemento, consulte Amazon EKS-integration en la documentación de Datree.

Nombre de la cuenta de servicio

El nombre de la cuenta de servicio es datree-webhook-server-awsmp.

Política de IAM administrada de AWS

La política administrada es AWSLicenseManagerConsumptionPolicy. Para obtener más información, consulte AWSLicenseManagerConsumptionPolicy en la guía de referencia de políticas administradas por AWS.

Comando para crear el rol de IAM necesario

El siguiente comando requiere que tenga un proveedor de OpenID Connect (OIDC) de IAM para su clúster. Para determinar si ya tiene uno o si debe crearlo, consulte the section called “Proveedor de OIDC de IAM”. Reemplace *`my-cluster`* con el nombre del clúster y reemplace *`my-datree-role`* con el nombre que desea para el rol. Este comando requiere que tenga eksctl instalado en su dispositivo. Si necesita usar una herramienta diferente para crear el rol y anotar la cuenta de servicio de Kubernetes, consulte the section called “Asignación del rol de IAM”.

```
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
```

Permisos personalizados

No se utilizan permisos personalizados con este complemento.

Datadog

El nombre del complemento es `datadog_operator` y el espacio de nombres es `datadog-agent`. Datadog publica el complemento.

Para obtener información sobre el complemento, consulte Installing the Datadog Agent on Amazon EKS with the Datadog Operator Add-on en la documentación de Datadog.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Groundcover

El nombre del complemento es `groundcover_agent` y el espacio de nombres es `groundcover`. Groundcover publica el complemento.

Para obtener más información sobre el complemento, consulte Installing the groundcover Amazon EKS Add-on en la documentación de groundcover.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

IBM Instana

El nombre del complemento es `instana-agent` y el espacio de nombres es `instana-agent`. IBM publica el complemento.

Para obtener información sobre el complemento, consulte Implement observability for Amazon EKS workloads using the Instana Amazon EKS Add-on y Monitor and optimize Amazon EKS costs with IBM Instana and Kubecost en el blog de AWS.

Instana Observability (Instana) ofrece un complemento de Amazon EKS que implementa agentes de Instana en los clústeres de Amazon EKS. Los clientes pueden usar este complemento para recopilar y analizar datos de rendimiento en tiempo real a fin de obtener información sobre sus aplicaciones incluidas en contenedores. El complemento Instana de Amazon EKS proporciona visibilidad en todos sus entornos de Kubernetes. Una vez implementado, el agente de Instana detecta automáticamente los componentes de los clústeres de Amazon EKS, incluidos los nodos, los espacios de nombres, las implementaciones, los servicios y los pods.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Grafana Labs

El nombre del complemento es `grafana-labs_kubernetes-monitoring` y el espacio de nombres es `monitoring`. Grafana Labs publica el complemento.

Para obtener más información sobre el complemento, consulte Configure Kubernetes Monitoring as an Add-on with Amazon EKS en la documentación de Grafana Labs.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Guance

- Publicador: GUANCE
- Nombre: `guance_datakit`
- Espacio de nombres: `datakit`
- Nombre de la cuenta de servicio: no se usa una cuenta de servicio con este complemento.
- Política de IAM administrada de AWS: no se utiliza una política administrada con este complemento.
- Permisos de IAM personalizados: los permisos personalizados no se utilizan con este complemento.
- Instrucciones de configuración y uso: consulte Using Amazon EKS add-on en la documentación de Guance.

HA Proxy

El nombre es `haproxy-technologies_kubernetes-ingress-ee` y el espacio de nombres es `haproxy-controller`. HA Proxy publica el complemento.

Para obtener más información sobre el complemento, consulte Amazon EKS-integration en la documentación de Datree.

Nombre de la cuenta de servicio

El nombre de la cuenta de servicio es `customer defined`.

Política de IAM administrada de AWS

La política administrada es AWSLicenseManagerConsumptionPolicy. Para obtener más información, consulte AWSLicenseManagerConsumptionPolicy en la guía de referencia de políticas administradas por AWS.

Comando para crear el rol de IAM necesario

El siguiente comando requiere que tenga un proveedor de OpenID Connect (OIDC) de IAM para su clúster. Para determinar si ya tiene uno o si debe crearlo, consulte the section called “Proveedor de OIDC de IAM”. Reemplace *`my-cluster`* con el nombre del clúster y reemplace *`my-haproxy-role`* con el nombre que desea para el rol. Este comando requiere que tenga eksctl instalado en su

dispositivo. Si necesita usar una herramienta diferente para crear el rol y anotar la cuenta de servicio de Kubernetes, consulte the section called “Asignación del rol de IAM”.

```
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
```

Permisos personalizados

No se utilizan permisos personalizados con este complemento.

Kpow

El nombre del complemento es `factorhouse_kpow` y el espacio de nombres es `factorhouse`. Factorhouse publica el complemento.

Para obtener más información sobre el complemento, consulte AWS Marketplace LM en la documentación de Kpow.

Nombre de la cuenta de servicio

El nombre de la cuenta de servicio es `kpow`.

Política de IAM administrada de AWS

La política administrada es AWSLicenseManagerConsumptionPolicy. Para obtener más información, consulte AWSLicenseManagerConsumptionPolicy en la guía de referencia de políticas administradas por AWS.

Comando para crear el rol de IAM necesario

El siguiente comando requiere que tenga un proveedor de OpenID Connect (OIDC) de IAM para su clúster. Para determinar si ya tiene uno o si debe crearlo, consulte the section called “Proveedor de OIDC de IAM”. Reemplace *`my-cluster`* con el nombre del clúster y reemplace *`my-role-name`* con el nombre que desea para el rol. Este comando requiere que tenga eksctl instalado en su dispositivo. Si necesita usar una herramienta diferente para crear el rol y anotar la cuenta de servicio de Kubernetes, consulte the section called “Asignación del rol de IAM”.

```
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
```

Permisos personalizados

No se utilizan permisos personalizados con este complemento.

Kubecost

El nombre del complemento es `kubecost_kubecost` y el espacio de nombres es `kubecost`. Kubecost publica el complemento.

Para obtener más información sobre el complemento, consulte AWS Cloud Billing Integration en la documentación de Kubecost.

Debe haber instalado antes el almacenamiento de volúmenes de Kubernetes con Amazon EBS en el clúster; de lo contrario, recibirá un error.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Kasten

El nombre del complemento es `kasten_k10` y el espacio de nombres es `kasten-io`. Kasten by Veeam publica el complemento.

Para obtener más información sobre el complemento, consulte Installing K10 on AWS using Amazon EKS Add-on en la documentación de Kasten.

Debe haber instalado el controlador de CSI de Amazon en el clúster con una `StorageClass` predeterminada.

Nombre de la cuenta de servicio

El nombre de la cuenta de servicio es `k10-k10`.

Política de IAM administrada de AWS

La política administrada es AWSLicenseManagerConsumptionPolicy. Para obtener más información, consulte AWSLicenseManagerConsumptionPolicy en la guía de referencia de políticas administradas por AWS.

Comando para crear el rol de IAM necesario

El siguiente comando requiere que tenga un proveedor de OpenID Connect (OIDC) de IAM para su clúster. Para determinar si ya tiene uno o si debe crearlo, consulte the section called “Proveedor de OIDC de IAM”. Reemplace *my-cluster* con el nombre del clúster y reemplace *my-kasten-role* con el nombre que desea para el rol. Este comando requiere que tenga eksctl instalado en su dispositivo. Si necesita usar una herramienta diferente para crear el rol y anotar la cuenta de servicio de Kubernetes, consulte the section called “Asignación del rol de IAM”.

```
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
```

Permisos personalizados

No se utilizan permisos personalizados con este complemento.

Kong

El nombre del complemento es `kong_konnect-ri` y el espacio de nombres es `kong`. Kong publica el complemento.

Para obtener más información sobre el complemento, consulte Installing the Kong Gateway EKS Add-on en la documentación de Kong.

Debe haber instalado antes el almacenamiento de volúmenes de Kubernetes con Amazon EBS en el clúster; de lo contrario, recibirá un error.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

LeakSignal

El nombre del complemento es `leaksignal_leakagent` y el espacio de nombres es `leakagent`. LeakSignal publica el complemento.

Para obtener más información sobre el complemento, consulte https://www.leaksignal.com/docs/LeakAgent/Deployment/ AWS%20EKS%20Addon/[Install the LeakAgent add-on] en la documentación de LeakSignal

Debe haber instalado antes el almacenamiento de volúmenes de Kubernetes con Amazon EBS en el clúster; de lo contrario, recibirá un error.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

NetApp

El nombre del complemento es `netapp_trident-operator` y el espacio de nombres es `trident`. NetApp publica el complemento.

Para obtener más información sobre el complemento, consulte Configuración del complemento de EKS de Trident en la documentación de NetApp.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

New Relic

El nombre del complemento es `new-relic_kubernetes-operator` y el espacio de nombres es `newrelic`. New Relic publica el complemento.

Para obtener más información sobre el complemento, consulte Installing the New Relic Add-on for EKS en la documentación de New Relic.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Rafay

El nombre del complemento es `rafay-systems_rafay-operator` y el espacio de nombres es `rafay-system`. Rafay publica el complemento.

Para obtener más información sobre el complemento, consulte Installing the Rafay Amazon EKS Add-on en la documentación de Rafay.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Rad Security

- Publicador: RAD SECURITY
- Nombre: `rad-security_rad-security`
- Espacio de nombres: `ksoc`
- Nombre de la cuenta de servicio: no se usa una cuenta de servicio con este complemento.
- Política de IAM administrada de AWS: no se utiliza una política administrada con este complemento.
- Permisos de IAM personalizados: los permisos personalizados no se utilizan con este complemento.
- Instrucciones de configuración y uso: consulte Installing Rad Through The AWS Marketplace en la documentación de Rad Security.

SolarWinds

- Publicador: SOLARWINDS
- Nombre: `solarwinds_swo-k8s-collector-addon`
- Espacio de nombres: `solarwinds`
- Nombre de la cuenta de servicio: no se usa una cuenta de servicio con este complemento.
- Política de IAM administrada de AWS: no se utiliza una política administrada con este complemento.
- Permisos de IAM personalizados: los permisos personalizados no se utilizan con este complemento.
- Instrucciones de configuración y uso: consulte Monitor an Amazon EKS cluster en la documentación de SolarWinds.

Solo

El nombre del complemento es `solo-io_istio-distro` y el espacio de nombres es `istio-system`. Solo publica el complemento.

Para obtener más información sobre el complemento, consulte Installing Istio en la documentación de Solo.io.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Snyk

- Publicador: SNYK
- Nombre: `snyk_runtime-sensor`
- Espacio de nombres: `snyk_runtime-sensor`
- Nombre de la cuenta de servicio: no se usa una cuenta de servicio con este complemento.
- Política de IAM administrada de AWS: no se utiliza una política administrada con este complemento.
- Permisos de IAM personalizados: los permisos personalizados no se utilizan con este complemento.
- Instrucciones de configuración y uso: consulte Snyk runtime sensor en los documentos para usuarios de Snyk.

Stormforge

El nombre del complemento es `stormforge_optimize-Live` y el espacio de nombres es `stormforge-system`. Stormforge publica el complemento.

Para obtener más información sobre el complemento, consulte Installing the StormForge Agent en la documentación de StormForge.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

SUSE

- Publicador: SUSE
- Nombre: `suse_observability-agent`
- Espacio de nombres: `suse-observability`
- Nombre de la cuenta de servicio: no se usa una cuenta de servicio con este complemento.
- Política de IAM administrada de AWS: no se utiliza una política administrada con este complemento.
- Permisos de IAM personalizados: los permisos personalizados no se utilizan con este complemento.
- Instrucciones de configuración y uso: consulte Inicio rápido en la documentación de SUSE.

Splunk

El nombre del complemento es `splunk_splunk-otel-collector-chart` y el espacio de nombres es `splunk-monitoring`. Splunk publica el complemento.

Para obtener más información sobre el complemento, consulte Install the Splunk add-on for Amazon EKS en la documentación de Splunk.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Teleport

El nombre del complemento es `teleport_teleport` y el espacio de nombres es `teleport`. Teleport publica el complemento.

Para obtener más información sobre el complemento, consulte How Teleport Works en la documentación de Teleport.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Tetrate

El nombre del complemento es `tetrate-io_istio-distro` y el espacio de nombres es `istio-system`. Tetrate publica el complemento.

Para obtener más información sobre el complemento, consulte el sitio web de Tetrate Istio Distro.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Upbound Universal Crossplane

El nombre del complemento es `upbound_universal-crossplane` y el espacio de nombres es `upbound-system`. Upbound publica el complemento.

Para obtener más información sobre el complemento, consulte Upbound Universal Crossplane (UXP) en la documentación de Upbound.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Upwind

El nombre del complemento es `upwind` y el espacio de nombres es `upwind`. Upwind publica el complemento.

Para obtener más información sobre el complemento, consulte la documentación de Upwind.

Nombre de la cuenta de servicio

No se usa una cuenta de servicio con este complemento.

Política de IAM administrada de AWS

No se utiliza una política administrada con este complemento.

Permisos de IAM personalizados

No se utilizan permisos personalizados con este complemento.

Cómo crear un complemento de Amazon EKS

Los complementos de Amazon EKS son software de complementos para clústeres de Amazon EKS. Todos los complementos de Amazon EKS:

- Incluyen las revisiones de seguridad y correcciones de errores más recientes.
- Están validados por AWS para poder usarse con Amazon EKS.
- Reducen la cantidad de trabajo necesaria para administrar el software de complementos.

Puede crear un complemento de Amazon EKS mediante `eksctl`, la Consola de administración de AWS o la CLI de AWS. Si el complemento requiere un rol de IAM, consulte los detalles del complemento específico en Complementos de Amazon EKS para obtener más información sobre cómo crear el rol.

Requisitos previos

Siga estos pasos antes de crear un complemento:

- El clúster debe existir antes de crear un complemento para él. Para obtener más información, consulte the section called “Creación de un clúster ”.
- Compruebe si el complemento requiere un rol de IAM. Para obtener más información, consulte the section called “Comprobación de la compatibilidad”.
- Compruebe que la versión del complemento de Amazon EKS es compatible con el clúster. Para obtener más información, consulte the section called “Comprobación de la compatibilidad”.
- Compruebe que la versión 0.190.0 o posterior de la herramienta de línea de comandos de `eksctl` esté instalada en su equipo o AWS CloudShell. Para obtener más información, consulte Instalación en el sitio web de `eksctl`.

Procedimiento

Puede crear un complemento de Amazon EKS mediante `eksctl`, la Consola de administración de AWS o la CLI de AWS. Si el complemento requiere un rol de IAM, consulte los detalles del complemento específico en Complementos de Amazon EKS disponibles desde AWS para obtener información sobre cómo crear el rol.

Creación de complemento (eksctl)

1. Visualización de los nombres de los complementos disponibles para una versión de clúster. Reemplace *1.33* por la versión del clúster.

```
eksctl utils describe-addon-versions --kubernetes-version 1.33 | grep AddonName
```

   Un ejemplo de salida sería el siguiente.

```
"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. Visualización de las versiones disponibles para el complemento que le gustaría crear. Reemplace *1.33* por la versión del clúster. Reemplace *name-of-addon* por el nombre del complemento para el que desea ver las versiones. El nombre debe ser uno de los nombres que obtuvo en el paso anterior.

```
eksctl utils describe-addon-versions --kubernetes-version 1.33 --name name-of-addon |
 grep AddonVersion
```

El siguiente resultado es un ejemplo de lo que se devuelve para el complemento denominado `vpc-cni`. Puede ver que el complemento tiene varias versiones disponibles.

```
"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. Determine si el complemento que desea crear es uno de Amazon EKS o de AWS Marketplace. AWS Marketplace tiene complementos de terceros que requieren que complete pasos adicionales para crear el complemento.

```
eksctl utils describe-addon-versions --kubernetes-version 1.33 --name name-of-
addon | grep ProductUrl
```

Si no se devuelve ningún resultado, el complemento es de Amazon EKS. Si se devuelve algún resultado, significa que el complemento es un complemento de AWS Marketplace. El siguiente resultado corresponde a un complemento denominado `teleport_teleport`.

```
"ProductUrl": "https://aws.amazon.com/marketplace/pp?sku=3bda70bb-566f-4976-806c-
f96faef18b26"
```

Puede obtener más información sobre el complemento en AWS Marketplace con la URL devuelta. Si el complemento requiere una suscripción, puede suscribirse al complemento a través de AWS Marketplace. Si va a crear un complemento desde AWS Marketplace, la entidad principal de IAM que utilice para crear el complemento debe tener permiso para crear el rol vinculado al servicio AWSServiceRoleForAWSLicenseManagerRole. Para obtener información

sobre cómo asignar los permisos a una entidad principal de IAM, consulte Adición y eliminación de permisos de identidad de IAM en la Guía del usuario de IAM.

3. Creación de un complemento de Amazon EKS. Copie el comando y sustituya *user-data* de la siguiente manera:
   - Reemplace *my-cluster* por el nombre de su clúster.
   - Reemplace *name-of-addon* por el nombre del complemento que desea crear.
   - Si quiere una versión del complemento anterior a la más reciente, reemplace *más recientes* por el número de versión que quiera utilizar, tomando como referencia la salida del paso anterior.
   - Si el complemento usa un rol de cuenta de servicio, reemplace *111122223333* por el ID de la cuenta y *role-name* por el nombre del rol. Para obtener instrucciones sobre cómo crear un rol para su cuenta de servicio, consulte la documentación del complemento que está creando. Para obtener una lista de los complementos, consulte the section called " Complementos de AWS". Para especificar un rol de cuenta de servicio, es necesario disponer de un proveedor de OpenID Connect (OIDC) de IAM para el clúster. Para determinar si ya tiene uno para su clúster o si debe crearlo, consulte the section called "Proveedor de OIDC de IAM".

     Si el complemento no usa un rol de cuenta de servicio, elimine `--service-account-role-arnarn:aws:iam::111122223333:role/role-name`.
   - Este comando de ejemplo sobrescribe la configuración de cualquier versión autoadministrada existente del complemento, si la hay. Si no quiere sobrescribir la configuración de un complemento autoadministrado existente, elimine la opción *--force*. Si lo hace, y el complemento de Amazon EKS necesita sobrescribir la configuración existente de un complemento autoadministrado, se produce un error en la creación del complemento de Amazon EKS y recibe un mensaje para ayudarlo a resolver el conflicto. Antes de especificar esta opción, asegúrese de que el complemento de Amazon EKS no administra la configuración que necesita administrar, ya que dicha configuración se sobrescribe con esta opción.

```
eksctl create addon --cluster my-cluster --name name-of-addon --version latest \
    --service-account-role-arn arn:aws:iam::111122223333:role/role-name --force
```

   Puede ver una lista completa de las opciones disponibles para el comando.

```
eksctl create addon --help
```

Para obtener más información acerca de otras opciones, consulte Addons (Complementos) en la documentación de `eksctl`.

Creación de complemento (consola de AWS)

1. Abra la consola de Amazon EKS.
2. En el panel de navegación izquierdo, elija Clusters (Clústeres).
3. Elija el nombre del clúster para el que desea crear el complemento.
4. Elija la pestaña Complementos.
5. Escoja Obtener más complementos.
6. En la página Seleccionar complementos, elija los complementos que desea agregar al clúster. Puede agregar tantos complementos de Amazon EKS y complementos de AWS Marketplace como necesite.

   En el caso de los complementos de AWS Marketplace, la entidad principal de IAM que se utilice para crear un complemento debe tener permisos para leer los derechos del complemento desde AWS LicenseManager. AWS LicenseManager requiere el rol vinculado al servicio (SLR) AWSServiceRoleForAWSLicenseManagerRole que permite que los recursos de AWS administren las licencias en su nombre. El SLR es obligatorio una sola vez, por cuenta, y no tendrá que crear SLR independientes para cada complemento ni para cada clúster. Para obtener información sobre cómo asignar los permisos a una entidad principal de IAM, consulte Adición y eliminación de permisos de identidad de IAM en la Guía del usuario de IAM.

   Si los complementos de AWS Marketplace que desea instalar no aparecen en la lista, puede hacer clic en la numeración de páginas para ver resultados de páginas adicionales o buscar en el cuadro de búsqueda. En Opciones de filtrado, también puede buscar por categoría, proveedor o modelo de precios y, a continuación, seleccionar los complementos en los resultados de la búsqueda. Una vez que haya seleccionado los complementos que desee instalar, seleccione Siguiente.
7. En la página Configurar las opciones de complementos seleccionados, haga lo siguiente:
   a. Seleccione Ver opciones de suscripción para abrir el formulario Opciones de suscripción. Revise las secciones Detalles de precios y Legal y, a continuación, pulse el botón Suscribirse para continuar.
   b. En Versión, seleccione la versión que desee instalar. Recomendamos la versión marcada como la más reciente, a menos que el complemento individual que está creando recomiende una versión diferente. Para determinar si un complemento tiene una versión recomendada,

consulte la documentación del complemento que está creando. Para obtener una lista de los complementos, consulte the section called “ Complementos de AWS”.

c. Tiene dos opciones para configurar roles para complementos: rol de IAM de EKS Pod Identities y roles de IAM para cuentas de servicio (IRSA). Siga el paso correspondiente a la opción que prefiera. Si todos los complementos que seleccionó tienen la opción Requiere suscripción en Estado, elija Siguiente. No podrá continuar con la configuración de esos complementos hasta que se haya suscrito a estos después de crear el clúster. Para los complementos que no tienen Requiere suscripción en Estado, haga lo siguiente:

   i. En el caso del rol de IAM de Pod Identity para la cuenta de servicio, puede utilizar un rol de IAM de EKS Pod Identity existente o crear uno mediante el botón Crear rol recomendado. Este campo solo proporcionará opciones con la política de confianza adecuada. Si no hay ningún rol que seleccionar, significa que no existe ningún rol con la misma política de confianza. Para configurar un rol de IAM de EKS Pod Identity para las cuentas de servicio del complemento seleccionado, seleccione Crear rol recomendado. El asistente de creación de roles se abre en una ventana independiente. El asistente completará automáticamente la información del rol de la siguiente manera. Para cada complemento en el que desee crear el rol de IAM de EKS Pod Identity, complete los pasos del asistente de IAM como se indica a continuación.

      - En el paso Seleccionar entidad de confianza, la opción de servicio de AWS para EKS y el caso de uso para EKS: Pod Identity están preseleccionados, y la política de confianza adecuada se completará automáticamente para el complemento. Por ejemplo, el rol se creará con la política de confianza adecuada que contenga la entidad principal de IAM pods.eks.amazonaws.com, tal y como se detalla en the section called “Ventajas de las Pod Identities de EKS”. Elija Siguiente.
      - En el paso Agregar permisos, se preselecciona la política administrada adecuada para la política de roles para el complemento. Por ejemplo, para el complemento CNI de Amazon VPC, el rol se creará con la política administrada `AmazonEKS_CNI_Policy`, como se detalla en the section called “Complemento CNI de Amazon VPC para Kubernetes”. Elija Siguiente.
      - En el paso Nombrar, revisar y crear, en Nombre de rol, el nombre de rol predeterminado se completa automáticamente para el complemento. Por ejemplo, para el complemento de CNI de Amazon VPC, el rol se creará con el nombre AmazonEKSPodIdentityAmazonVPCCNIRole. En Descripción, la descripción predeterminada se completa automáticamente con la descripción adecuada para el complemento. Por ejemplo, para el complemento CNI de Amazon VPC, el rol se creará

con la descripción Permite que los pods que se ejecutan en el clúster de Amazon EKS accedan a los recursos de AWS. En Política de confianza, podrá ver la política de confianza completada para el complemento. Seleccione Crear rol.

NOTA: Al conservar el nombre predeterminado del rol, EKS puede preseleccionar el rol para complementos en nuevos clústeres o al agregar complementos a clústeres existentes. Puede sobrescribir este nombre y el rol estará disponible para el complemento en todos los clústeres, pero se tendrá que seleccionar manualmente en el menú desplegable.

ii. Para los complementos que no muestran Requiere suscripción en el Estado y en los que desea configurar roles mediante IRSA, consulte la documentación del complemento que va a crear para definir una política de IAM y asociarla a un rol. Para obtener una lista de los complementos, consulte the section called “ Complementos de AWS”. Para seleccionar un rol de IAM es necesario que tenga un proveedor de OpenID Connect (OIDC) de IAM para el clúster. Para determinar si ya tiene uno para su clúster o si debe crearlo, consulte the section called “Proveedor de OIDC de IAM”.

iii. Seleccione Valores de configuración opcionales.

iv. Si el complemento requiere configuración, introdúzcala en el cuadro Configuration values (Valores de configuración). Para determinar si el complemento requiere información de configuración, consulte la documentación del complemento que está creando. Para obtener una lista de los complementos, consulte the section called “ Complementos de AWS”.

v. En Método de resolución de conflictos, elija una de las opciones disponibles. Si elije Anular en Método de resolución de conflictos, una o varias de las configuraciones del complemento existente pueden sobrescribirse con la configuración del complemento de Amazon EKS. Si no habilita esta opción y hay un conflicto con la configuración existente, la operación fallará. Puede utilizar el mensaje de error resultante para solucionar el conflicto. Antes de elegir esta opción, asegúrese de que el complemento de Amazon EKS no administra las configuraciones que debe autoadministrar.

vi. Si desea instalar el complemento en un espacio de nombres específico, ingréselo en el campo Espacio de nombres. Para los complementos de AWS y de la comunidad, puede definir un espacio de nombres de Kubernetes personalizado en el que instalar el complemento. Para obtener más información, consulte the section called “Espacio de nombres personalizado para complementos”.

vii. Elija Siguiente.

8. En la página Revisar y añadir, elija Crear. Una vez finalizada la instalación del complemento, verá los complementos instalados.
9. Si alguno de los complementos que ha instalado requiere una suscripción, siga estos pasos:
   a. Selecciona el botón Subscribe (Suscribirse) en la esquina inferior derecha del complemento. Se lo redirigirá a la página del complemento en AWS Marketplace. Lea la información sobre el complemento, como Product Overview (Descripción general del producto) y Pricing Information (Información sobre precios).
   b. Seleccione el botón Continue to Subscribe (Continuar con la suscripción) en la parte superior derecha de la página del complemento.
   c. Lea la sección Terms and Conditions (Condiciones generales). Si está de acuerdo con ellas, seleccione Accept Terms (Aceptar las condiciones). La suscripción puede tardar varios minutos en procesarse. Mientras se procesa la suscripción, el botón Return to Amazon EKS Console (Volver a la consola de Amazon EKS) aparece en gris.
   d. Una vez que la suscripción haya terminado de procesarse, el botón Return to Amazon EKS Console (Volver a la consola de Amazon EKS) ya no aparecerá en gris. Elija el botón para volver a la pestaña Add-ons (Complementos) de la consola de Amazon EKS de su clúster.
   e. Para el complemento al que se suscribió, seleccione Remove and reinstall (Eliminar y volver a instalar) y, a continuación, elija Reinstalar el complemento (Reinstall add-on). La instalación del complemento puede tardar varios minutos. Cuando finalice la instalación, podrá configurar el complemento.

Creación de complemento (AWS CLI)

1. Necesita una versión `2.12.3` o posterior, o bien una versión `1.27.160` o posterior de la Interfaz de la línea de comandos de AWS (AWS CLI) instalada y configurada en su dispositivo o AWS CloudShell. Para comprobar su versión actual, utilice `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente. Para actualizarla, consulte Instalación de la CLI de AWS en su directorio principal en la Guía del usuario de AWS CloudShell.

2. Determine qué complementos están disponibles. Puede ver todos los complementos disponibles, su tipo y su editor. También puede ver la URL de los complementos que están disponibles a través de AWS Marketplace. Reemplace *1.33* por la versión del clúster.

```
aws eks describe-addon-versions --kubernetes-version 1.33 \
    --query 'addons[].{MarketplaceProductUrl: marketplaceInformation.productUrl,
 Name: addonName, Owner: owner Publisher: publisher, Type: type}' --output table
```

Un ejemplo de salida sería el siguiente.

```
-------------------------------------------------------------------------------------------------
|
 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         |
```

```
|  [...]                                                        |  [...]
          |  [...]          |  [...]      |  [...]              |
+-------------------------------------------------------------
+------------------------------+-----------------+-------------
+---------------------+
```

El resultado puede ser diferente. En el resultado de este ejemplo, hay tres complementos diferentes disponibles de tipo `networking` y cinco complementos con un editor de tipo `eks`. Es posible que los complementos con `aws-marketplace`, que aparecen en la columna `Owner`, requieran una suscripción antes de poder instalarlos. Puede visitar la URL para obtener más información sobre el complemento y suscribirse a él.

3. Puede ver qué versiones están disponibles para cada complemento. Reemplace *`1.33`* por la versión de su clúster y reemplace *`vpc-cni`* por el nombre de un complemento devuelto en el paso anterior.

```
aws eks describe-addon-versions --kubernetes-version 1.33 --addon-name vpc-cni \
    --query 'addons[].addonVersions[].{Version: addonVersion, Defaultversion:
 compatibilities[0].defaultVersion}' --output table
```

Un ejemplo de salida sería el siguiente.

```
-----------------------------------------
|          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   |
+-----------------+----------------------+
```

La versión con `True` que aparece en la columna `Defaultversion` es la versión con la que se creó el complemento, de forma predeterminada.

4. (Opcional) Busque las opciones de configuración del complemento elegido ejecutando el siguiente comando:

```
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\"}}}"
}
```


El resultado es un esquema JSON estándar.

Este es un ejemplo de valores de configuración válidos, en formato JSON, que funcionan con el esquema anterior.

```
{
  "resources": {
    "limits": {
      "cpu": "100m"
    }
  }
}
```

Este es un ejemplo de valores de configuración válidos, en formato YAML, que funcionan con el esquema anterior.

```
  resources:
    limits:
      cpu: 100m
```

5. Determine si el complemento requiere permisos de IAM. Si es así, debe (1) determinar si desea utilizar las Pod Identity de EKS o los roles de IAM para las cuentas de servicio (IRSA), (2) determinar el ARN de la función de IAM que se va a utilizar con el complemento y (3) determinar el nombre de la cuenta de servicio de Kubernetes que utiliza el complemento. Para obtener más información, consulte the section called “Cómo recuperar información de IAM”.
   - Amazon EKS sugiere usar las Pod Identity de EKS si el complemento lo admite. Esto requiere que el agente de Pod Identity esté instalado en el clúster. Para obtener más información sobre el uso de las Pod Identity con los complementos, consulte the section called “Roles de IAM”.
   - Si el complemento o el clúster no están configurados para las Pod Identity de EKS, utilice IRSA. Confirme que IRSA esté configurado en su clúster.
   - Consulte la documentación de complementos de Amazon EKS para determinar si el complemento requiere permisos de IAM y el nombre de la cuenta de servicio de Kubernetes asociada.
     a. Creación de un complemento de Amazon EKS. Copie el comando que sigue en su dispositivo. Realice las siguientes modificaciones en el comando según sea necesario y, a continuación, ejecute el comando modificado:
   - Reemplace *my-cluster* por el nombre de su clúster.

- Reemplace *vpc-cni* por un nombre de complemento devuelto en el paso anterior que desee crear.
- Reemplace *version-number* por una versión devuelta en el paso anterior que desee utilizar.
- Si desea instalar el complemento en un espacio de nombres de Kubernetes personalizado, agregue la opción `--namespace-config 'namespace=<my-namespace>`. Esta opción solo está disponible para los complementos de AWS y de la comunidad. Para obtener más información, consulte the section called “Espacio de nombres personalizado para complementos”
- Si el complemento no requiere permisos de IAM, elimine *<service-account-configuration>*.
- Realice una de las siguientes acciones:
  - Si el complemento (1) requiere permisos de IAM y (2) su clúster utiliza Pod Identities de EKS, reemplace *<service-account-configuration>* por la siguiente asociación de Pod Identity. Reemplace *<service-account-name>* por el nombre de la cuenta de servicio que utiliza el complemento. Reemplace *<role-arn>* con el ARN de un rol de IAM. El rol debe tener la política de confianza requerida por las Pod Identity de EKS.

    ```
    --pod-identity-associations 'serviceAccount=<service-account-name>,roleArn=<role-arn>'
    ```

  - Si el complemento (1) requiere permisos de IAM y (2) su clúster utiliza IRSA, reemplace *<service-account-configuration>* por la siguiente configuración de IRSA. Reemplace *111122223333* por el ID de su cuenta y *role-name* por el nombre del rol de IAM existente que creó. Para obtener instrucciones sobre cómo crear un rol, consulte la documentación del complemento que está creando. Para obtener una lista de los complementos, consulte the section called “ Complementos de AWS”. Para especificar un rol de cuenta de servicio, es necesario disponer de un proveedor de OpenID Connect (OIDC) de IAM para el clúster. Para determinar si ya tiene uno para su clúster o si debe crearlo, consulte the section called “Proveedor de OIDC de IAM”.

    ```
    --service-account-role-arn arn:aws::iam::111122223333:role/role-name
    ```

- Estos comandos de ejemplo sobrescriben la opción `--configuration-values` de cualquier versión autoadministrada existente del complemento, si la hay. Sustitúyalo por los valores de configuración que desee, como una cadena o una entrada de archivo. Si no desea proporcionar valores de configuración, elimine la opción `--configuration-`

`values`. Si no quiere que la CLI de AWS sobrescriba la configuración de un complemento autoadministrado existente, elimine la opción *`--resolve-conflicts OVERWRITE`*. Si lo hace, y el complemento de Amazon EKS necesita sobrescribir la configuración existente de un complemento autoadministrado, se produce un error en la creación del complemento de Amazon EKS y recibe un mensaje para ayudarlo a resolver el conflicto. Antes de especificar esta opción, asegúrese de que el complemento de Amazon EKS no administra la configuración que necesita administrar, ya que dicha configuración se sobrescribe con esta opción.

```
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
```

Para ver una lista completa de las opciones disponibles, consulte `create-addon` en la Referencia de los comandos de Amazon EKS. Si el complemento que ha creado aparece en `aws-marketplace` en la columna `Owner` de un paso anterior, es posible que no se pueda crear y que reciba un mensaje de error similar al siguiente.

```
{
    "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"
                }
            ]
        }
    }
```

```
}
```

Si recibe un error similar al del resultado anterior, visite la URL del resultado del paso anterior para suscribirse al complemento. Una vez suscrito, vuelva a ejecutar el comando `create-addon`.

Actualización de un complemento de Amazon EKS

Amazon EKS no actualiza de forma automática el complemento cuando se lanzan versiones nuevas o después de que actualice el clúster a una nueva versión secundaria de Kubernetes. Para actualizar el complemento en un clúster existente, debe iniciar la actualización. Luego de iniciar la actualización, Amazon EKS actualiza el complemento en su nombre. Antes de actualizar un complemento, consulte la documentación actual del complemento. Para obtener una lista de los complementos disponibles, consulte the section called " Complementos de AWS". Si el complemento requiere un rol de IAM, consulte los detalles del complemento específico en Complementos de Amazon EKS disponibles desde AWS para obtener información sobre cómo crear el rol.

Requisitos previos

Siga estos pasos antes de crear un complemento:

- Compruebe si el complemento requiere un rol de IAM. Para obtener más información, consulte the section called "Complementos de Amazon EKS".
- Compruebe que la versión del complemento de Amazon EKS sea compatible con el clúster. Para obtener más información, consulte the section called "Comprobación de la compatibilidad".

Procedimiento

Puede actualizar un complemento de Amazon EKS mediante `eksctl`, la Consola de administración de AWS o la AWS CLI.

Actualización del complemento (eksctl)

1. Determine los complementos y las versiones actuales de los complementos instalados en su clúster. Reemplace *`my-cluster`* por el nombre de su clúster.

```
eksctl get addon --cluster my-cluster
```

Un ejemplo de salida sería el siguiente.

```
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
```

El resultado puede tener un aspecto diferente, según los complementos y las versiones que tenga en su clúster. Puede ver que en el resultado del ejemplo anterior, dos complementos existentes en el clúster tienen versiones más recientes disponibles en la columna `UPDATE AVAILABLE`.

2. Actualice el complemento.

   a. Copie el comando que sigue en su dispositivo. Realice las siguientes modificaciones en el comando según sea necesario:

      - Reemplace *`my-cluster`* por el nombre de su clúster.
      - Reemplace *`region-code`* por la región de AWS en la que se encuentra el clúster.
      - Reemplace *`vpc-cni`* por el nombre de un complemento devuelto en el paso anterior que desea actualizar.
      - Si quiere una versión del complemento anterior a la versión más reciente, sustituya *`latest`* por el número de versión que aparece en el resultado del paso anterior que desea usar. Algunos complementos tienen versiones recomendadas. Para obtener más información, consulte la documentación del complemento que está actualizando. Para ver una lista de los complementos, consulte the section called “ Complementos de AWS”.* Si el complemento usa una cuenta de servicio de Kubernetes y un rol de IAM, sustituya *`111122223333`* por el ID de la cuenta y *`role-name`* por el nombre de un rol de IAM existente que haya creado. Para obtener instrucciones sobre cómo crear un rol, consulte la documentación del complemento que está creando. Para obtener una lista de los complementos, consulte the section called “ Complementos de AWS”. Para especificar un rol de cuenta de servicio, es necesario disponer de un proveedor de OpenID Connect (OIDC) de IAM para el clúster. Para determinar si ya tiene uno para su clúster o si debe crearlo, consulte the section called “Proveedor de OIDC de IAM”.

        Si el complemento no usa una cuenta de servicio de Kubernetes y un rol de IAM, elimine la línea `serviceAccountRoleARN: arn:aws:iam::`*`111122223333`*`:role/`*`role-name`* .

- La opción *`preserve`* (conservar) conserva los valores existentes del complemento. Si ha establecido valores personalizados para la configuración del complemento y no utiliza esta opción, Amazon EKS sobrescribe los valores con sus valores predeterminados. Si utiliza esta opción, le recomendamos que pruebe cualquier cambio de campo y valor en un clúster que no sea de producción antes de actualizar el complemento de su clúster de producción. Si cambia este valor a `overwrite`, todas las configuraciones cambiarán a los valores predeterminados de Amazon EKS. Si ha establecido valores personalizados para cualquier configuración, es posible que se sobrescriban con los valores predeterminados de Amazon EKS. Si cambia este valor a `none`, Amazon EKS no cambia el valor de ninguna configuración, pero la actualización podría fallar. Si se produce un error en la actualización, recibe un mensaje de error que lo ayuda a resolver el conflicto.

```
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. Ejecute el comando modificado para crear el archivo `update-addon.yaml`.

c. Aplique el archivo de configuración al clúster.

```
eksctl update addon -f update-addon.yaml
```

Para obtener más información acerca de cómo actualizar complementos, consulte Updating addons en la documentación de `eksctl`.

Actualización del complemento (consola de AWS)

1. Abra la consola de Amazon EKS.
2. En el panel de navegación izquierdo, elija Clusters (Clústeres).

3. Elija el nombre del clúster para el que desea actualizar el complemento.
4. Elija la pestaña Complementos.
5. Elija el complemento que desea actualizar.
6. Seleccione Editar.
7. En la página Configuración de ***`nombre del complemento`*** , haga lo siguiente:
   a. Elija la versión que desea utilizar. Es posible que el complemento tenga una versión recomendada. Para obtener más información, consulte la documentación del complemento que está actualizando. Para obtener una lista de los complementos, consulte the section called “ Complementos de AWS”.
   b. Tiene dos opciones para configurar roles para complementos: rol de IAM de EKS Pod Identities y roles de IAM para cuentas de servicio (IRSA). Siga el paso correspondiente a la opción que prefiera. Si todos los complementos que seleccionó tienen la opción Requiere suscripción en Estado, elija Siguiente. Para los complementos que no tienen Requiere suscripción en Estado, haga lo siguiente:
      i. En el caso del rol de IAM de Pod Identity para la cuenta de servicio, puede utilizar un rol de IAM de EKS Pod Identity existente o crear uno mediante el botón Crear rol recomendado. Este campo solo proporcionará opciones con la política de confianza adecuada. Si no hay ningún rol que seleccionar, significa que no existe ningún rol con la misma política de confianza. Para configurar un rol de IAM de EKS Pod Identity para las cuentas de servicio del complemento seleccionado, seleccione Crear rol recomendado. El asistente de creación de roles se abre en una ventana independiente. El asistente completará automáticamente la información del rol de la siguiente manera. Para cada complemento en el que desee crear el rol de IAM de EKS Pod Identity, complete los pasos del asistente de IAM como se indica a continuación.
         - En el paso Seleccionar entidad de confianza, la opción de servicio de AWS para EKS y el caso de uso para EKS: Pod Identity están preseleccionados, y la política de confianza adecuada se completará automáticamente para el complemento. Por ejemplo, el rol se creará con la política de confianza adecuada que contenga la entidad principal de IAM pods.eks.amazonaws.com, tal y como se detalla en the section called “Ventajas de las Pod Identities de EKS”. Elija Siguiente.
         - En el paso Agregar permisos, se preselecciona la política administrada adecuada para la política de roles para el complemento. Por ejemplo, para el complemento CNI de Amazon VPC, el rol se creará con la política administrada `AmazonEKS_CNI_Policy`, como se detalla en the section called “Complemento CNI de Amazon VPC para Kubernetes”. Elija Siguiente.

- En el paso Nombrar, revisar y crear, en Nombre de rol, el nombre de rol predeterminado se completa automáticamente para el complemento. Por ejemplo, para el complemento de CNI de Amazon VPC, el rol se creará con el nombre AmazonEKSPodIdentityAmazonVPCCNIRole. En Descripción, la descripción predeterminada se completa automáticamente con la descripción adecuada para el complemento. Por ejemplo, para el complemento CNI de Amazon VPC, el rol se creará con la descripción Permite que los pods que se ejecutan en el clúster de Amazon EKS accedan a los recursos de AWS. En Política de confianza, podrá ver la política de confianza completada para el complemento. Seleccione Crear rol.

> **Note**
>
> NOTA: Retener el nombre de rol predeterminado permite a EKS preseleccionar el rol para complementos en clústeres nuevos o al agregar complementos a clústeres existentes. Aún así, puede anular este nombre y el rol estará disponible para el complemento en todos los clústeres, pero el rol se tendrá que seleccionar manualmente en el menú desplegable.

ii. Para los complementos que no muestran Requiere suscripción en el Estado y en los que desea configurar roles mediante IRSA, consulte la documentación del complemento que va a crear para definir una política de IAM y asociarla a un rol. Para obtener una lista de los complementos, consulte the section called “ Complementos de AWS”. Para seleccionar un rol de IAM es necesario que tenga un proveedor de OpenID Connect (OIDC) de IAM para el clúster. Para determinar si ya tiene uno para su clúster o si debe crearlo, consulte the section called “Proveedor de OIDC de IAM”.

c. Seleccione Ajustes de configuración opcionales.

d. En Valores de configuración, ingrese cualquier información de configuración específica del complemento. Para obtener más información, consulte la documentación del complemento que está actualizando. Para ver una lista de los complementos, consulte the section called “ Complementos de AWS”…En Método de resolución de conflictos, seleccione una de las opciones. Si ha establecido valores personalizados para la configuración del complemento, le recomendamos que utilice la opción Preserve (Conservar). Si no elige esta opción, Amazon EKS sobrescribe sus valores con los valores predeterminados. Si utiliza esta opción, le recomendamos que pruebe cualquier cambio de campo y valor en un clúster que no sea de producción antes de actualizar el complemento de su clúster de producción. Si cambia este valor a Sobrescribir, todas las configuraciones cambiarán a los valores predeterminados de

Amazon EKS. Si ha establecido valores personalizados para cualquier configuración, es posible que se sobrescriban con los valores predeterminados de Amazon EKS. Si cambia este valor a Ninguno, Amazon EKS no cambia el valor de ninguna configuración, pero la actualización podría fallar. Si se produce un error en la actualización, recibe un mensaje de error que lo ayuda a resolver el conflicto.

8. Seleccione Save changes (Guardar cambios).

Actualización del complemento (AWS CLI)

1. Necesita la versión `2.12.3` o posterior, o bien, la versión `1.27.160` o posterior de la AWS Interfaz de la línea de comandos (AWS CLI) instalada y configurada en su dispositivo o AWS CloudShell. Para comprobar su versión actual, utilice `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente. Para actualizarla, consulte Instalación de AWS CLI en su directorio de usuarios principal en la Guía del usuario de AWS CloudShell.
2. Consulte la lista de complementos instalados. Reemplace *`my-cluster`* por el nombre de su clúster.

```
aws eks list-addons --cluster-name my-cluster
```

Un ejemplo de salida sería el siguiente.

```
{
    "addons": [
        "coredns",
        "kube-proxy",
        "vpc-cni"
    ]
}
```

3. Vea la versión actual del complemento que desea actualizar. Reemplace *`my-cluster`* por el nombre de su clúster y *`vpc-cni`* por el nombre del complemento que desea actualizar.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 "addon.addonVersion" --output text
```

Un ejemplo de salida sería el siguiente.

```
v1.10.4-eksbuild.1
```

4. Determine qué versiones del complemento se encuentran disponibles para la versión de su clúster. Reemplace *1.33* por la versión de su clúster y *vpc-cni* por el nombre del complemento que desea actualizar.

```
aws eks describe-addon-versions --kubernetes-version 1.33 --addon-name vpc-cni \
    --query 'addons[].addonVersions[].{Version: addonVersion, Defaultversion:
 compatibilities[0].defaultVersion}' --output table
```

Un ejemplo de salida sería el siguiente.

```
-----------------------------------------
|          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   |
+-----------------+----------------------+
```

La versión con `True` que aparece en la columna `Defaultversion` es la versión con la que se creó el complemento, de forma predeterminada.

5. Actualice su complemento. Copie el comando que sigue en su dispositivo. Realice las siguientes modificaciones en el comando según sea necesario y, a continuación, ejecute el comando modificado. Para obtener más información sobre este comando, consulte update-addon en la Referencia de la línea de comandos de Amazon EKS.
   - Reemplace *my-cluster* por el nombre de su clúster.
   - Reemplace *vpc-cni* por el nombre del complemento que desea actualizar y que se ha devuelto en el resultado de un paso anterior.

- Reemplace *version-number* por la versión a la que desea actualizar devuelta en el resultado del paso anterior. Algunos complementos tienen versiones recomendadas. Para obtener más información, consulte la documentación del complemento que está actualizando. Para ver una lista de los complementos, consulte the section called " Complementos de AWS".* Si el complemento usa una cuenta de servicio de Kubernetes y un rol de IAM, reemplace *111122223333* por el ID de su cuenta y *role-name* por el nombre de un rol de IAM existente que haya creado. Para obtener instrucciones sobre cómo crear un rol, consulte la documentación del complemento que está creando. Para obtener una lista de los complementos, consulte the section called " Complementos de AWS". Para especificar un rol de cuenta de servicio, es necesario disponer de un proveedor de OpenID Connect (OIDC) de IAM para el clúster. Para determinar si ya tiene uno para su clúster o si debe crearlo, consulte the section called "Proveedor de OIDC de IAM".

  Si el complemento no usa una cuenta de servicio de Kubernetes y un rol de IAM, elimine la línea `serviceAccountRoleARN: arn:aws:iam::`*111122223333*`:role/`*role-name* .

- La opción `--resolve-conflicts PRESERVE` conserva los valores existentes del complemento. Si ha establecido valores personalizados para la configuración del complemento y no utiliza esta opción, Amazon EKS sobrescribe los valores con sus valores predeterminados. Si utiliza esta opción, le recomendamos que pruebe cualquier cambio de campo y valor en un clúster que no sea de producción antes de actualizar el complemento de su clúster de producción. Si cambia este valor a `OVERWRITE`, todas las configuraciones cambiarán a los valores predeterminados de Amazon EKS. Si ha establecido valores personalizados para cualquier configuración, es posible que se sobrescriban con los valores predeterminados de Amazon EKS. Si cambia este valor a `NONE`, Amazon EKS no cambia el valor de ninguna configuración, pero la actualización podría fallar. Si se produce un error en la actualización, recibe un mensaje de error que lo ayuda a resolver el conflicto.

- Si desea eliminar toda la configuración personalizada, realice la actualización mediante la opción *`--configuration-values '{}'`*. Esto vuelve a establecer toda la configuración personalizada a los valores predeterminados. Si no quiere cambiar su configuración personalizada, no proporcione el indicador *`--configuration-values`*. Si desea ajustar una configuración personalizada, sustituya el *{}* por los nuevos parámetros.

```
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. Compruebe el estado de la actualización. Reemplace *my-cluster* por el nombre de su clúster y *vpc-cni* por el nombre del complemento que está actualizando.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni
```

Un ejemplo de salida sería el siguiente.

```
{
    "addon": {
        "addonName": "vpc-cni",
        "clusterName": "my-cluster",
        "status": "UPDATING",
    }
}
```

Cuando la actualización se completa, el estado cambia a `ACTIVE`.

Comprobación de la compatibilidad de la versión del complemento de Amazon EKS con un clúster

Antes de crear o actualizar un complemento de Amazon EKS, necesita comprobar si la versión de este es compatible con su clúster.

Use la API describe-addon-versions para enumerar las versiones disponibles de los complementos de EKS y las versiones de Kubernetes que admite cada versión del complemento.

1. Compruebe que la AWS CLI esté instalada y funcione con `aws sts get-caller-identity`. Si este comando no funciona, obtenga información en Introducción a AWS CLI.
2. Determine el nombre del complemento del que desea recuperar la información de compatibilidad de versiones; por ejemplo, `amazon-cloudwatch-observability`.
3. Determine la versión de Kubernetes del clúster, como `1.33`.
4. Utilice la AWS CLI para recuperar versiones de complementos compatibles con la versión de Kubernetes del clúster.

```
aws eks describe-addon-versions --addon-name amazon-cloudwatch-observability --
kubernetes-version 1.33
```

Un ejemplo de salida sería el siguiente.

```
{
    "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
                        }
                    ],
                }
            ]
        }
    ]
}
```

Este resultado muestra que la versión complementaria `vX.X.X-eksbuild.X` es compatible con la versión `1.33` del clúster de Kubernetes.

Compatibilidad de complementos con tipos de computación

El campo `computeTypes` del resultado `describe-addon-versions` indica la compatibilidad de un complemento con los nodos administrados del modo automático o los nodos híbridos de EKS. Los

complementos marcados con `auto` funcionan con la infraestructura administrada por AWS y basada en la nube del modo automático de EKS, mientras que los marcados con `hybrid` se pueden ejecutar en nodos en las instalaciones conectados al plano de control en la nube de EKS.

Para obtener más información, consulte the section called “Consideraciones para el modo automático de Amazon EKS”.

Cómo eliminar un complemento de Amazon EKS de un clúster

Puede quitar un complemento de Amazon EKS de su clúster mediante `eksctl`, la Consola de administración de AWS o AWS CLI.

Cuando elimine un complemento de Amazon EKS de un clúster, tenga en cuenta lo siguiente:

- No hay tiempo de inactividad para la funcionalidad que proporciona el complemento.
- Si utiliza los roles de IAM para las cuentas de servicio (IRSA) y el complemento tiene una función de IAM asociada, esta no se elimina.
- Si utiliza Pod Identities, se eliminarán todas las asociaciones de Pod Identity que sean propiedad del complemento. Si especifica la opción `--preserve` en la AWS CLI, las asociaciones se conservan.
- Amazon EKS deja de administrar la configuración del complemento.
- La consola deja de avisarle cuando haya nuevas versiones disponibles.
- No puede actualizar el complemento con ninguna herramienta o API de AWS.
- Puede optar por dejar el software de complemento en el clúster para poder autoadministrar el software de complemento o puede eliminar el software de complemento del clúster. Solo debe eliminar el complemento de software si ninguno de los recursos del clúster depende de la funcionalidad que proporciona el complemento.

Requisitos previos

Siga estos pasos antes de crear un complemento:

- Un clúster existente de Amazon EKS. Para implementar uno, consulte Introducción.
- Compruebe si el complemento requiere un rol de IAM. Para más información, consulte
- La versión `0.215.0` o posterior de la herramienta de línea de comandos `eksctl` instalada en su dispositivo o AWS CloudShell. Para instalar o actualizar `eksctl`, consulte la sección Installation de la documentación de `eksctl`.

Procedimiento

Tiene dos opciones al eliminar un complemento de Amazon EKS.

- Conservar el software del complemento en el clúster: esta opción elimina la administración de Amazon EKS de cualquier configuración. También elimina la capacidad de Amazon EKS de notificarle las actualizaciones y actualizar de forma automática el complemento de Amazon EKS después de iniciar una actualización. Sin embargo, conserva el software del complemento en el clúster. Esta opción hace que la instalación sea autoadministrada, en lugar de un complemento de Amazon EKS. Con esta opción, no hay tiempo de inactividad para el complemento.
- Eliminar por completo el software del complemento del clúster: recomendamos que elimine el complemento de Amazon EKS del clúster solo si no hay recursos en el clúster que dependan de él.

Puede eliminar un complemento de Amazon EKS mediante `eksctl`, la Consola de administración de AWS o AWS CLI.

Eliminar complemento (eksctl)

1. Determine los complementos instalados en su clúster. Reemplace *`my-cluster`* por el nombre de su clúster.

   ```
   eksctl get addon --cluster my-cluster
   ```

   Un ejemplo de salida sería el siguiente.

   ```
   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
   [...]
   ```

   El resultado puede tener un aspecto diferente, según los complementos y las versiones que tenga en su clúster.

2. Elimine el complemento. Reemplace *`my-cluster`* por el nombre de su clúster y *`name-of-add-on`* por el nombre del complemento que obtuvo en la salida del paso anterior que desea eliminar. Si elimina la opción *`--preserve`*, además de que Amazon EKS deja de administrar el complemento, se elimina el software del complemento del clúster.

```
eksctl delete addon --cluster my-cluster --name name-of-addon --preserve
```

Para obtener más información acerca de cómo eliminar complementos, consulte Deleting addons en la documentación de `eksctl`.

Eliminación del complemento (consola de AWS)

1. Abra la consola de Amazon EKS.
2. En el panel de navegación izquierdo, elija Clusters (Clústeres).
3. Elija el nombre del clúster para el que desea eliminar el complemento de Amazon EKS.
4. Elija la pestaña Complementos.
5. Seleccione el complemento que desea eliminar.
6. Elija Eliminar.
7. En el cuadro de diálogo de confirmación Eliminar: ***nombre del complemento*** , haga lo siguiente:
   a. Si desea que Amazon EKS deje de administrar la configuración del complemento, seleccione Conservar en clúster. Haga esto si desea retener el software del complemento en el clúster. Esto es para que pueda administrar todas las configuraciones del complemento por su cuenta.
   b. Ingrese el nombre del complemento.
   c. Elija Eliminar.

Eliminar complemento (AWS CLI)

1. Necesita tener la versión `0.215.0` o posterior de la herramienta de línea de comandos `eksctl` instalada en su dispositivo o AWS CloudShell. Para instalar o actualizar `eksctl`, consulte Instalación en la documentación de `eksctl`.
2. Consulte la lista de complementos instalados. Reemplace *`my-cluster`* por el nombre de su clúster.

```
aws eks list-addons --cluster-name my-cluster
```

Un ejemplo de salida sería el siguiente.

```
{
```

```
    "addons": [
        "coredns",
        "kube-proxy",
        "vpc-cni",
        "name-of-addon"
    ]
}
```

3. Elimine el complemento instalado. Reemplace *my-cluster* por el nombre de su clúster y *name-of-add-on* por el nombre del complemento que desea eliminar. Al eliminar *--preserve*, se elimina el software del complemento del clúster.

```
aws eks delete-addon --cluster-name my-cluster --addon-name name-of-addon --preserve
```

   A continuación se muestra el ejemplo abreviado de salida.

```
{
    "addon": {
        "addonName": "name-of-add-on",
        "clusterName": "my-cluster",
        "status": "DELETING",
    }
}
```

4. Compruebe el estado de la eliminación. Reemplace *my-cluster* por el nombre de su clúster y *name-of-add-on* por el nombre del complemento que desea eliminar.

```
aws eks describe-addon --cluster-name my-cluster --addon-name name-of-addon
```

   El resultado del ejemplo luego de que se elimina el complemento es el siguiente.

```
An error occurred (ResourceNotFoundException) when calling the DescribeAddon
 operation: No addon: name-of-addon found in cluster: my-cluster
```

Roles de IAM para los complementos de Amazon EKS

Algunos complementos de Amazon EKS necesitan permisos y roles de IAM para llamar a las API de AWS. Por ejemplo, el complemento CNI de Amazon VPC llama a determinadas API de AWS para configurar los recursos de red de su cuenta. Es necesario conceder permiso a estos complementos

mediante el IAM de . Más específicamente, la cuenta de servicio del pod que ejecuta el complemento debe estar asociada a un rol de IAM con una política de IAM específica.

La forma recomendada de conceder permisos AWS para agrupar cargas de trabajo es mediante la característica Pod Identity de Amazon EKS. Puede usar una asociación de Pod Identity para asignar la cuenta de servicio de un complemento a un rol de IAM. Si un pod usa una cuenta de servicio que tiene una asociación, Amazon EKS establece las variables de entorno en los contenedores del pod. Las variables de entorno configuran los SDK de AWS, incluida la AWS de CLI, para usar las credenciales de la Pod Identity de EKS. Para obtener más información, consulte the section called “Pod Identity”

Los complementos de Amazon EKS pueden ayudar a administrar el ciclo de vida de las asociaciones de Pod Identity correspondientes al complemento. Por ejemplo, puede crear o actualizar un complemento de Amazon EKS y la asociación de Pod Identity necesaria en una sola llamada a la API. Amazon EKS también proporciona una API para recuperar las políticas de IAM sugeridas.

1. Confirme que el agente de Pod Identity de Amazon EKS esté configurado en su clúster.
2. Determine si el complemento que desea instalar requiere permisos de IAM mediante la operación `describe-addon-versions` AWS CLI. Si el indicador `requiresIamPermissions` es `true`, entonces debe usar la operación `describe-addon-configurations` para determinar los permisos que necesita el complemento. La respuesta incluye una lista de políticas de IAM administradas sugeridas.
3. Recupere el nombre de la cuenta de servicio de Kubernetes y la política de IAM mediante la operación de la CLI `describe-addon-configuration`. Evalúe el alcance de la política sugerida en función de sus requisitos de seguridad.
4. Cree un rol de IAM con la política de permisos sugerida y la política de confianza que exige Pod Identity. Para obtener más información, consulte the section called “Creación de una asociación de Pod Identity (consola de AWS)”.
5. Cree o actualice el complemento de Amazon EKS con la CLI. Especifique al menos una asociación de Pod Identity. Una asociación de Pod Identity es el nombre de una cuenta de servicio de Kubernetes y el ARN de un rol de IAM.
   - Las asociaciones de Pod Identity creadas con las API de los complementos son propiedad del complemento correspondiente. Si elimina el complemento, también se eliminará la asociación de Pod Identity. Para evitar esta eliminación en cascada, utilice la opción `preserve` cuando elimine un complemento mediante la AWS CLI o la API. Si es necesario, también puede actualizar o eliminar directamente la asociación de Pod Identity. Los complementos no pueden asumir la propiedad de las asociaciones de Pod Identity existentes. Debe eliminar la

asociación existente y volver a crearla mediante una operación de creación o actualización de complementos.

- Amazon EKS recomienda usar asociaciones de Pod Identity para administrar los permisos de IAM para los complementos. El método anterior, los roles de IAM para cuentas de servicio (IRSA), aún se admite. Puede especificar tanto un `serviceAccountRoleArn` de IRSA como una asociación de Pod Identity para un complemento. Si el agente del Pod Identity de EKS está instalado en el clúster, `serviceAccountRoleArn` se ignorará y EKS utilizará la asociación de Pod Identity proporcionada. Si Pod Identity no está habilitada, se utilizará `serviceAccountRoleArn`.
- Si actualiza las asociaciones de Pod Identity de un complemento existente, Amazon EKS inicia un reinicio continuo de los pods del complemento.

Obtención de información de IAM sobre un complemento de Amazon EKS

Antes de crear un complemento, utilice AWS CLI para determinar lo siguiente:

- Si el complemento requiere permisos de IAM
- La política de IAM que se recomienda utilizar

Procedimiento

1. Determine el nombre del complemento que quiere instalar y la versión de Kubernetes del clúster. Para obtener más información sobre los complementos, consulte the section called “Complementos de Amazon EKS”.
2. Utilice AWS CLI para determinar si el complemento requiere permisos de IAM.

```
aws eks describe-addon-versions \
--addon-name <addon-name> \
--kubernetes-version <kubernetes-version>
```

Por ejemplo:

```
aws eks describe-addon-versions \
--addon-name aws-ebs-csi-driver \
--kubernetes-version 1.30
```

Revise la siguiente salida de ejemplo. Tenga en cuenta que `requiresIamPermissions` es `true` y la versión predeterminada del complemento. Debe especificar la versión del complemento al recuperar la política de IAM recomendada.

```
{
    "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. Si el complemento requiere permisos de IAM, utilice AWS CLI para recuperar una política de IAM recomendada.

```
aws eks describe-addon-configuration \
--query podIdentityConfiguration \
--addon-name <addon-name> \
--addon-version <addon-version>
```

Por ejemplo:

```
aws eks describe-addon-configuration \
```

```
--query podIdentityConfiguration \
--addon-name aws-ebs-csi-driver \
--addon-version v1.31.0-eksbuild.1
```

Revise la siguiente salida. Anote el `recommendedManagedPolicies`.

```
[
    {
        "serviceAccount": "ebs-csi-controller-sa",
        "recommendedManagedPolicies": [
            "arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy"
        ]
    }
]
```

4. Cree un rol de IAM y adjunte la política administrada recomendada. Como alternativa, revise la política administrada y reduzca los permisos según corresponda. Para obtener más información, consulte the section called “Creación de una asociación de Pod Identity (consola de AWS)”.

Referencia de compatibilidad de Pod Identity

En la siguiente tabla se indica si determinados complementos de Amazon EKS admiten EKS Pod Identity.

Nombre del complemento	Compatibilidad con Pod Identity	Versión mínima requerida
Controlador de CSI de Amazon EBS	Sí	v1.26.0-eksbuild.1
CNI de Amazon VPC	Sí	v1.15.5-eksbuild.1
Controlador de CSI de Amazon EFS	Sí	v2.0.5-eksbuild.1
AWS Distro para OpenTelemetry	Sí	v0.94.1-eksbuild.1
Controlador CSI del Mountpoint para Amazon S3	No	N/A
Agente de observabilidad de Amazon CloudWatch	Sí	v3.1.0-eksbuild.1

Esta tabla se actualizó por última vez el 28 de octubre de 2024.

Uso de Pod Identities para asignar un rol de IAM a un complemento de Amazon EKS

Algunos complementos de Amazon EKS necesitan permisos y roles de IAM. Antes de agregar o actualizar un complemento de Amazon EKS para que utilice una asociación de Pod Identity, compruebe el rol y la política que vaya a utilizar. Para obtener más información, consulte the section called “Cómo recuperar información de IAM”.

1. Determine:
   - `cluster-name`: el nombre del clúster de EKS en el que se va a instalar el complemento.
   - `addon-name`: el nombre del complemento de Amazon EKS que se va a instalar.
   - `service-account-name`— El nombre de la cuenta de servicio de Kubernetes utilizada por el complemento.
   - `iam-role-arn`— El ARN de un rol de IAM con permisos suficientes para el complemento. El rol debe tener la política de confianza requerida para Pod Identity de EKS. Para obtener más información, consulte the section called “Creación de una asociación de Pod Identity (consola de AWS)”.
2. Actualización del complemento mediante la AWS de CLI. También puede especificar las asociaciones de Pod Identity al crear un complemento, utilizando la misma sintaxis `--pod-identity-assocations`. Tenga en cuenta que si especifica las asociaciones de Pod Identity al actualizar un complemento, se sobrescriben todas las anteriores.

   ```
   aws eks update-addon --cluster-name <cluster-name> \
   --addon-name <addon-name> \
   --pod-identity-associations 'serviceAccount=<service-account-name>,roleArn=<role-arn>'
   ```

   Por ejemplo:

   ```
   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. Compruebe que se haya creado la asociación de Pod Identity:

   ```
   aws eks list-pod-identity-associations --cluster-name <cluster-name>
   ```

Si la operación se realiza correctamente, debería ver un resultado similar al siguiente. Anote el ARN del propietario del complemento EKS.

```
{
    "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"
        }
    ]
}
```

Eliminación de las asociaciones de Pod Identity de un complemento de Amazon EKS

Elimine las asociaciones de Pod Identity de un complemento de Amazon EKS.

1. Determine:
   - `cluster-name`: el nombre del clúster de EKS en el que se va a instalar el complemento.
   - `addon-name`: el nombre del complemento de Amazon EKS que se va a instalar.
2. Actualice el complemento para especificar una matriz vacía de asociaciones de Pod Identity.

```
aws eks update-addon --cluster-name <cluster-name> \
--addon-name <addon-name> \
--pod-identity-associations "[]"
```

Solución de problemas de Pod Identities de los complementos de EKS

Si sus complementos encuentran errores al intentar realizar operaciones de la API de AWS, SDK o CLI, confirme lo siguiente:

- El agente de Pod Identity está instalado en el clúster.

  - Para obtener información acerca de cómo instalar el agente de Pod Identity, consulte the section called “Configuración del agente”.
- El complemento tiene una asociación de Pod Identity válida.
  - Utilice la AWS CLI para recuperar las asociaciones del nombre de la cuenta de servicio que utiliza el complemento.

```
aws eks list-pod-identity-associations --cluster-name <cluster-name>
```

- El rol de IAM tiene la política de confianza necesaria para Pod Identities.
  - Utilice AWS CLI para recuperar la política de confianza de un complemento.

```
aws iam get-role --role-name <role-name> --query Role.AssumeRolePolicyDocument
```

- El rol de IAM tiene los permisos necesarios para el complemento.
  - Utilice AWS CloudTrail para revisar los eventos `AccessDenied` o `UnauthorizedOperation`.
- El nombre de la cuenta de servicio en la asociación de Pod Identity coincide con el nombre de la cuenta de servicio que utiliza el complemento.
  - Para obtener información acerca de los complementos disponibles, consulte the section called “ Complementos de AWS”.
- Compruebe la configuración de MutatingWebhookConfiguration denominada `pod-identity-webhook`
  - `admissionReviewVersions` del webhook debe ser `v1beta1` y no funciona con `v1`.

Determinación de los campos que se pueden personalizar para los complementos de Amazon EKS

Los complementos de Amazon EKS se instalan en el clúster mediante configuraciones estándar de prácticas recomendadas. Para obtener más información acerca de cómo agregar un complemento de Amazon EKS al clúster, consulte the section called “Complementos de Amazon EKS”.

Es posible que desee personalizar la configuración de un complemento de Amazon EKS para habilitar características avanzadas. Amazon EKS utiliza la característica de aplicación del servidor de Kubernetes para habilitar la administración de un complemento por parte de Amazon EKS sin sobrescribir la configuración de los ajustes que Amazon EKS no administra. Para obtener más información, consulte Server Side Apply (Aplicación del lado del servidor) en la documentación

de Kubernetes. Para ello, Amazon EKS administra un conjunto mínimo de campos para cada complemento que instala. Puede modificar todos los campos que no estén administrados por Amazon EKS u otro proceso de plano de control de Kubernetes, como `kube-controller-manager`, sin problema.

> **Important**
>
> La modificación de un campo administrado por Amazon EKS impide que Amazon EKS administre el complemento y puede provocar que los cambios se sobrescriban cuando se actualiza un complemento.

Sintaxis de administración de campos

Cuando se visualizan los detalles de un objeto de Kubernetes, los campos administrados y los no administrados se devuelven en la salida. Los campos administrados pueden ser de uno de los tipos siguientes:

- Completamente administrado: Amazon EKS administra todas las claves del campo. Las modificaciones de algún valor provocan un conflicto.
- Parcialmente administrado: Amazon EKS administra algunas claves del campo. Solo las modificaciones de las claves administradas explícitamente por Amazon EKS provocan un conflicto.

Ambos tipos de campos se etiquetan con `manager: eks`.

Cada clave es o bien un `.` que representa el campo en sí, que siempre se asigna a un conjunto vacío, o una cadena que representa un subcampo o elemento. La salida para la administración del campo consta de los siguientes tipos de declaraciones:

- `f:`*`name`* , donde *`name`* es el nombre de un campo de una lista.
- `k:`*`keys`* , donde *`keys`* es un mapa de los campos de un elemento de lista.
- `v:`*`value`* , donde *`value`* es el valor exacto con formato JSON de un elemento de lista.
- `i:`*`index`* , donde *`index`* es la posición de un elemento en la lista.

Las siguientes partes de salida para el complemento CoreDNS ilustran las declaraciones anteriores:

- Campos completamente administrados: si para un campo administrado se ha especificado `f:` (campo), pero no `k:` (clave), se administra todo el campo. Las modificaciones a los valores de este campo provocan un conflicto.

  En la siguiente salida, puede ver que el contenedor llamado `coredns` está administrado por `eks`. Los subcampos `args`, `image` y `imagePullPolicy` también están administrados por `eks`. Las modificaciones de algún valor de estos campos provocan un conflicto.

  ```
  [...]
  f:containers:
    k:{"name":"coredns"}:
    .: {}
    f:args: {}
    f:image: {}
    f:imagePullPolicy: {}
  [...]
  manager: eks
  [...]
  ```

- Campos parcialmente administrados: si una clave administrada tiene especificado un valor, se administran las claves declaradas para ese campo. La modificación de las claves especificadas provoca un conflicto.

  En el siguiente resultado, puede ver que `eks` administra los volúmenes `config-volume` y `tmp` establecidos con la clave `name`.

  ```
  [...]
  f:volumes:
    k:{"name":"config-volume"}:
      .: {}
      f:configMap:
        f:items: {}
        f:name: {}
      f:name: {}
    k:{"name":"tmp"}:
      .: {}
      f:name: {}
  [...]
  manager: eks
  [...]
  ```

- Adición de claves a campos parcialmente administrados: si solo se administra un valor de clave específico, puede agregar claves adicionales, como argumentos, a un campo sin provocar ningún conflicto. Si agrega claves adicionales, asegúrese de que el campo no esté administrado primero. Agregar o modificar cualquier valor administrado provoca un conflicto.

  En el siguiente resultado, puede ver que tanto la clave `name` como el campo `name` están administrados. Agregar o modificar cualquier nombre de contenedor provoca un conflicto con esta clave administrada.

```
[...]
f:containers:
  k:{"name":"coredns"}:
[...]
    f:name: {}
[...]
manager: eks
[...]
```

Procedimiento

Puede utilizar `kubectl` para ver qué campos administra Amazon EKS en cualquier complemento de Amazon EKS.

Puede modificar todos los campos que no estén administrados por Amazon EKS u otro proceso de plano de control de Kubernetes, como `kube-controller-manager`, sin problema.

1. Determine el complemento que desea examinar. Para ver todas las `deployments` y los DaemonSets implementados en el clúster, consulte the section called “Acceso a recursos del clúster”.
2. Para ver los campos administrados por un complemento, ejecute el siguiente comando:

```
kubectl get type/add-on-name -n add-on-namespace -o yaml
```

   Por ejemplo, puede ver los campos administrados para el complemento CoreDNS con el siguiente comando.

```
kubectl get deployment/coredns -n kube-system -o yaml
```

La administración de campos se muestra en la siguiente sección de la salida devuelta.

```
[...]
managedFields:
  - apiVersion: apps/v1
    fieldsType: FieldsV1
    fieldsV1:
[...]
```

> **Note**
>
> Si no ve `managedFields` en la salida, agregue `--show-managed-fields` al comando y ejecútelo de nuevo. La versión de `kubectl` que utiliza determina si los campos administrados se devuelven de forma predeterminada.

Pasos a seguir a continuación

Para su complemento, personalice los campos que no son propiedad de AWS.

Validación de las firmas de imágenes de contenedores durante la implementación

Si utiliza AWS Signer y desea verificar las imágenes de contenedores firmadas en el momento de la implementación, puede utilizar una de las siguientes soluciones:

- Gatekeeper y Ratify: utilice Gatekeeper como el controlador de admisión y a Ratify configurado con un complemento de AWS Signer como enlace web para validar las firmas.
- Kyverno: un motor de políticas de Kubernetes configurado con un complemento de AWS Signer para validar firmas.

> **Note**
>
> Antes de comprobar las firmas de las imágenes del contenedor, configure el almacén de confianza de Notation y la política de confianza, según lo exija el controlador de admisión seleccionado.

Capacidades de EKS

> **Tip**
>
> Introducción: Creación de una capacidad de ACK | Creación de una capacidad de Argo CD | Creación de una capacidad de kro

Las capacidades de Amazon EKS son un conjunto en capas de características de clúster completamente administradas que ayudan a acelerar la velocidad de los desarrolladores y reducir la complejidad de la creación y el escalado con Kubernetes. Las capacidades de EKS son características nativas de Kubernetes para la implementación continua declarativa, la administración de recursos de AWS y la creación y orquestación de recursos de Kubernetes, todo ello completamente administrado por AWS. Con las capacidades de EKS, puede centrarse más en crear y escalar sus cargas de trabajo, lo que lo libera de la carga operativa que suponen estos servicios fundamentales de la plataforma y que asumirá AWS. Estas capacidades se ejecutan dentro de EKS y no en los clústeres, lo que elimina la necesidad de instalar, mantener y escalar los componentes críticos de la plataforma en los nodos de trabajo.

Para comenzar, puede crear una o más capacidades de EKS en un clúster de EKS nuevo o existente. Para ello, puede utilizar la AWS CLI, la Consola de administración de AWS, las API de EKS, eksctl o las herramientas de infraestructura como código que prefiera. Si bien las capacidades de EKS están diseñadas para funcionar en conjunto, son recursos de nube independientes que puede elegir en función de su caso de uso y sus requisitos.

Todas las versiones de Kubernetes compatibles con EKS son compatibles con las capacidades de EKS.

> **Note**
>
> Las capacidades de EKS están disponibles en todas las regiones comerciales de AWS en las que Amazon EKS esté disponible. Para ver una lista de las regiones admitidas, consulte Puntos de conexión y cuotas de Amazon EKS en la referencia general de AWS.

Capacidades disponibles

Controladores para Kubernetes (ACK) de AWS

ACK permite administrar los recursos de AWS mediante las API de Kubernetes, lo que le permite crear y administrar buckets de S3, bases de datos de RDS, roles de IAM y otros recursos de AWS mediante los recursos personalizados de Kubernetes. ACK concilia continuamente el estado deseado con el estado real en AWS, lo que corrige cualquier desviación a lo largo del tiempo para mantener el sistema en buen estado y los recursos configurados según lo especificado. Puede administrar los recursos de AWS junto con sus cargas de trabajo de Kubernetes con las mismas herramientas y flujos de trabajo, con soporte para más de 50 servicios de AWS, como S3, RDS, DynamoDB y Lambda. ACK admite la administración de recursos entre cuentas y regiones, lo que permite arquitecturas complejas de administración de sistemas de varias cuentas y varios clústeres. ACK admite recursos de solo lectura y adopción de solo lectura, lo que facilita la migración de otras herramientas de infraestructura como código a sus sistemas basados en Kubernetes.

Más información sobre ACK →

Argo CD

Argo CD implementa la implementación continua basado en GitOps para sus aplicaciones, con los repositorios de Git como origen de información verdadera para sus cargas de trabajo y el estado del sistema. Argo CD sincroniza automáticamente los recursos de las aplicaciones con sus clústeres desde los repositorios de Git mediante la detección y la corrección de las desviaciones para garantizar que las aplicaciones implementadas coincidan con el estado deseado. Puede implementar y administrar aplicaciones en varios clústeres desde una sola instancia de Argo CD, con una implementación automatizada desde los repositorios de Git siempre que se hagan cambios. El uso conjunto de Argo CD y ACK proporciona un sistema de GitOps fundamental, que simplifica la administración de las dependencias de la carga de trabajo y permite diseñar todo el sistema, lo que incluye la administración de clústeres e infraestructuras a escala. Argo CD se integra con AWS Identity Center para la autenticación y la autorización, y proporciona una interfaz de usuario de Argo alojada para visualizar el estado de la aplicación y de implementación.

Más información sobre Argo CD →

kro (Kube Resource Orchestrator)

kro le permite crear API de Kubernetes personalizadas que agrupan varios recursos en abstracciones de nivel superior, lo que permite a los equipos de plataformas definir patrones

reutilizables para componentes de la nube de combinaciones de recursos comunes. Con kro, puede agrupar recursos de AWS y Kubernetes en abstracciones unificadas mediante una sintaxis simple para permitir configuraciones dinámicas y lógica condicional. kro permite a los equipos de plataformas proporcionar capacidades de autoservicio con las barreras de protección adecuadas, lo que permite a los desarrolladores aprovisionar infraestructuras complejas mediante API simples y personalizadas mientras cumplen con los estándares organizativos y las prácticas recomendadas. Los recursos de kro son simplemente recursos de Kubernetes y se especifica en los manifiestos de Kubernetes cuáles se pueden almacenar en Git o insertar en registros compatibles con OCI como Amazon ECR para una amplia distribución organizativa.

Más información sobre kro →

Ventajas de las capacidades de EKS

AWS administra completamente las capacidades de EKS, lo que elimina la necesidad de instalar, mantener y escalar los servicios de clúster fundamentales. AWS gestiona los parches de seguridad, las actualizaciones y la administración operativa, lo que permite a sus equipos centrarse en crear con AWS en lugar de operaciones del clúster. A diferencia de los complementos tradicionales de Kubernetes que consumen recursos del clúster, las capacidades se ejecutan en EKS y no en los nodos de trabajo. De este modo, se liberan recursos del clúster y capacidad para las cargas de trabajo y, al mismo tiempo, se minimiza la carga operativa que supone administrar los controladores integrados en el clúster y otros componentes de la plataforma.

Con las capacidades de EKS, puede administrar las implementaciones, los recursos de AWS, los recursos personalizados de Kubernetes y las composiciones mediante herramientas y API de Kubernetes nativas, como `kubectl`. Todas las capacidades operan en el contexto de sus clústeres, lo que hace que se detecten y se corrijan automáticamente las desviaciones de configuración en los recursos de la infraestructura en la nube y de las aplicaciones. Puede implementar y administrar los recursos en varios clústeres, cuentas de AWS y regiones desde un único punto de control, lo que simplifica las operaciones en entornos complejos y distribuidos.

Las capacidades de EKS están diseñadas para los flujos de trabajo de GitOps, lo que proporciona una administración de aplicaciones e infraestructuras declarativa y controlada por versiones. Los cambios fluyen desde Git a través del sistema y proporcionan registros de auditoría, capacidades de reversión y flujos de trabajo de colaboración que se integran con las prácticas de desarrollo existentes. Gracias a este enfoque nativo de Kubernetes, no es necesario utilizar varias herramientas ni administrar sistemas de infraestructura como código externos a los clústeres y, además, hay un

único origen de información verdadera al que acudir. El estado deseado, definido en la configuración declarativa de Kubernetes con control de versiones, se aplica continuamente en todo el entorno.

Precios

Con las capacidades de EKS, no hay cuotas de pago iniciales ni compromisos. Se le cobrará por cada hora que esté activo cada recurso de capacidad en su clúster de Amazon EKS. Los recursos específicos de Kubernetes administrados por las capacidades de EKS también se facturan con una tarifa por hora.

Para obtener información actual acerca de los precios, consulte la página de precios de Amazon EKS.

> **Tip**
>
> Puede usar el Informe de costos y usos y el Explorador de costos de AWS para hacer un seguimiento de los costos de la capacidad por separado en relación con otros cargos de EKS. Puede etiquetar las capacidades con el nombre del clúster, el tipo de capacidad y otros detalles para poder asignar los costos.

Cómo funcionan las capacidades de EKS

Cada capacidad es un recurso de AWS que se crea en el clúster de EKS. Una vez creada, la capacidad se ejecuta en EKS y AWS la administra completamente.

> **Note**
>
> Puede crear un recurso de capacidad de cada tipo (Argo CD, ACK y kro) para un clúster determinado. No puede crear varios recursos de capacidad del mismo tipo en el mismo clúster.

Interactuará con las capacidades del clúster mediante las API y herramientas estándar de Kubernetes:

- Utilice `kubectl` para aplicar los recursos personalizados de Kubernetes.

- Utilice los repositorios de Git como origen de información verdadera para los flujos de trabajo de GitOps.

Algunas capacidades incluyen herramientas adicionales compatibles. Por ejemplo:

- Utilice la CLI de Argo CD para configurar y administrar repositorios y clústeres en la capacidad de Argo CD.
- Utilice la interfaz de usuario de Argo CD para ver y administrar las aplicaciones administradas por la capacidad de Argo CD.

Las capacidades están diseñadas para funcionar en conjunto, pero son independientes y completamente opcionales. Puede habilitar una, dos o las tres capacidades en función de sus necesidades y actualizar la configuración a medida que evolucionen sus requisitos.

Se admiten todos los tipos de computación de EKS para su uso con las capacidades de EKS. Para obtener más información, consulte Administración de la computación.

Para obtener información sobre la configuración de seguridad y los detalles sobre los roles de IAM, consulte the section called “Consideraciones para las capacidades de EKS”. Para conocer los patrones de arquitectura de varios clústeres, consulte the section called “Consideraciones”-

Casos de uso común

GitOps para aplicaciones e infraestructura

Use Argo CD para implementar aplicaciones y componentes operativos y ACK para administrar las configuraciones de los clústeres y aprovisionar la infraestructura, ambos desde los repositorios de Git. Toda la pila (aplicaciones, bases de datos, almacenamiento y redes) se define como código y se implementa automáticamente.

Ejemplo: un equipo de desarrollo hace cambios en Git. Argo CD implementa la aplicación actualizada y ACK aprovisiona una nueva base de datos de RDS con la configuración correcta. Todos los cambios son auditables, reversibles y coherentes en todos los entornos.

Ingeniería de plataformas con autoservicio

Utilice kro para crear API personalizadas que compongan los recursos de ACK y Kubernetes. Los equipos de plataformas definen los patrones aprobados con barreras de protección. Los equipos de aplicaciones utilizan API sencillas y de alto nivel para aprovisionar pilas completas.

Ejemplo: un equipo de plataformas crea una API “WebApplication” que aprovisiona una implementación, un servicio, una entrada y un bucket de S3. Los desarrolladores utilizan esta API sin necesidad de comprender la complejidad subyacente ni los permisos de AWS.

Administración de aplicaciones de varios clústeres

Utilice Argo CD para implementar aplicaciones en varios clústeres de EKS en diferentes regiones o cuentas. Administre todas las implementaciones desde una única instancia de Argo CD con políticas y flujos de trabajo coherentes.

Ejemplo: implemente la misma aplicación en clústeres de desarrollo, almacenamiento y producción en varias regiones. Argo CD garantiza que cada entorno se mantenga sincronizado con su rama de Git correspondiente.

Administración de varios clústeres

Utilice ACK para definir y aprovisionar los clústeres de EKS, kro para personalizar las configuraciones de los clústeres con los estándares organizativos y Argo CD para administrar el ciclo de vida y la configuración de los clústeres. De este modo, se proporciona una administración integral de los clústeres, desde su creación hasta las operaciones continuas.

Ejemplo: defina los clústeres de EKS mediante ACK y kro para aprovisionar y aprovisionar la infraestructura de los clústeres, lo que define los estándares organizativos para las redes, las políticas de seguridad, los complementos y otras configuraciones. Utilice Argo CD para crear y administrar de forma continua los clústeres, la configuración y las actualizaciones de las versiones de Kubernetes en toda su flota, de modo que aproveche los estándares uniformes y la administración automatizada del ciclo de vida.

Migraciones y modernización

Simplifique la migración a EKS con el aprovisionamiento nativo de recursos en la nube y los flujos de trabajo de GitOps. Utilice ACK para adoptar los recursos de AWS existentes sin volver a crearlos y Argo CD para poner en marcha las implementaciones de cargas de trabajo desde Git.

Ejemplo: un equipo que migra de EC2 a EKS adopta sus bases de datos de RDS y buckets de S3 existentes mediante ACK y, a continuación, utiliza Argo CD para implementar aplicaciones en contenedores desde Git. La ruta de migración es clara y las operaciones se estandarizan desde el primer día.

Arranque de cuentas y regiones

Automatice la implementación de la infraestructura en todas las cuentas y regiones con Argo CD y ACK a la vez. Defina su infraestructura como código en Git y deje que las capacidades se ocupen de la implementación y la administración.

Ejemplo: un equipo de plataformas mantiene los repositorios de Git que definen las configuraciones de cuentas estándar: VPC, roles de IAM, instancias de RDS y pilas de supervisión. Argo CD implementa estas configuraciones automáticamente en nuevas cuentas y regiones, lo que garantiza la coherencia y reduce el tiempo de configuración manual de días a minutos.

Uso de recursos de capacidades

En este tema, se describen las operaciones comunes para administrar los recursos de capacidades en todos los tipos de capacidades.

Recursos de capacidades de EKS

Las capacidades de EKS son recursos de AWS que permiten la funcionalidad administrada de su clúster de Amazon EKS. Las capacidades se ejecutan en EKS, lo que elimina la necesidad de instalar y mantener controladores y otros componentes operativos en los nodos de trabajo. Las capacidades se crean para un clúster de EKS específico y permanecen asociadas a ese clúster durante todo su ciclo de vida.

Cada recurso de capacidad tiene lo siguiente:

- Un nombre único dentro de su clúster
- Un tipo de capacidad (ACK, ARGOCD o KRO)
- Un nombre de recurso de Amazon (ARN), que especifica tanto el nombre como el tipo
- Un rol de IAM de capacidad
- Un estado que indica su estado actual
- Configuración, tanto genérica como específica del tipo de capacidad

Descripción de los estados de las capacidades

Los recursos de las capacidades tienen un estado que indica su estado actual. Puede ver el estado de la capacidad en la consola de EKS o mediante la AWS CLI.

Consola:

1. Abra la consola de Amazon EKS en https://console.aws.amazon.com/eks/home#/clusters.
2. Seleccione el nombre del clúster.
3. Seleccione la pestaña Capacidades para ver el estado de todas las capacidades.
4. Para obtener información detallada sobre el estado, seleccione la pestaña Observabilidad, luego Supervisar clúster y, por último, la pestaña Capacidades.

AWS CLI:

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-capability-name
```

Estados de las capacidades

CREANDO: Se está configurando la capacidad. Puede salir de la consola, ya que la capacidad se continuará creando en segundo plano.

ACTIVA: la capacidad está en ejecución y lista para usarse. Si los recursos no funcionan según lo esperado, compruebe el estado de los recursos y los permisos de IAM. Para obtener orientación, consulte the section called “Solución de problemas de capacidades”.

ACTUALIZANDO: se están aplicando cambios en la configuración. Espere a que el estado vuelva a ser `ACTIVE`.

ELIMINANDO: se está eliminando la capacidad del clúster.

CREATE_FAILED: la configuración detectó un error. Entre las causas comunes, se incluyen las siguientes:

- Falta la política de confianza del rol de IAM o no se encuentra
- El rol de IAM no existe o no se puede acceder a él
- Problemas de acceso al clúster
- Parámetros de configuración no válidos

Consulte la sección de estado de la capacidad para obtener detalles específicos sobre los errores.

UPDATE_FAILED: no se pudo actualizar la configuración. Consulte la sección de estado de la capacidad para obtener más información y compruebe los permisos de IAM.

> **Tip**
>
> Para obtener ayuda sobre la solución de problemas, consulte:
>
> - the section called “Solución de problemas de capacidades”: solución de problemas general de la capacidad
> - the section called “Solución de problemas”: problemas específicos de ACK
> - the section called “Solución de problemas”: problemas específicos de Argo CD
> - the section called “Solución de problemas”: problemas específicos de kro

Creación de capacidades

Para crear una capacidad en el clúster, consulte los temas siguientes:

- the section called “Creación de una capacidad de ACK”: creación de una capacidad de ACK para administrar los recursos de AWS mediante las API de Kubernetes
- the section called “Creación de la capacidad Argo CD”: creación de una capacidad de Argo CD para la entrega continua de GitOps
- the section called “Creación de una capacidad de kro”: creación de una capacidad de kro para la composición y orquestación de los recursos

Enumeración de capacidades

Puede enumerar todos los recursos de capacidades de un clúster.

Consola

1. Abra la consola de Amazon EKS en https://console.aws.amazon.com/eks/home#/clusters.
2. Seleccione el nombre del clúster para abrir la página de detalles del clúster.
3. Elija la pestaña Capacidades.
4. Consulte los recursos de capacidades en Capacidades administradas.

AWS CLI

Use el comando `list-capabilities` para ver todas las capacidades del clúster. Reemplace *`region-code`* por la región de AWS donde creó el clúster y *`my-cluster`* por el nombre de su clúster.

```
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"
        }
```

```
    ]
}
```

Descripción de una capacidad

Obtenga información detallada sobre una capacidad específica, lo que incluye su configuración y estado.

Consola

1. Abra la consola de Amazon EKS en https://console.aws.amazon.com/eks/home#/clusters.
2. Seleccione el nombre del clúster para abrir la página de detalles del clúster.
3. Elija la pestaña Capacidades.
4. Elija la capacidad que desee ver en Capacidades administradas.
5. Vea los detalles de la capacidad, lo que incluye el estado, la configuración y la hora de creación.

AWS CLI

Utilice el comando `describe-capability` para ver información detallada. Sustituya *region-code* por la región de AWS en la que se encuentra el clúster, *my-cluster* por el nombre del clúster y *capability-name* por el nombre de la capacidad (ack, argocd o kro).

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name capability-name
```

Ejemplo de código de salida:

```
{
  "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"
  }
}
```

Actualización de la configuración de una capacidad

Puede actualizar ciertos aspectos de la configuración de una capacidad después de su creación. Las opciones de configuración específicas varían según el tipo de capacidad.

> **Note**
>
> Los recursos de capacidades de EKS se administran completamente, lo que incluye las actualizaciones de versiones y los parches. Al actualizar una capacidad, se actualizará la configuración de los recursos y no se actualizarán las versiones de los componentes de la capacidad administrada.

AWS CLI

Utilice el comando `update-capability` para modificar una capacidad:

```
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**
>
> No todas las propiedades de capacidades se pueden actualizar después de su creación. Consulte la documentación específica de la capacidad para obtener detalles sobre lo que se puede modificar.

Eliminación de una capacidad

Cuando ya no necesite una capacidad del clúster, puede eliminar el recurso de la capacidad.

Important

Elimine los recursos del clúster antes de eliminar la capacidad.
La eliminación de un recurso de la capacidad no elimina automáticamente los recursos creados a través de esa capacidad:

- Todas las definiciones de recursos personalizados (CRD) de Kubernetes permanecen instaladas en el clúster.
- Los recursos de ACK permanecen en el clúster, mientras que los recursos de AWS correspondientes permanecen en la cuenta.
- Las aplicaciones de Argo CD y sus recursos de Kubernetes permanecen en el clúster.
- Las instancias y ResourceGraphDefinitions de kro permanecen en el clúster.

Debe eliminar estos recursos antes de eliminar la capacidad para evitar recursos huérfanos. Si lo desea, puede optar por retener los recursos de AWS asociados a los recursos de Kubernetes de ACK. Consulte Consideraciones sobre ACK.

Consola

1. Abra la consola de Amazon EKS en https://console.aws.amazon.com/eks/home#/clusters.
2. Seleccione el nombre del clúster para abrir la página de detalles del clúster.
3. Elija la pestaña Capacidades.
4. Seleccione en la lista Capacidades administradas la capacidad que desea eliminar.
5. Elija Eliminar capacidad.
6. En el cuadro de diálogo de confirmación, ingrese el nombre de la capacidad para confirmar la eliminación.
7. Elija Eliminar.

AWS CLI

Utilice el comando `delete-capability` para eliminar un recurso de la capacidad:

Sustituya *region-code* por la región de AWS en la que se encuentra el clúster, *my-cluster* por el nombre del clúster y *capability-name* por el nombre de la capacidad que desea eliminar.

```
aws eks delete-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name capability-name
```

Siguientes pasos

- the section called “Recursos de Kubernetes”: más información sobre los recursos de Kubernetes que proporciona cada tipo de capacidad
- the section called “Conceptos de ACK”: descripción de los conceptos de ACK y el ciclo de vida de los recursos
- the section called “Uso de Argo CD”: uso de las capacidades de Argo CD para flujos de trabajo de GitOps
- the section called “Conceptos de kro”: descripción de los conceptos de kro y la composición de recursos

Recursos de Kubernetes de las capacidades

Una vez que active una capacidad en el clúster, por lo general, interactuará con ella mediante la creación y administración de los recursos personalizados de Kubernetes en el clúster. Cada capacidad proporciona su propio conjunto de definiciones de recursos personalizados (CRD) que amplían la API de Kubernetes con funcionalidades específicas de cada capacidad.

Recursos de Argo CD

Al activar la capacidad de Argo CD, puede crear y administrar los siguientes recursos de Kubernetes:

Aplicación

Define una implementación desde un repositorio de Git a un clúster de destino. Los recursos de `Application` especifican el repositorio de origen, el espacio de nombres de destino y la política de sincronización. Puede crear hasta 1000 recursos de `Application` por instancia de capacidad de Argo CD.

ApplicationSet

Genera varios recursos de `Application` a partir de plantillas, lo que permite implementaciones en varios clústeres y entornos. Los recursos de `ApplicationSet` utilizan generadores para crear recursos de `Application` de forma dinámica en función de listas de clústeres, directorios de Git u otros orígenes.

AppProject

Proporciona agrupamiento lógico y control de acceso a los recursos de `Application`. Los recursos de `AppProject` definen qué repositorios, clústeres y espacios de nombres pueden usar los recursos de `Application`, lo que permite establecer límites de seguridad y de tenencia múltiple.

Ejemplo de recurso de `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
```

Para obtener más información sobre los conceptos y recursos de Argo CD, consulte the section called “Conceptos de Argo CD”.

Recursos de kro

Al activar la capacidad de kro, puede crear y administrar los siguientes recursos de Kubernetes:

ResourceGraphDefinition (RGD)

Define una API personalizada que compone varios Kubernetes y recursos de AWS en una abstracción de nivel superior. Los equipos de plataformas crean recursos de

`ResourceGraphDefinition` para proporcionar patrones reutilizables con barreras de protección.

Instancias de recursos personalizados

Tras crear un recurso de `ResourceGraphDefinition`, puede crear instancias de la API personalizada definida por la `ResourceGraphDefinition`. kro crea y administra automáticamente los recursos especificados en la `ResourceGraphDefinition`.

Ejemplo de recurso de `ResourceGraphDefinition`:

```
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
```

Ejemplo de instancia de `WebApplication`:

```
apiVersion: v1alpha1
kind: WebApplication
metadata:
  name: my-web-app
  namespace: default
spec:
```

```
  name: my-web-app
  replicas: 3
```

Al aplicar esta instancia, kro crea automáticamente los recursos de `Deployment` y `Service` definidos en la `ResourceGraphDefinition`.

Para obtener más información sobre los conceptos y recursos de kro, consulte the section called "Conceptos de kro".

Recursos de ACK

Al activar la capacidad de ACK, puede crear y administrar los recursos de AWS mediante los recursos personalizados de Kubernetes. ACK proporciona más de 200 CRD para más de 50 servicios de AWS, lo que le permite definir los recursos de AWS junto con sus cargas de trabajo de Kubernetes y administrar los recursos de infraestructura de AWS dedicados con Kubernetes.

Ejemplos de recursos de ACK:

Bucket de S3

: Los recursos de `Bucket` crean y administran buckets de Amazon S3 con políticas de control de versiones, cifrado y ciclo de vida.

DBInstance de RDS

: Los recursos de `DBInstance` aprovisionan y administran las instancias de bases de datos de Amazon RDS con copias de seguridad automatizadas y plazos de mantenimiento.

Tabla de DynamoDB

: Los recursos de `Table` crean y administran tablas de DynamoDB con capacidad aprovisionada o bajo demanda.

Rol de IAM

: Los recursos de `Role` definen los roles de IAM con políticas de confianza y políticas de permisos para el acceso a los servicios de AWS.

Función Lambda

: Los recursos de `Function` crean y administran funciones de Lambda con configuración de código, tiempo de ejecución y rol de ejecución.

Ejemplo de especificación de un recurso de `Bucket`:

```
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
```

Para obtener más información sobre los conceptos y recursos de ACK, consulte the section called "Conceptos de ACK".

Límites de recursos

Las capacidades de EKS tienen los siguientes límites de recursos:

Límites de uso de Argo CD:

- Máximo de 1000 recursos de `Application` por instancia de capacidad de Argo CD
- Máximo de 100 clústeres remotos configurados por instancia de capacidad de Argo CD

Límites de configuración de recursos:

- Máximo 64 recursos de Kubernetes por recurso de `Application` en Argo CD
- Máximo 64 recursos de Kubernetes por `ResourceGraphDefinition` en kro

> **Note**
>
> Estos límites se aplican a la cantidad de recursos administrados por cada instancia de capacidad. Si necesita límites más elevados, puede implementar capacidades en varios clústeres.

Siguientes pasos

Para obtener información sobre tareas específicas de las capacidades y configuración avanzada, consulte los siguientes temas:

- the section called “Conceptos de ACK”: descripción de los conceptos de ACK y el ciclo de vida de los recursos
- the section called “Uso de Argo CD”: uso de las capacidades de Argo CD para flujos de trabajo de GitOps
- the section called “Conceptos de kro”: descripción de los conceptos de kro y la composición de recursos

Consideraciones sobre las capacidades de EKS

En este tema, se tratan aspectos importantes a la hora de utilizar las capacidades de EKS, como el diseño del control de acceso, la elección entre las capacidades de EKS y las soluciones autoadministradas, los patrones de arquitectura para las implementaciones de varios clústeres y las prácticas recomendadas operativas.

RBAC de Kubernetes y roles de IAM de la capacidad

Cada recurso de la capacidad de EKS tiene un rol de IAM de capacidad configurado. El rol de capacidad se utiliza para otorgar permisos del servicio de AWS para que las capacidades de EKS actúen en su nombre. Por ejemplo, para utilizar la capacidad de EKS para ACK con el fin de administrar buckets de Amazon S3, debe otorgar permisos administrativos al bucket de S3 para la capacidad, lo que le permitirá crear y administrar buckets.

Una vez configurada la capacidad, los recursos de S3 en AWS se pueden crear y administrar con los recursos personalizados de Kubernetes en el clúster. El RBAC de Kubernetes es el mecanismo de control de acceso integrado en el clúster para determinar qué usuarios y grupos pueden crear y administrar esos recursos personalizados. Por ejemplo, otorga permisos específicos a usuarios y grupos del RBAC de Kubernetes para crear y administrar los recursos de `Bucket` en los espacios de nombres que elija.

De este modo, IAM y el RBAC de Kubernetes son dos mitades del sistema de control de acceso integral que regula los permisos relacionados con los recursos y las capacidades de EKS. Es importante diseñar la combinación adecuada de políticas de acceso de RBAC y permisos de IAM para su caso de uso.

Para obtener información adicional sobre los permisos de Kubernetes y los roles de IAM de la capacidad, consulte the section called “Consideraciones para las capacidades de EKS”.

Patrones de arquitectura de varios clústeres

Al implementar capacidades en varios clústeres, tenga en cuenta estos patrones arquitectónicos comunes:

Radial con administración centralizada

Ejecute las tres capacidades en un clúster administrado de forma centralizada para organizar las cargas de trabajo y administrar la infraestructura en la nube en varios clústeres de cargas de trabajo.

- Argo CD en el clúster de administración implementa aplicaciones en clústeres de cargas de trabajo de distintas regiones o cuentas
- ACK en el clúster de administración aprovisiona los recursos de AWS (RDS, S3, IAM) para todos los clústeres
- kro en el clúster de administración crea abstracciones de plataforma portátiles que funcionan en todos los clústeres

Este patrón centraliza la administración de la carga de trabajo y la infraestructura en la nube, y puede simplificar las operaciones de las organizaciones que administran muchos clústeres.

GitOps descentralizadas

Las cargas de trabajo y la infraestructura en la nube se administran mediante capacidades del mismo clúster en el que se ejecutan las cargas de trabajo.

- Argo CD administra los recursos de las aplicaciones en el clúster local.
- Los recursos de ACK se utilizan para las necesidades de los clústeres y las cargas de trabajo.
- Las abstracciones de la plataforma de kro se instalan y orquestan los recursos locales.

Este patrón descentraliza las operaciones, ya que los equipos administran sus propios servicios dedicados de la plataforma en uno o más clústeres.

Radial con implementación híbrida de ACK

Combine modelos centralizados y descentralizados, con implementaciones de aplicaciones centralizadas y administración de recursos en función del ámbito y la propiedad.

- Clúster de concentrador:
  - Argo CD administra las implementaciones de GitOps en el clúster local y en todos los clústeres de cargas de trabajo remotos.
  - ACK se utiliza en el clúster de administración para los recursos del ámbito administrativo (bases de datos de producción, roles de IAM, VPC).
  - kro se usa en el clúster de administración para abstracciones de plataforma reutilizables.
- Clúster de radio:
  - Las cargas de trabajo se administran mediante Argo CD en el clúster de concentrador centralizado.
  - ACK se usa localmente para los recursos del ámbito de la carga de trabajo (buckets de S3, instancias de ElastiCache, colas de SQS).
  - kro se usa localmente para la composición de recursos y los patrones de componentes

Este patrón separa las preocupaciones: los equipos de plataformas administran la infraestructura crítica de forma centralizada en clústeres de administración, lo que incluye opcionalmente los clústeres de cargas de trabajo, mientras que los equipos de aplicaciones especifican y administran los recursos de la nube junto con las cargas de trabajo.

Elección de un patrón

Tenga en cuenta estos factores al seleccionar una arquitectura:

- Estructura organizativa: los equipos de plataformas centralizadas prefieren los patrones de concentrador, mientras que los equipos descentralizados pueden preferir las capacidades por clúster.
- Ámbito de los recursos: los recursos de ámbito administrativo (bases de datos, IAM) suelen beneficiarse de la administración central, mientras que los recursos de cargas de trabajo (buckets, colas) se pueden administrar de forma local.
- Autoservicio: los equipos de plataformas centralizadas pueden crear y distribuir recursos personalizados prescriptivos para permitir el autoservicio seguro de los recursos de la nube para las necesidades de carga de trabajo comunes.
- Administración de flotas del clúster: los clústeres de administración centralizada proporcionan un plano de control que es propiedad del cliente para la administración de flotas en clústeres de EKS, junto con otros recursos del ámbito administrativo.

- Requisitos de cumplimiento: algunas organizaciones requieren un control centralizado para la auditoría y la gobernanza.
- Complejidad operativa: un número menor de instancias de capacidad simplifica las operaciones, pero puede crear cuellos de botella.

> **Note**
>
> Puede comenzar con un patrón y evolucionar hacia otro a medida que la plataforma vaya madurando. Las capacidades son independientes: puede implementarlas de forma diferente en los clústeres en función de sus necesidades.

Comparación de las capacidades de EKS con las soluciones autoadministradas

Las capacidades de EKS proporcionan experiencias completamente administradas para las herramientas y controladores populares de Kubernetes que se ejecutan en EKS. Difieren de las soluciones auto administradas, que se instalan y se utilizan en el clúster.

Diferencias clave

Implementación y administración

AWS administra completamente las capacidades de EKS sin necesidad de instalar, configurar ni mantener el software de los componentes. AWS instala y administra automáticamente todas las definiciones de recursos personalizados (CRD) de Kubernetes necesarias en el clúster.

Con las soluciones autoadministradas, puede instalar y configurar el software de clústeres mediante gráficos de Helm, kubectl u otros operadores. Tiene el control total sobre el ciclo de vida del software y la configuración del tiempo de ejecución de las soluciones autoadministradas, lo que permite personalizar cualquier capa de la solución.

Operaciones y mantenimiento

AWS administra la aplicación de parches y otras operaciones del ciclo de vida del software para las capacidades de EKS, con actualizaciones automáticas y parches de seguridad. Las capacidades de EKS se integran con características de AWS que permiten simplificar las configuraciones, ofrecen alta disponibilidad y tolerancia a errores integradas, y eliminan la solución de problemas de las cargas de trabajo de los controladores dentro del clúster.

Las soluciones autoadministradas requieren que supervise el estado y los registros de los componentes, aplique parches de seguridad y actualizaciones de versiones, configure la alta disponibilidad con varias réplicas y presupuestos para interrupciones de pods, solucione y corrija los problemas de cargas de trabajo de los controladores y administre los lanzamientos y las versiones. Tiene el control total sobre las implementaciones, pero esto suele requerir soluciones personalizadas para el acceso a clústeres privados y otras integraciones que deben ajustarse a los estándares de la organización y a los requisitos de cumplimiento en materia de seguridad.

Consumo de recursos

Las capacidades de EKS se ejecutan en EKS y fuera de los clústeres, lo que libera recursos de nodos y clústeres. Las capacidades no utilizan los recursos de cargas de trabajo de clústeres, no consumen CPU ni memoria en los nodos de trabajo, se escalan automáticamente y tienen un impacto mínimo en la planificación de la capacidad del clúster.

Las soluciones autoadministradas ejecutan controladores y otros componentes en los nodos de trabajo, lo que consume directamente los recursos de los nodos de trabajo, las direcciones IP del clúster y otros recursos del clúster. La administración de los servicios del clúster requiere planificar la capacidad de las cargas de trabajo, así como planificar y configurar las solicitudes y los límites de los recursos para administrar los requisitos de escalabilidad y alta disponibilidad.

Compatibilidad de características

Al tratarse de características del servicio completamente administradas, las capacidades de EKS son, por naturaleza, obstinadas en comparación con las soluciones autoadministradas. Si bien las capacidades admiten la mayoría de las características y los casos de uso, habrá una diferencia en la cobertura en comparación con las soluciones autoadministradas.

Con las soluciones autoadministradas, controla completamente la configuración, las características opcionales y otros aspectos de la funcionalidad del software. Puede optar por ejecutar sus propias imágenes personalizadas, personalizar todos los aspectos de la configuración y controlar completamente la funcionalidad de la solución autoadministrada.

Consideraciones sobre costos

Cada recurso de capacidad de EKS tiene un costo por hora relacionado, que varía según el tipo de capacidad. Los recursos del clúster administrados por la capacidad también tienen un costo por hora asociado con sus propios precios. Para obtener más información, consulte los precios de Amazon EKS.

Las soluciones autoadministradas no tienen costos directos relacionados con los cargos de AWS, pero paga por los recursos de computación en clústeres que utilizan los controladores y las cargas de trabajo relacionadas. Además del consumo de recursos de nodos y clústeres, el costo total de la propiedad de las soluciones autoadministradas incluye los gastos operativos y los gastos de mantenimiento, solución de problemas y soporte.

Elección entre las capacidades de EKS y las soluciones autoadministradas

Capacidades de EKS: tenga en cuenta esta opción si desea reducir los gastos operativos y centrarse en el valor diferenciado de su software y sus sistemas, en lugar de centrarse en las operaciones de plataformas del clúster para cumplir con los requisitos fundamentales. Utilice las capacidades de EKS cuando desee minimizar la carga operativa que suponen los parches de seguridad y la administración del ciclo de vida del software, liberar recursos de nodos y clústeres para las cargas de trabajo de las aplicaciones, simplificar la administración de la configuración y la seguridad y beneficiarse de la cobertura de AWS Support. Las capacidades de EKS son ideales para la mayoría de los casos de uso de producción y son el enfoque recomendado para las nuevas implementaciones.

Soluciones autoadministradas: tenga en cuenta esta opción cuando necesite versiones específicas de la API de recursos de Kubernetes, personalice compilaciones de controladores, tenga la automatización y las herramientas existentes basadas en implementaciones autoadministradas o necesite una personalización profunda de las configuraciones de tiempo de ejecución de los controladores. Las soluciones autoadministradas ofrecen flexibilidad para casos de uso especializados y tiene el control total sobre la configuración de implementación y tiempo de ejecución.

> **Note**
>
> Las capacidades de EKS pueden coexistir en el clúster con soluciones autoadministradas, y es posible llevar a cabo migraciones escalonadas.

Comparaciones específicas de la capacidad

Para obtener comparaciones detalladas, incluidas las características específicas de la capacidad, las diferencias ascendentes y las rutas de migración, consulte:

- the section called “Comparación con autoadministrado”

- the section called “Comparación con autoadministrado”
- the section called “Comparación con autoadministrado”

Implementación de recursos de AWS de Kubernetes con Controladores de AWS para Kubernetes (ACK)

Controladores de AWS para Kubernetes (ACK) le permite definir y administrar los recursos de servicios de AWS directamente desde Kubernetes. Con Controladores de AWS para Kubernetes (ACK), puede administrar los recursos de cargas de trabajo y la infraestructura en la nube mediante los recursos personalizados de Kubernetes, junto con las cargas de trabajo de sus aplicaciones, a través de las API y herramientas de Kubernetes que ya conoce.

Con las capacidades de EKS, AWS administra ACK completamente, lo que elimina la necesidad de instalar, mantener y escalar los controladores de ACK en sus clústeres.

Cómo funciona ACK

ACK traduce las especificaciones de recursos personalizadas de Kubernetes a llamadas a la API de AWS. Cuando crea, actualiza o elimina un recurso personalizado de Kubernetes que represente un recurso de servicio de AWS, ACK hace las llamadas a la API de AWS necesarias para crear, actualizar o eliminar el recurso de AWS.

Cada recurso de AWS compatible con ACK tiene su propia definición de recursos personalizados (CRD) que define el esquema de la API de Kubernetes para especificar su configuración. Por ejemplo, ACK proporciona las CRD para S3, lo que incluye los buckets, las políticas de bucket y otros recursos de S3.

ACK concilia continuamente el estado de sus recursos de AWS con el estado deseado definido en sus recursos personalizados de Kubernetes. Si un recurso se desvía del estado deseado, ACK lo detecta y toma medidas correctivas para volver a alinearlo. Los cambios en los recursos de Kubernetes se reflejan inmediatamente en el estado de los recursos de AWS, mientras que la detección pasiva de las desviaciones y la corrección de los cambios iniciales en los recursos de AWS pueden tardar hasta 10 horas (el periodo de resincronización), pero se suelen producir mucho antes.

Ejemplo de manifiesto de recursos de buckets de S3

```
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
```

```
metadata:
  name: my-ack-bucket
spec:
  name: my-unique-bucket-name
```

Cuando aplique este recurso personalizado a su clúster, ACK crea un bucket de Amazon S3 en su cuenta si aún no existe. Los cambios posteriores en este recurso, como especificar un nivel de almacenamiento no predeterminado o agregar una política, se aplicarán al recurso de S3 en AWS. Cuando se elimina este recurso del clúster, el bucket de S3 de AWS se elimina de forma predeterminada.

Beneficios de ACK

ACK proporciona una administración de recursos de AWS nativa de Kubernetes, lo que le permite administrar los recursos de AWS con las mismas API y herramientas de Kubernetes que usa para sus aplicaciones. Este enfoque unificado simplifica el flujo de trabajo de administración de infraestructuras al eliminar la necesidad de cambiar de una herramienta a otra o aprender a utilizar sistemas independientes de infraestructura como código. Definirá los recursos de AWS de forma declarativa en los manifiestos de Kubernetes, lo que permite que los flujos de trabajo de GitOps y las prácticas de infraestructura como código se integren sin problemas con los procesos de desarrollo existentes.

ACK concilia continuamente el estado deseado de sus recursos de AWS con su estado real, lo que corrige la desviación y garantiza la coherencia en toda la infraestructura. Esta conciliación continua significa que los cambios imperativos en los recursos de AWS sin conexión se revierten automáticamente para que coincidan con la configuración declarada, lo que mantiene la integridad de su infraestructura como código. Puede configurar ACK para administrar los recursos en varias regiones y cuentas de AWS, lo que permite arquitecturas complejas de varias cuentas sin herramientas adicionales.

Para las organizaciones que migran desde otras herramientas de administración de infraestructura, ACK admite la adopción de recursos, lo que le permite administrar los recursos de AWS existentes sin tener que volver a crearlos. ACK también proporciona recursos de solo lectura para la observación de los recursos de AWS sin acceso a modificaciones, y anotaciones para retener los recursos de AWS de forma opcional, incluso cuando el recurso de Kubernetes se elimina del clúster.

Para obtener más información y comenzar a utilizar la capacidad de EKS para ACK, consulte the section called “Conceptos de ACK” y the section called “Consideraciones”.

Servicios de AWS compatibles

ACK admite una amplia gama de servicios de AWS, que incluyen, entre otros, los siguientes:

- Amazon EC2
- Amazon S3
- Amazon RDS
- Amazon DynamoDB
- Amazon ElastiCache
- Amazon EKS
- Amazon SQS
- Amazon SNS
- AWS Lambda
- AWS IAM

Todos los servicios de AWS que figuran como de disponibilidad general ascendente son compatibles con la capacidad de EKS para ACK. Consulte la lista completa de servicios de AWS compatibles para obtener más información.

Integración con otras capacidades administradas de EKS

ACK se integra con otras capacidades administradas de EKS.

- Argo CD: utilice Argo CD para administrar la implementación de los recursos de ACK en varios clústeres, lo que activa los flujos de trabajo de GitOps para su infraestructura de AWS.
  - ACK amplía los beneficios de GitOps cuando se combina con ArgoCD, pero ACK no requiere la integración con git.
- kro (Kube Resource Orchestrator): utilice kro para componer recursos complejos a partir de recursos de ACK, lo que crea abstracciones de alto nivel que simplifican la administración de recursos.
  - Puede crear recursos personalizados compuestos con kro que definan tanto los recursos como los de Kubernetes como los de AWS. Los miembros del equipo pueden usar estos recursos personalizados para implementar rápidamente aplicaciones complejas.

Introducción a ACK

Para comenzar a utilizar la capacidad de EKS para ACK:

1. Cree y configure un rol de capacidad de IAM con los permisos necesarios para que ACK administre los recursos de AWS en su nombre.
2. Cree un recurso de capacidad de ACK en su clúster de EKS a través de la consola de AWS, la AWS CLI o su infraestructura preferida como herramienta de código.
3. Aplique los recursos personalizados de Kubernetes al clúster para comenzar a administrar sus recursos de AWS en Kubernetes.

Creación de una capacidad de ACK

En este capítulo, se explica cómo crear una capacidad de ACK en un clúster de Amazon EKS.

Requisitos previos

Antes de crear una capacidad de ACK, asegúrese de que disponga de lo siguiente:

- Un clúster de Amazon EKS
- Un rol de capacidad de IAM con permisos para que ACK administre los recursos de AWS
- Permisos de IAM suficientes para crear recursos de capacidad en los clústeres de EKS
- La herramienta de la CLI adecuada instalada y configurada o el acceso a la consola de EKS

Para obtener instrucciones sobre cómo crear el rol de capacidad de IAM, consulte the section called “Rol de IAM de capacidad”.

> **Important**
>
> ACK es una capacidad de administración de infraestructuras que permite crear, modificar y eliminar recursos de AWS. Se trata de una capacidad de ámbito administrativo que debe controlarse cuidadosamente. Cualquier persona que tenga permiso para crear recursos de Kubernetes en el clúster puede crear recursos de AWS de forma eficaz mediante ACK, siempre que se cumplan los permisos del rol de capacidad de IAM. El rol de capacidad de IAM que proporcione determina qué recursos de AWS puede crear y administrar ACK. Para obtener orientación sobre cómo crear un rol adecuado con los permisos de privilegio mínimo,

consulte the section called “Rol de IAM de capacidad” y the section called “Consideraciones para las capacidades de EKS”.

Elección de la herramienta

Puede crear una capacidad de ACK mediante la Consola de administración de AWS, la AWS CLI o eksctl:

- the section called “Consola”: uso de la consola para una experiencia guiada
- the section called “ AWS CLI”: uso de la AWS CLI para scripts y automatización
- the section called “eksctl”: uso de eksctl para una experiencia nativa de Kubernetes

Qué ocurre cuando se crea una capacidad de ACK

Al crear una capacidad de ACK:

1. EKS crea el servicio de capacidad de ACK y lo configura para supervisar y administrar los recursos del clúster.
2. Las definiciones de recursos personalizados (CRD) de Kubernetes se instalan en el clúster.
3. La capacidad asume el rol de capacidad de IAM que proporcione.
4. ACK comienza a supervisar sus recursos personalizados en el clúster.
5. El estado de la capacidad cambia de `CREATING` a `ACTIVE`.

Una vez activos, puede crear recursos personalizados de ACK en el clúster para administrar recursos de AWS.

Siguientes pasos

Después de crear la capacidad de ACK:

- the section called “Conceptos de ACK”: descripción de los conceptos de ACK e introducción a los recursos de AWS
- the section called “Conceptos de ACK”: más información sobre la conciliación, las exportaciones de campos y los patrones de adopción de recursos
- the section called “Configuración de permisos”: configuración de los permisos de IAM y los patrones de varias cuentas

Creación de una capacidad de ACK mediante la consola

En este tema, se describe cómo crear una capacidad de Controladores de AWS para Kubernetes (ACK) mediante la Consola de administración de AWS.

Creación de la capacidad de ACK

1. Abra la consola de Amazon EKS en https://console.aws.amazon.com/eks/home#/clusters.
2. Seleccione el nombre del clúster para abrir la página de detalles del clúster.
3. Elija la pestaña Capacidades.
4. En el menú de navegación de la izquierda, elija Controladores de AWS para Kubernetes (ACK).
5. Elija Crear capacidad de Controladores de AWS para Kubernetes.
6. Para el rol de capacidad de IAM:
   - Si ya tiene un rol de capacidad de IAM, selecciónelo en el menú desplegable.
   - Si necesita crear un rol, elija Crear rol de administrador.

     De este modo, se abre la consola de IAM en una nueva pestaña con la política de confianza rellenada previamente y la política administrada de `AdministratorAccess`. Si lo prefiere, puede anular la selección de esta política y agregar otros permisos.

     Después de crear el rol, regrese a la consola de EKS y el rol se seleccionará automáticamente.

     > **Important**
     >
     > La política `AdministratorAccess` sugerida otorga amplios permisos y tiene por objeto simplificar el inicio. Para uso en producción, reemplácela por una política personalizada que otorgue solo los permisos necesarios para los servicios de AWS específicos que tiene previsto administrar con ACK. Para obtener orientación sobre la creación de políticas de privilegio mínimo, consulte the section called “Configuración de permisos” y the section called “Consideraciones para las capacidades de EKS”.
7. Seleccione Crear.

Comenzará el proceso de creación de la capacidad.

Comprobación de la activación de la capacidad

1. En la pestaña Capacidades, consulte el estado de la capacidad de ACK.

2. Espere a que el estado cambie de `CREATING` a `ACTIVE`.
3. Una vez activa, la capacidad está lista para usarse.

Para obtener información sobre los estados de la capacidad y la solución de problemas, consulte the section called “Uso de capacidades”.

Comprobación de la disponibilidad de los recursos personalizados

Una vez que la capacidad esté activa, compruebe que los recursos personalizados de ACK estén disponibles en el clúster.

Uso de la consola

1. Vaya a su clúster en la consola de Amazon EKS.
2. Elija la pestaña Recursos.
3. Elija Extensiones.
4. Elija CustomResourceDefinitions.

Deberías ver varias CRD en la lista de recursos de AWS.

Uso de kubectl

```
kubectl api-resources | grep services.k8s.aws
```

Deberías ver varias API en la lista de recursos de AWS.

> **Note**
>
> La capacidad de Controladores de AWS para Kubernetes permite instalar varios CRD para distintos recursos de AWS.

Siguientes pasos

- the section called “Conceptos de ACK”: descripción de los conceptos de ACK e introducción
- the section called “Configuración de permisos”: configuración de los permisos de IAM para otros servicios de AWS
- the section called “Uso de capacidades”: administración del recurso de la capacidad de ACK

Creación de una capacidad de ACK mediante la AWS CLI

En este tema, se describe cómo crear una capacidad de Controladores de AWS para Kubernetes (ACK) mediante la AWS CLI.

Requisitos previos

- AWS CLI: versión `2.12.3` o posterior. Para comprobar la versión, ejecute `aws --version`. Para obtener más información, consulte Instalación en la Guía del usuario de la interfaz de la línea de comandos de AWS.
- **`kubectl`** – una herramienta de línea de comandos para trabajar con clústeres de Kubernetes. Para obtener más información, consulte the section called “Configure kubectl y eksctl”.

Paso 1: creación de un rol de capacidad de IAM

Cree un archivo de política de confianza:

```
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
```

Cree el rol de IAM:

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

Adjunte la política administrada de `AdministratorAccess` al rol:

```
aws iam attach-role-policy \
  --role-name ACKCapabilityRole \
  --policy-arn arn:aws:iam::aws:policy/AdministratorAccess
```

> **Important**
>
> La política `AdministratorAccess` sugerida otorga amplios permisos y tiene por objeto simplificar el inicio. Para uso en producción, reemplácela por una política personalizada que otorgue solo los permisos necesarios para los servicios de AWS específicos que tiene previsto administrar con ACK. Para obtener orientación sobre la creación de políticas de privilegio mínimo, consulte the section called “Configuración de permisos” y the section called “Consideraciones para las capacidades de EKS”.

Paso 2: creación de la capacidad de ACK

Cree el recurso de la capacidad de ACK en su clúster. Reemplace *`region-code`* por la región de AWS donde creó el clúster y *`my-cluster`* por el nombre de su clúster.

```
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
```

El comando devuelve una respuesta inmediatamente, pero la capacidad tarda algún tiempo en activarse mientras EKS crea la infraestructura y los componentes de la capacidad necesarios. EKS instalará las definiciones de recursos personalizados de Kubernetes relacionadas con esta capacidad en el clúster según se vaya creando.

> **Note**
>
> Si recibe un error que indica que el clúster no existe o que no tiene permisos, compruebe lo siguiente:

- El nombre del clúster es correcto
- La AWS CLI está configurada para la región correcta
- Dispone de los permisos de IAM necesarios

Paso 3: comprobación de la activación de la capacidad

Espere a que se active la capacidad. Reemplace *region-code* por la región de AWS donde creó el clúster y *my-cluster* por el nombre de su clúster.

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-ack \
  --query 'capability.status' \
  --output text
```

La capacidad estará lista cuando aparezca el estado `ACTIVE`. No continúe con el paso siguiente hasta que el estado sea `ACTIVE`.

También puede ver todos los detalles de la capacidad:

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-ack
```

Paso 4: comprobación de la disponibilidad de los recursos personalizados

Una vez que la capacidad esté activa, compruebe que los recursos personalizados de ACK estén disponibles en el clúster:

```
kubectl api-resources | grep services.k8s.aws
```

Deberías ver varias API en la lista de recursos de AWS.

> **Note**
>
> La capacidad de Controladores de AWS para Kubernetes permite instalar varios CRD para distintos recursos de AWS.

Siguientes pasos

- the section called “Conceptos de ACK”: descripción de los conceptos de ACK e introducción
- the section called “Configuración de permisos”: configuración de los permisos de IAM para otros servicios de AWS
- the section called “Uso de capacidades”: administración del recurso de la capacidad de ACK

Creación de una capacidad de ACK mediante eksctl

En este tema, se describe cómo crear una capacidad de Controladores de AWS para Kubernetes (ACK) mediante eksctl.

> **Note**
>
> Los siguientes pasos requieren la versión `0.220.0` o posterior de eksctl. Para comprobar la versión, ejecute `eksctl version`.

Paso 1: creación de un rol de capacidad de IAM

Cree un archivo de política de confianza:

```
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
```

Cree el rol de IAM:

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

Adjunte la política administrada de `AdministratorAccess` al rol:

```
aws iam attach-role-policy \
  --role-name ACKCapabilityRole \
  --policy-arn arn:aws:iam::aws:policy/AdministratorAccess
```

Important

La política `AdministratorAccess` sugerida otorga amplios permisos y tiene por objeto simplificar el inicio. Para uso en producción, reemplácela por una política personalizada que otorgue solo los permisos necesarios para los servicios de AWS específicos que tiene previsto administrar con ACK. Para obtener orientación sobre la creación de políticas de privilegio mínimo, consulte the section called “Configuración de permisos” y the section called “Consideraciones para las capacidades de EKS”.

Important

Esta política otorga permisos para la administración de buckets de S3 con `"Resource": "*"`, lo que permite llevar a cabo operaciones en todos los buckets de S3.
Para uso en producción: * Restrinja el campo `Resource` a patrones de nombres o ARN de buckets específicos. * Use claves de condición de IAM para limitar el acceso por etiquetas de recursos. * Otorgue solo los permisos mínimos necesarios para su caso de uso.
Para otros servicios de AWS, consulte the section called “Configuración de permisos”.

Asocie la política al rol:

```
aws iam attach-role-policy \
  --role-name ACKCapabilityRole \
  --policy-arn arn:aws:iam::$(aws sts get-caller-identity --query Account --output
 text):policy/ACKS3Policy
```

Paso 2: creación de la capacidad de ACK

Cree la capacidad de ACK mediante eksctl Reemplace *region-code* por la región de AWS donde creó el clúster y *my-cluster* por el nombre de su clúster.

```
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**
>
> La etiqueta `--ack-service-controllers` es opcional. Si se omite, ACK activa todos los controladores disponibles. Para mejorar el rendimiento y la seguridad, considere la posibilidad de activar solo los controladores que necesite. Puede especificar varios controladores: `--ack-service-controllers s3,rds,dynamodb`

El comando vuelve inmediatamente, pero la capacidad tarda algún tiempo en activarse.

Paso 3: comprobación de la activación de la capacidad

Compruebe el estado de la capacidad:

```
eksctl get capability \
  --cluster [.replaceable]`my-cluster` \
  --region [.replaceable]`region-code` \
  --name ack
```

La capacidad estará lista cuando aparezca el estado `ACTIVE`.

Paso 4: comprobación de la disponibilidad de los recursos personalizados

Una vez que la capacidad esté activa, compruebe que los recursos personalizados de ACK estén disponibles en el clúster:

```
kubectl api-resources | grep services.k8s.aws
```

Deberías ver varias API en la lista de recursos de AWS.

> **Note**
>
> La capacidad de Controladores de AWS para Kubernetes permite instalar varios CRD para distintos recursos de AWS.

Siguientes pasos

- the section called “Conceptos de ACK”: descripción de los conceptos de ACK e introducción
- the section called “Configuración de permisos”: configuración de los permisos de IAM para otros servicios de AWS
- the section called “Uso de capacidades”: administración del recurso de la capacidad de ACK

Conceptos de ACK

ACK administra los recursos de AWS a través de las API de Kubernetes mediante la conciliación continua del estado deseado de los manifiestos con el estado real en AWS. Al crear o actualizar un recurso personalizado de Kubernetes, ACK hace las llamadas a la API de AWS necesarias para crear o modificar el recurso de AWS correspondiente, lo supervisa para detectar posibles desviaciones y actualiza el estado de Kubernetes para reflejar el estado actual. Este enfoque le permite administrar la infraestructura con herramientas y flujos de trabajo de Kubernetes que ya conoce y, al mismo tiempo, mantener la coherencia entre el clúster y AWS.

En este tema se explican los conceptos fundamentales de cómo ACK administra los recursos de AWS a través de las API de Kubernetes.

Introducción a ACK

Tras crear la capacidad de ACK (consulte the section called “Creación de una capacidad de ACK”), puede comenzar a administrar los recursos de AWS mediante los manifiestos de Kubernetes en su clúster.

Por ejemplo, cree este manifiesto de bucket de S3 en `bucket.yaml` y elija su propio nombre de bucket exclusivo.

```
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: my-test-bucket
  namespace: default
spec:
  name: my-unique-bucket-name-12345
```

Aplique el manifiesto:

```
kubectl apply -f bucket.yaml
```

Compruebe el estado:

```
kubectl get bucket my-test-bucket
kubectl describe bucket my-test-bucket
```

Verifique que el bucket se haya creado en AWS:

```
aws s3 ls | grep my-unique-bucket-name-12345
```

Elimine el recurso de Kubernetes:

```
kubectl delete bucket my-test-bucket
```

Compruebe que el bucket se haya eliminado de AWS:

```
aws s3 ls | grep my-unique-bucket-name-12345
```

El bucket ya no debería aparecer en la lista, lo que demuestra que ACK administra todo el ciclo de vida de los recursos de AWS.

Para obtener más información sobre cómo comenzar a utilizar Athena, consulte Introducción a ACK.

Ciclo de vida de los recursos y conciliación

ACK utiliza un ciclo de conciliación continuo para garantizar que sus recursos de AWS coincidan con el estado deseado definido en los manifiestos de Kubernetes.

Cómo funciona la conciliación:

1. Cree o actualice un recurso personalizado de Kubernetes (por ejemplo, un bucket de S3).
2. ACK detecta el cambio y compara el estado deseado con el estado real en AWS.
3. Si son diferentes, ACK hace llamadas a la API de AWS para conciliar la diferencia.
4. ACK actualiza el estado del recurso en Kubernetes para reflejar el estado actual.
5. El ciclo se repite de forma continua, normalmente cada pocas horas.

La conciliación se activa cuando se crea un nuevo recurso de Kubernetes, se actualiza la `spec` de un recurso existente o cuando ACK detecta una desviación en AWS debido a cambios manuales hechos fuera de ACK. Además, ACK lleva a cabo una conciliación periódica con un periodo de resincronización de 10 horas. Los cambios en los recursos de Kubernetes activan una conciliación inmediata, mientras que la detección pasiva de los cambios en los recursos de AWS ascendentes se produce durante la resincronización periódica.

Al trabajar en el ejemplo de introducción anterior, ACK lleva a cabo los siguientes pasos:

1. Comprueba si el bucket existe en AWS.
2. Si no, llama a `s3:CreateBucket`.
3. Actualiza el estado de Kubernetes con el ARN y el estado del bucket.
4. Continúa supervisando para detectar desviaciones.

Para obtener más información sobre cómo funciona ACK, consulte ACK Reconciliation.

Condiciones de estado

Los recursos de ACK utilizan las condiciones de estado para comunicar su estado. Comprender estas condiciones lo ayuda a solucionar problemas y entender el estado de los recursos.

- Listo: indica que el recurso está listo para consumirse (condición estandarizada de Kubernetes).
- ACK.ResourceSynced: indica que la especificación del recurso coincide con el estado del recurso de AWS.
- ACK.Terminal: indica que se ha producido un error irrecuperable.
- ACK.Adopted: indica que el recurso se adoptó de un recurso de AWS existente en lugar de crearse.
- ACK.Recoverable: indica un error recuperable que puede resolverse sin actualizar la especificación.
- ACK.Advisory: proporciona información de asesoramiento sobre el recurso.
- ACK.LateInitialized: indica si se ha completado la inicialización tardía de los campos.
- ACK.ReferencesResolved: indica si se han resuelto todos los campos `AWSResourceReference`.
- ACK.IAMRoleSelected: indica si se ha seleccionado un IAMRoleSelector para administrar este recurso.

Compruebe el estado del recurso:

```
# 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")]}'
```

Ejemplo de estado:

```
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
```

Para obtener más información sobre los estados y las condiciones de ACK, consulte ACK Conditions.

Políticas de eliminación

La política de eliminación de ACK controla lo que ocurre con los recursos de AWS cuando se elimina el recurso de Kubernetes.

Eliminación (opción predeterminada)

El recurso de AWS se elimina al eliminar el recurso de Kubernetes: este es el comportamiento predeterminado.

```
# No annotation needed - this is the default
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: temp-bucket
spec:
  name: temporary-bucket
```

Al eliminar este recurso, se elimina el bucket de S3 en AWS.

Retain

El recurso de AWS se conserva al eliminar el recurso de Kubernetes:

```
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: important-bucket
  annotations:
    services.k8s.aws/deletion-policy: "retain"
spec:
  name: production-data-bucket
```

Al eliminar este recurso, se elimina de Kubernetes, pero se deja el bucket de S3 en AWS.

La política `retain` es útil para las bases de datos de producción que deberían durar más tiempo que el recurso de Kubernetes, los recursos compartidos utilizados por varias aplicaciones, los recursos con datos importantes que no deberían eliminarse accidentalmente o la administración temporal de ACK donde se adopta un recurso, se configura y, a continuación, se vuelve a administrar manualmente.

Para obtener más información sobre la política de eliminación de ACK, consulte ACK Deletion Policy.

Adopción de recursos

La adopción le permite administrar los recursos de AWS existentes sin necesidad de volver a crearlos.

Cuándo se utiliza la adopción:

- Para la migración de la infraestructura existente a la administración de ACK
- Para la recuperación de recursos de AWS huérfanos en caso de eliminación accidental de recursos en Kubernetes
- Para la importación de recursos creados por otras herramientas (CloudFormation, Terraform)

Cómo funciona la adopción:

```
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
```

Al crear este recurso:

1. ACK comprueba si existe un bucket con ese nombre en AWS.
2. Si lo encuentra, ACK lo adopta (no es necesario crear llamadas a la API).
3. ACK lee la configuración actual de AWS.
4. ACK actualiza el estado de Kubernetes para reflejar el estado actual,
5. Las actualizaciones futuras concilian el recurso con normalidad.

Una vez adoptados, los recursos se administran como cualquier otro recurso de ACK y, al eliminar el recurso de Kubernetes, se eliminará el recurso de AWS, a menos que utilice la política de eliminación `retain`.

Al adoptar un recurso, el recurso de AWS debe existir previamente y ACK necesita permisos de lectura para detectarlo. La política `adopt-or-create` adopta el recurso si existe o lo crea si no existe. Esto resulta útil cuando se desea un flujo de trabajo declarativo que funcione independientemente de si el recurso existe o no.

Para obtener más información sobre la adopción de recursos de ACK, consulte ACK Resource Adoption.

Recursos entre cuentas y regiones

ACK puede administrar los recursos en diferentes cuentas y regiones de AWS desde un solo clúster.

Anotaciones de recursos entre regiones

Puede especificar la región de un recurso de AWS mediante una anotación:

```
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
```

También puede especificar la región de todos los recursos de AWS creados en un espacio de nombres determinado:

Anotaciones de espacios de nombres

Establezca una región predeterminada para todos los recursos de un espacio de nombres:

```
apiVersion: v1
kind: Namespace
metadata:
  name: production
```

```
  annotations:
    services.k8s.aws/default-region: us-west-2
```

Los recursos creados en este espacio de nombres utilizan esta región a menos que se sustituya por una anotación de recurso.

Entre cuentas

Utilice los selectores de roles de IAM para asignar roles de IAM específicos a espacios de nombres:

```
apiVersion: services.k8s.aws/v1alpha1
kind: IAMRoleSelector
metadata:
  name: target-account-config
spec:
  arn: arn:aws:iam::444455556666:role/ACKTargetAccountRole
  namespaceSelector:
    names:
      - production
```

Los recursos creados en el espacio de nombres asignado utilizan automáticamente el rol especificado.

Para obtener más información sobre los selectores de roles de IAM, consulte ACK Cross-Account Resource Management. Para obtener información sobre la configuración entre cuentas, consulte the section called “Configuración de permisos”.

Gestión de errores y comportamiento de los reintentos

ACK gestiona automáticamente los errores transitorios y reintenta las operaciones fallidas.

Estrategia de reintento:

- Los errores transitorios (limitación de velocidad, problemas temporales del servicio, permisos insuficientes) provocan reintentos automáticos.
- El retroceso exponencial evita que las API de AWS se sobrecarguen.
- El número máximo de reintentos varía según el tipo de error.
- Los errores permanentes (parámetros no válidos, conflictos en el nombre del recurso) no se vuelven a intentar.

Compruebe el estado del recurso para ver los detalles del error mediante `kubectl describe`:

```
kubectl describe bucket my-bucket
```

Busque las condiciones de estado con mensajes de error, los eventos que muestren los intentos de conciliación recientes y el campo `message` en las condiciones de estado que explique los errores. Los errores más comunes son la insuficiencia de permisos de IAM, los conflictos en los nombres de los recursos en AWS, los valores de configuración no válidos en la `spec` y la superación de AWS Service Quotas.

Para solucionar errores comunes, consulte the section called “Solución de problemas”.

Composición de recursos con kro

Para componer y conectar varios recursos de ACK entre sí, utilice la capacidad de EKS para kro (Kube Resource Orchestrator). kro proporciona una forma declarativa de definir grupos de recursos, lo que transfiere la configuración entre recursos para administrar patrones de infraestructura complejos de forma sencilla.

Para ver ejemplos detallados de cómo crear composiciones de recursos personalizados con recursos de ACK, consulte the section called “Conceptos de kro”.

Siguientes pasos

- the section called “Consideraciones”: patrones y estrategias de integración específicos de EKS

Configuración de los permisos de ACK

ACK necesita permisos de IAM para crear y administrar recursos de AWS en su nombre. En este tema, se explica cómo funciona IAM con ACK y se proporciona orientación sobre la configuración de los permisos para distintos casos de uso.

Cómo funciona IAM con ACK

ACK usa roles de IAM para autenticarse con AWS y llevar a cabo acciones en los recursos. Hay dos formas de proporcionar permisos a ACK:

Rol de capacidad: el rol de IAM que proporciona al crear la capacidad de ACK. Este rol se utiliza de forma predeterminada para todas las operaciones de ACK.

Selectores de roles de IAM: roles de IAM adicionales que se pueden asignar a espacios de nombres o recursos específicos. Estos roles anulan el rol de capacidad para los recursos de su ámbito.

Cuando ACK necesita crear o administrar un recurso, determina qué rol de IAM se utiliza:

1. Compruebe si un IAMRoleSelector coincide con el espacio de nombres del recurso.
2. Si hay una coincidencia, asuma el rol de IAM.
3. De lo contrario, use el rol de capacidad.

Este enfoque permite una administración flexible de los permisos, desde configuraciones simples de un solo rol hasta configuraciones complejas de varias cuentas y varios equipos.

Introducción: configuración sencilla de permisos

Para el desarrollo, las pruebas o los casos de uso sencillos, puede agregar todos los permisos del servicio necesarios directamente al rol de capacidad.

Este enfoque funciona bien cuando:

- Comienza a usar ACK.
- Todos los recursos están en la misma cuenta de AWS.
- Un solo equipo administra todos los recursos de ACK.
- Confía en que todos los usuarios de ACK tengan los mismos permisos.

Práctica recomendada de producción: selectores de roles de IAM

Para los entornos de producción, utilice los selectores de roles de IAM para implementar el acceso con privilegio mínimo y el aislamiento de espacios de nombres.

Cuando se utilizan los selectores de roles de IAM, el rol de capacidad solo necesita el permiso `sts:AssumeRole` para asumir los roles específicos del servicio. No es necesario agregar ningún permiso de servicio de AWS (como S3 o RDS) al propio rol de capacidad, ya que esos permisos se conceden a los roles de IAM individuales que asume el rol de capacidad.

Cómo elegir entre los modelos de permisos:

Utilice los permisos directos (adición de permisos de servicio al rol de capacidad) cuando:

- Esté comenzando y desee la configuración más sencilla.
- Todos los recursos estén en la misma cuenta que el clúster.
- Tenga requisitos de permisos administrativos para todo el clúster.
- Todos los equipos puedan compartir los mismos permisos.

Utilice los selectores de roles de IAM cuando:

- Administre recursos en varias cuentas de AWS.
- Los diferentes equipos o espacios de nombres necesiten permisos diferentes.
- Necesite un control de acceso detallado por espacio de nombres.
- Desee seguir las prácticas de seguridad de privilegio mínimo.

Puede comenzar con permisos directos y migrar a los selectores de roles de IAM más adelante, a medida que aumenten sus necesidades.

Por qué debería utilizar los selectores de roles de IAM en producción:

- Privilegio mínimo: cada espacio de nombres obtiene solo los permisos que necesita.
- Aislamiento del equipo: el equipo A no puede usar los permisos del equipo B accidentalmente.
- Auditoría más sencilla: se asigna claramente el espacio de nombres que utiliza cada rol.
- Compatibilidad entre cuentas: es necesario para administrar los recursos en varias cuentas.
- Separación de preocupaciones: los diferentes servicios o entornos utilizan roles diferentes.

Configuración básica del selector de roles de IAM

Paso 1: creación de un rol de IAM específico del servicio

Cree un rol de IAM con permisos para servicios de AWS específicos:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:*"
```

```
      ],
      "Resource": "*"
    }
  ]
}
```

Configure la política de confianza para permitir que el rol de capacidad la asuma:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/ACKCapabilityRole"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Paso 2: concesión del permiso AssumeRole al rol de capacidad

Agregue el permiso al rol de capacidad para asumir el rol específico del servicio:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "arn:aws:iam::111122223333:role/ACK-S3-Role"
    }
  ]
}
```

Paso 3: creación de IAMRoleSelector

Asigne el rol de IAM a un espacio de nombres:

```
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
```

Paso 4: creación de recursos en el espacio de nombres asignado

Los recursos del espacio de nombres `s3-resources` utilizan automáticamente el rol especificado:

```
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: my-bucket
  namespace: s3-resources
spec:
  name: my-production-bucket
```

Administración multicuenta

Utilice los selectores de roles de IAM para administrar los recursos de varias cuentas de AWS.

Paso 1: creación de un rol de IAM entre cuentas

En la cuenta de destino (444455556666), cree un rol que confíe en el rol de capacidad de la cuenta de origen:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/ACKCapabilityRole"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Adjunte permisos específicos del servicio a este rol.

Paso 2: concesión del permiso AssumeRole

En la cuenta de origen (111122223333), permita que el rol de capacidad asuma el rol de cuenta de destino:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "arn:aws:iam::444455556666:role/ACKTargetAccountRole"
    }
  ]
}
```

Paso 3: creación de IAMRoleSelector

Asigne el rol de acceso entre cuentas a un espacio de nombres:

```
apiVersion: services.k8s.aws/v1alpha1
kind: IAMRoleSelector
metadata:
  name: production-account-config
spec:
  arn: arn:aws:iam::444455556666:role/ACKTargetAccountRole
  namespaceSelector:
    names:
      - production
```

Paso 4: creación de recursos

Los recursos del espacio de nombres `production` se crean en la cuenta de destino:

```
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: my-bucket
  namespace: production
spec:
```

```
  name: my-cross-account-bucket
```

Patrones avanzados del selector de roles de IAM

Para obtener la configuración avanzada, que incluye selectores de roles, asignación de roles específica del recurso y ejemplos adicionales, consulte la documentación sobre IRSA de ACK.

Siguientes pasos

- the section called “Conceptos de ACK”: descripción de los conceptos de ACK y el ciclo de vida de los recursos
- the section called “Conceptos de ACK”: información sobre las políticas de adopción y eliminación de recursos
- the section called “Consideraciones para las capacidades de EKS”: descripción de las prácticas recomendadas de seguridad para capacidades

Consideraciones sobre ACK para EKS

En este tema, se tratan aspectos importantes al utilizar la capacidad de EKS para ACK, como la configuración de IAM, los patrones de varias cuentas y la integración con otras capacidades de EKS.

Patrones de configuración de IAM

La capacidad de ACK utiliza un rol de capacidad de IAM para autenticarse con AWS. Elija el patrón de IAM adecuado en función de sus requisitos.

Sencillo: rol de capacidad único

Para el desarrollo, las pruebas o los casos de uso sencillos, conceda todos los permisos necesarios directamente al rol de capacidad.

Cuándo se debe usar:

- Introducción a ACK
- Implementaciones de una sola cuenta
- Todos los recursos administrados por un equipo
- Entornos de desarrollo y pruebas

Ejemplo: agregue permisos de S3 y RDS al rol de capacidad con las condiciones de etiquetado de recursos.

```
{
  "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"]
        },
      }
    }
  ]
}
```

Este ejemplo limita las operaciones de S3 y RDS a regiones específicas y requiere que los recursos de RDS tengan una etiqueta `ManagedBy: ACK`.

Producción: selectores de roles de IAM

Para los entornos de producción, utilice los selectores de roles de IAM para implementar el acceso con privilegio mínimo y el aislamiento de espacios de nombres.

Cuándo se debe usar:

- Entornos de producción
- Clústeres de varios equipos
- Administración de recursos con varias cuentas

- Requisitos de seguridad de privilegio mínimo
- Diferentes servicios que necesitan diferentes permisos

Ventajas:

- Cada espacio de nombres obtiene solo los permisos que necesita.
- Aislamiento del equipo: el equipo A no puede usar los permisos del equipo B
- Auditoría y conformidad más fáciles
- Necesario para la administración de recursos entre cuentas

Para obtener información sobre la configuración de selectores de roles de IAM, consulte the section called “Configuración de permisos”.

Integración con otras capacidades de EKS

GitOps con Argo CD

Utilice la capacidad de EKS para Argo CD para implementar recursos de ACK desde repositorios de Git, lo que permite usar los flujos de trabajo de GitOps para la administración de infraestructuras.

Consideraciones:

- Almacene recursos de ACK junto con manifiestos de aplicaciones para GitOps integrales.
- Organice por entorno, servicio o tipo de recurso según la estructura de su equipo.
- Utilice la sincronización automática de Argo CD para una conciliación continua.
- Active la poda para eliminar automáticamente los recursos eliminados.
- Tenga en cuenta los patrones radiales para la administración de infraestructuras de varios clústeres.

GitOps proporciona registros de auditoría, capacidades de reversión y administración declarativa de infraestructuras. Para obtener más información sobre Argo CD, consulte the section called “Uso de Argo CD”.

Composición de recursos con kro

Utilice la capacidad de EKS para kro (Kube Resource Orchestrator) para componer varios recursos de ACK en abstracciones de nivel superior y API personalizadas.

Cuándo se usa kro con ACK:

- Para la creación de patrones reutilizables para pilas de infraestructura comunes (base de datos + copias de seguridad + supervisión)
- Para la creación de plataformas de autoservicio con API simplificadas para los equipos de aplicaciones
- Para la administración de dependencias de los recursos y para la transferencia de valores entre recursos (de ARN de bucket de S3 a función de Lambda)
- Para la estandarización de las configuraciones de la infraestructura en todos los equipos
- Para la reducción de la complejidad mediante la ocultación de los detalles de la implementación detrás de recursos personalizados

Ejemplos de patrones:

- Pila de la aplicación: bucket de S3 + cola de SQS + configuración de notificaciones
- Configuración de la base de datos: instancia de RDS + grupo de parámetros + grupo de seguridad + secretos
- Redes: VPC + subredes + tablas de enrutamiento + grupos de seguridad

kro gestiona el orden de las dependencias, la propagación de estados y la administración del ciclo de vida de los recursos compuestos. Para obtener más información sobre kro, consulte the section called “Conceptos de kro”.

Organización de los recursos

Organice los recursos de ACK mediante espacios de nombres de Kubernetes y etiquetas de recursos de AWS para mejorar la administración, el control de acceso y el seguimiento de los costos.

Organización de espacios de nombres

Utilice los espacios de nombres de Kubernetes para separar de forma lógica los recursos de ACK por entorno (producción, prueba, desarrollo), equipo (plataforma, datos, ML) o aplicación.

Ventajas:

- RBAC con ámbito de espacio de nombres para el control de acceso
- Establecimiento de regiones predeterminadas por espacio de nombres mediante anotaciones

- Administración y limpieza de recursos más sencillas
- Separación lógica alineada con la estructura organizativa

Etiquetado de recursos

EKS aplica automáticamente etiquetas a los recursos de AWS administrados por ACK, lo que incluye el ARN del recurso de la capacidad. Agregue etiquetas adicionales para la asignación de costos, el seguimiento de la propiedad y fines organizativos.

Etiquetas recomendadas:

- Entorno (producción, prueba, desarrollo)
- Propiedad del equipo o departamento
- Centro de costos para la asignación de la facturación
- Nombre de la aplicación o del servicio
- ManagedBy: ACK (para identificar los recursos administrados por ACK)

Migración desde otras herramientas de infraestructura como código

Muchas organizaciones encuentran valor al estandarizar en Kubernetes más allá de la orquestación de sus cargas de trabajo. Al migrar la administración de la infraestructura y los recursos de AWS a ACK, puede estandarizar la administración de la infraestructura mediante las API de Kubernetes junto con las cargas de trabajo de aplicaciones.

Ventajas de estandarizar en Kubernetes para la infraestructura:

- Único origen de información verdadera: administre las aplicaciones y la infraestructura en Kubernetes, lo que posibilita una práctica de GitOps de extremo a extremo.
- Herramientas unificadas: los equipos utilizan los recursos y las herramientas de Kubernetes en lugar de aprender múltiples herramientas y marcos.
- Conciliación coherente: ACK concilia continuamente los recursos de AWS como lo hace Kubernetes con las cargas de trabajo, mediante la detección y la corrección de las desviaciones en comparación con las herramientas imperativas.
- Composiciones nativas: con kro y ACK juntos, hace referencia a los recursos de AWS directamente en los manifiestos de aplicaciones y recursos, al pasar las cadenas de conexión y los ARN entre los recursos.

- Operaciones simplificadas: dispone de un plano de control para las implementaciones, las reversiones y la observabilidad en todo el sistema.

ACK permite adoptar los recursos de AWS existentes sin volver a crearlos, lo que permite una migración sin tiempo de inactividad desde CloudFormation, Terraform o recursos externos al clúster.

Adopte un recurso existente:

```
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
```

Una vez adoptado, ACK administra el recurso y lo puede actualizar mediante los manifiestos de Kubernetes. Puede migrar de forma gradual: se adoptan los recursos según sea necesario y, al mismo tiempo, se mantienen las herramientas de IaC existentes para otros recursos.

ACK también admite recursos de solo lectura. En el caso de los recursos administrados por otros equipos o herramientas a los que quiera hacer referencia, pero no modificar, combine la adopción con la política de eliminación `retain` y otorgue permisos de IAM de solo lectura. De este modo, las aplicaciones pueden detectar la infraestructura compartida (VPC, roles de IAM, claves de KMS) a través de las API de Kubernetes sin correr el riesgo de sufrir modificaciones.

Para obtener más información sobre la adopción de recursos, consulte the section called “Conceptos de ACK”.

Políticas de eliminación

Las políticas de eliminación controlan lo que ocurre con los recursos de AWS cuando se elimina el recurso de Kubernetes correspondiente. Elija la política adecuada en función del ciclo de vida de los recursos y sus requisitos operativos.

Eliminación (opción predeterminada)

El recurso de AWS se elimina al eliminar el recurso de Kubernetes. De este modo, se mantiene la coherencia entre el clúster y AWS, lo que garantiza que los recursos no se acumulen.

Cuándo se usa la eliminación:

- Entornos de desarrollo y prueba en los que la limpieza es importante
- Recursos efímeros vinculados al ciclo de vida de la aplicación (bases de datos de prueba, buckets temporales)
- Recursos que no deberían durar más que la aplicación (colas de SQS, clústeres de ElastiCache)
- Optimización de costos: limpieza automática de los recursos no utilizados
- Entornos administrados con GitOps en los que la eliminación de recursos de Git debería eliminar la infraestructura

La política de eliminación predeterminada se ajusta al modelo declarativo de Kubernetes: lo que hay en el clúster coincide con lo que hay en AWS.

Retener

El recurso de AWS se conserva al eliminar el recurso de Kubernetes. De este modo, se protegen los datos críticos y se permite que los recursos sobrepasen su tiempo de representación de Kubernetes.

Cuándo se usa la retención:

- Bases de datos de producción con datos críticos que deben sobrevivir a los cambios del clúster
- Buckets de almacenamiento a largo plazo con requisitos de conformidad o auditoría
- Recursos compartidos que utilizan múltiples aplicaciones o equipos
- Recursos que se están migrando a diferentes herramientas de administración
- Escenarios de recuperación ante desastres en los que se desea preservar la infraestructura
- Recursos con dependencias complejas que requieren un desmantelamiento cuidadoso

```
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**
>
> Los recursos retenidos siguen generando costos de AWS y se deben eliminar manualmente de AWS cuando ya no se necesiten. Utilice el etiquetado de recursos a fin de hacer un seguimiento de los recursos retenidos para su limpieza.

Para obtener más información sobre las políticas de eliminación, consulte the section called "Conceptos de ACK".

Documentación ascendente

Para obtener información detallada sobre el uso de ACK:

- Guía de uso de ACK: creación y administración de recursos
- Referencia sobre la API de ACK: documentación completa sobre la API para todos los servicios
- Documentación de ACK: documentación completa para el usuario

Siguientes pasos

- the section called "Configuración de permisos": configuración de los permisos de IAM y los patrones de varias cuentas
- the section called "Conceptos de ACK": descripción de los conceptos de ACK y el ciclo de vida de los recursos
- the section called "Solución de problemas": solución de problemas de ACK
- the section called "Uso de Argo CD": implementación de recursos de ACK con GitOps
- the section called "Conceptos de kro": composición de recursos de ACK en abstracciones de nivel superior

Solución de problemas con capacidades de ACK

En este tema se proporcionan instrucciones para la solución de problemas de la capacidad de EKS para ACK, como las comprobaciones de estado de la capacidad, la verificación del estado de los recursos y los problemas de permisos de IAM.

> **Note**
>
> Las capacidades de EKS son completamente administradas y se ejecutan fuera del clúster. No tiene acceso a los registros ni a los espacios de nombres de los controladores. La solución de problemas se centra en el estado de la capacidad, el estado de los recursos y la configuración de IAM.

La capacidad está ACTIVA, pero no se crean recursos

Si la capacidad de ACK muestra el estado `ACTIVE`, pero no se están creando recursos en AWS, compruebe el estado de la capacidad, el estado de los recursos y los permisos de IAM.

Compruebe el estado de la capacidad:

Puede ver los problemas de estado de la capacidad en la consola de EKS o mediante la AWS CLI.

Consola:

1. Abra la consola de Amazon EKS en https://console.aws.amazon.com/eks/home#/clusters.
2. Seleccione el nombre del clúster.
3. Seleccione la pestaña Observabilidad.
4. Elija Supervisar clúster.
5. Seleccione la pestaña Capacidades para ver el estado de todas las capacidades.

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
```

Causas habituales:

- Faltan permisos de IAM: el rol de capacidad no tiene permisos para el servicio de AWS.

- Espacio de nombres incorrecto: los recursos se crearon en un espacio de nombres sin el IAMRoleSelector adecuado.
- Especificación de recurso no válida: compruebe las condiciones del estado del recurso para ver si hay errores de validación.
- Limitación de la API: se están alcanzando los límites de velocidad de la API de AWS.
- Webhooks de admisión: los webhooks de admisión impiden que el controlador corrija el estado de los recursos.

Compruebe el estado del recurso:

```
# 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
```

Verifique los permisos de IAM:

```
# 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
```

Recursos creados en AWS que no se muestran en Kubernetes

ACK solo hace el seguimiento de los recursos que crea a través de los manifiestos de Kubernetes. Para administrar los recursos de AWS existentes con ACK, use la característica de adopción.

```
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
```

Para obtener más información sobre la adopción de recursos, consulte the section called “Conceptos de ACK”.

No se están creando recursos entre cuentas

Si los recursos no se crean en una cuenta de AWS de destino al utilizar los selectores de roles de IAM, compruebe la relación de confianza y la configuración de IAMRoleSelector.

Verifique la relación de confianza:

```
# Check the trust policy in the target account role
aws iam get-role --role-name cross-account-ack-role --query
 'Role.AssumeRolePolicyDocument'
```

La política de confianza debe permitir que el rol de capacidad de la cuenta de origen la asuma.

Confirme la configuración de IAMRoleSelector:

```
# List IAMRoleSelectors (cluster-scoped)
kubectl get iamroleselector

# Describe specific selector
kubectl describe iamroleselector my-selector
```

Compruebe la alineación del espacio de nombres:

Los IAMRoleSelectors son recursos de ámbito del clúster, pero tienen como destino espacios de nombres específicos. Asegúrese de que sus recursos de ACK estén en un espacio de nombres que coincida con el selector de espacios de nombres de IAMRoleSelector:

```
# 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}'
```

Compruebe la condición de IAMRoleSelected:

Para comprobar que el IAMRoleSelector haya coincidido correctamente con su recurso, compruebe la condición `ACK.IAMRoleSelected`:

```
# Check if IAMRoleSelector was matched
kubectl get bucket my-cross-account-bucket -n production -o
 jsonpath='{.status.conditions[?(@.type=="ACK.IAMRoleSelected")]}'
```

Si la condición es `False` o falta, el selector de espacios de nombres de IAMRoleSelector no coincide con el espacio de nombres del recurso. Compruebe que el `namespaceSelector` del selector coincida con las etiquetas del espacio de nombres del recurso.

Compruebe los permisos del rol de capacidad:

El rol de capacidad necesita el permiso `sts:AssumeRole` para el rol de la cuenta de destino:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "arn:aws:iam::[.replaceable]`444455556666`:role/[.replaceable]`cross-
account-ack-role`"
    }
  ]
}
```

Para obtener información detallada sobre la configuración entre cuentas, consulte the section called “Configuración de permisos”.

Siguientes pasos

- the section called “Consideraciones”: consideraciones y prácticas recomendadas
- the section called “Configuración de permisos”: configuración de los permisos de IAM y los patrones de varias cuentas

- the section called “Conceptos de ACK”: descripción de los conceptos de ACK y el ciclo de vida de los recursos
- the section called “Solución de problemas de capacidades”: orientación general de solución de problemas de la capacidad

Comparación de la capacidad de EKS para ACK con ACK autoadministrados

La capacidad de EKS para ACK proporciona la misma funcionalidad que los controladores ACK autoadministrados, pero con importantes ventajas operativas. Para obtener una comparación general entre las capacidades de EKS y las soluciones autoadministradas, consulte the section called “Consideraciones”. Este tema se centra en las diferencias específicas de ACK.

Diferencias con respecto a ACK ascendentes

La capacidad de EKS para ACK se basa en los controladores ACK ascendentes, pero difiere en la integración de IAM.

Rol de capacidad de IAM: la capacidad utiliza un rol de IAM dedicado con una política de confianza que permite la entidad principal del servicio de `capabilities.eks.amazonaws.com`, pero no IRSA (roles de IAM para cuentas de servicio). Puede adjuntar las políticas de IAM directamente al rol de capacidad sin necesidad de crear ni anotar cuentas de servicio de Kubernetes ni configurar los proveedores de OIDC. Una práctica recomendada para los casos de uso de producción es configurar los permisos del servicio mediante `IAMRoleSelector`. Consulte the section called “Configuración de permisos” para obtener más detalles.

Compatibilidad de recursos: los recursos personalizados de ACK funcionan de forma idéntica a los ACK ascendentes sin que se produzcan cambios en los archivos YAML de los recursos de ACK. La capacidad utiliza las mismas API y CRD de Kubernetes, por lo que herramientas como `kubectl` funcionan de la misma manera. Se admiten todos los controladores y recursos de GA de ACK ascendentes.

Para obtener la documentación completa de ACK y las guías específicas del servicio, consulte la documentación de ACK.

Ruta de migración

Puede migrar de ACK autoadministrados a la capacidad administrada sin tiempo de inactividad:

1. Actualice el controlador de ACK autoadministrados para utilizar `kube-system` en las asignaciones de elección de líderes, por ejemplo:

   ```
   helm upgrade --install ack-s3-controller \
     oci://public.ecr.aws/aws-controllers-k8s/s3-chart \
     --namespace ack-system \
     --set leaderElection.namespace=kube-system
   ```

   De este modo, se traslada la asignación del controlador a `kube-system`, lo que permite que la capacidad administrada se coordine con él.
2. Cree la capacidad ACK en el clúster (consulte the section called “Creación de una capacidad de ACK”).
3. La capacidad administrada reconoce los recursos de AWS administrados por ACK existentes y se encarga de la conciliación.
4. Reduzca verticalmente o elimine de forma gradual las implementaciones de controladores autoadministrados:

   ```
   helm uninstall ack-s3-controller --namespace ack-system
   ```

Este enfoque permite que ambos controladores coexistan de forma segura durante la migración. La capacidad administrada adopta automáticamente los recursos que antes administraban los controladores autoadministrados, lo que garantiza una conciliación continua sin conflictos.

Siguientes pasos

- the section called “Creación de una capacidad de ACK”: creación de un recurso con la capacidad de ACK
- the section called “Conceptos de ACK”: descripción de los conceptos de ACK y el ciclo de vida de los recursos
- the section called “Configuración de permisos”: configuración de IAM y permisos

Implementación continua con Argo CD

Argo CD es una herramienta declarativa de entrega continua de GitOps para Kubernetes. Con Argo CD, puede automatizar la implementación y la administración del ciclo de vida de sus aplicaciones

en múltiples clústeres y entornos. Argo CD admite varios tipos de orígenes, como los repositorios de Git, los registros de Helm (HTTP y OCI) y las imágenes de OCI, lo que proporciona flexibilidad a las organizaciones con diferentes requisitos de seguridad y cumplimiento.

Con las capacidades de EKS, AWS administra Argo CD completamente, lo que elimina la necesidad de instalar, mantener y escalar los controladores de Argo CD y sus dependencias en sus clústeres.

Cómo funciona Argo CD

Argo CD sigue el patrón de GitOps, donde el origen de la aplicación (repositorio de Git, registro de Helm o imagen de OCI) es el origen de información verdadera para definir el estado de la aplicación deseado. Al crear un recurso `Application` de Argo CD, especifique un origen que contenga los manifiestos de la aplicación y un espacio de nombres y un clúster de Kubernetes de destino. Argo CD supervisa de forma continua tanto el origen como el estado activo del clúster y sincroniza automáticamente cualquier cambio para garantizar que el estado del clúster coincida con el estado deseado.

> **Note**
>
> Con la capacidad de EKS para Argo CD, el software de Argo CD se ejecuta en el plano de control de AWS, no en los nodos de trabajo. Esto significa que sus nodos de trabajo no necesitan acceso directo a los repositorios de Git o a los registros de Helm, ya que la capacidad gestiona el acceso al origen desde la cuenta de AWS.

Argo CD proporciona tres tipos de recursos principales:

- Aplicación: define una implementación desde un repositorio de Git a un clúster de destino.
- ApplicationSet: genera múltiples aplicaciones a partir de plantillas para implementaciones de varios clústeres.
- AppProject: proporciona agrupamiento lógico y control de acceso para las aplicaciones.

Ejemplo: creación de una aplicación de Argo CD

En el ejemplo siguiente, se muestra cómo crear un recurso `Application` de Argo CD:

```
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: my-cluster
    namespace: guestbook
  syncPolicy:
    automated:
      prune: true
      selfHeal: true
```

Ventajas de Argo CD

Argo CD implementa un flujo de trabajo de GitOps en el que define las configuraciones de las aplicaciones en los repositorios de Git y Argo CD sincroniza automáticamente las aplicaciones para que coincidan con el estado deseado. Este enfoque centrado en Git proporciona un registro de auditoría completo de todos los cambios, permite revertirlos fácilmente y se integra de forma natural con sus procesos de revisión y aprobación de código existentes. Argo CD detecta y concilia automáticamente la diferencia entre el estado deseado en Git y el estado real de los clústeres, lo que garantiza que las implementaciones se mantengan coherentes con la configuración declarada.

Con Argo CD, puede implementar y administrar aplicaciones en varios clústeres desde una única instancia de Argo CD, lo que simplifica las operaciones en entornos de varios clústeres y regiones. La interfaz de usuario de Argo CD ofrece capacidades de visualización y supervisión, lo que le permite ver el estado de la implementación, el estado y el historial de sus aplicaciones. La interfaz de usuario se integra con AWS Identity Center (anteriormente AWS SSO) para una autenticación y autorización fluidas, lo que le permite controlar el acceso mediante su infraestructura de administración de identidades existente.

Como parte de las capacidades administradas de EKS, AWS administra completamente Argo CD, lo que elimina la necesidad de instalar, configurar y mantener la infraestructura de Argo CD. AWS administra el escalado, la aplicación de parches y la administración operativa, lo que permite a sus equipos centrarse en la entrega de la aplicación y no en el mantenimiento de la herramienta.

Integración con AWS Identity Center

Las capacidades administradas de EKS proporcionan una integración directa entre Argo CD y AWS Identity Center, lo que permite una autenticación y autorización fluidas para sus usuarios. Al activar la capacidad de Argo CD, puede configurar la integración de AWS Identity Center para asignar los grupos y usuarios de Identity Center a los roles de RBAC de Argo CD, lo que le permite controlar quién puede acceder a las aplicaciones de Argo CD y administrarlas.

Integración con otras capacidades administradas de EKS

Argo CD se integra con otras capacidades administradas de EKS.

- Controladores de AWS para Kubernetes (ACK): utilice Argo CD para administrar la implementación de los recursos de ACK en varios clústeres, lo que activa los flujos de trabajo de GitOps para su infraestructura de AWS.
- kro (Kube Resource Orchestrator): utilice Argo CD para implementar composiciones de kro en varios clústeres, lo que permite una composición uniforme de los recursos en todo el entorno de Kubernetes.

Introducción a Argo CD

Para comenzar a utilizar la capacidad de EKS para Argo CD:

1. Cree y configure un rol de capacidad de IAM con los permisos necesarios para que Argo CD acceda a sus repositorios de Git y administre las aplicaciones.
2. Cree un recurso de capacidad de Argo CD en su clúster de EKS a través de la consola de AWS, la AWS CLI o su infraestructura preferida como herramienta de código.
3. Configure el acceso al repositorio y registre los clústeres para la implementación de aplicaciones.
4. Cree recursos de la aplicación para implementar las aplicaciones desde los repositorios de Git.

Creación de una capacidad de Argo CD

En este tema, se explica cómo crear una capacidad de Argo CD en un clúster de Amazon EKS.

Requisitos previos

Antes de crear una capacidad de Argo CD, asegúrese de que disponga de lo siguiente:

- Un clúster de Amazon EKS existente que ejecute una versión de Kubernetes compatible (se admiten todas las versiones con soporte estándar y ampliado)
- AWS Identity Center configurado: necesario para la autenticación de Argo CD (no se admiten los usuarios locales)
- Un rol de capacidad de IAM con permisos para Argo CD
- Permisos de IAM suficientes para crear recursos de capacidad en los clústeres de EKS
- `kubectl` configurado para comunicarse con el clúster
- (Opcional) La CLI de Argo CD instalada para facilitar la administración de clústeres y repositorios
- (Para la CLI o eksctl) La herramienta de la CLI adecuada instalada y configurada

Para obtener instrucciones sobre cómo crear el rol de capacidad de IAM, consulte the section called "Rol de IAM de capacidad". Para obtener la configuración de Identity Center, consulte Introducción a AWS Identity Center.

> **Important**
>
> El rol de capacidad de IAM que proporcione determina a qué recursos de AWS puede acceder Argo CD. Esto incluye el acceso al repositorio de Git a través de CodeConnections y los secretos en Secrets Manager. Para obtener orientación sobre cómo crear un rol adecuado con los permisos de privilegio mínimo, consulte the section called "Rol de IAM de capacidad" y the section called "Consideraciones para las capacidades de EKS".

Elección de la herramienta

Puede crear una capacidad de Argo CD mediante la Consola de administración de AWS, la AWS CLI o eksctl:

- the section called "Consola": uso de la consola para una experiencia guiada
- the section called " AWS CLI": uso de la AWS CLI para scripts y automatización
- the section called "eksctl": uso de eksctl para una experiencia nativa de Kubernetes

Qué ocurre cuando se crea una capacidad de Argo CD

Cuando crea una capacidad de Argo CD:

1. EKS crea el servicio de capacidad de Argo CD y lo configura para supervisar y administrar los recursos del clúster.
2. Las definiciones de recursos personalizados (CRD) de Kubernetes se instalan en el clúster.
3. La capacidad asume el rol de capacidad de IAM que proporcione.
4. Argo CD comienza a supervisar sus recursos personalizados.
5. El estado de la capacidad cambia de `CREATING` a `ACTIVE`.
6. Se puede acceder a la interfaz de usuario de Argo CD a través de su punto de conexión.

Una vez activo, puede crear aplicaciones de Argo CD en el clúster para implementarlas desde los repositorios de Git.

Siguientes pasos

Después de crear la capacidad de Argo CD:

- the section called “Uso de Argo CD”: configuración del acceso a repositorios, registro de clústeres de destino y creación de aplicaciones
- the section called “Conceptos de Argo CD”: más información sobre los principios de GitOps, las políticas de sincronización y los patrones de varios clústeres
- the section called “Consideraciones sobre Argo CD”: exploración de los patrones de arquitectura de varios clústeres y configuración avanzada

Creación de una capacidad de Argo CD mediante la consola

En este tema, se describe cómo crear una capacidad de Argo CD mediante la Consola de administración de AWS.

Requisitos previos

- AWS Identity Center configurado: Argo CD requiere AWS Identity Center para la autenticación. No se admiten usuarios locales. Si no ha configurado AWS Identity Center, consulte Introducción a AWS Identity Center para crear una instancia de Identity Center y Adición de usuarios y Adición de grupos para crear usuarios y grupos a fin de acceder a Argo CD.

Creación de la capacidad de Argo CD

1. Abra la consola de Amazon EKS en https://console.aws.amazon.com/eks/home#/clusters.

2. Seleccione el nombre del clúster para abrir la página de detalles del clúster.
3. Elija la pestaña Capacidades.
4. En el panel de navegación izquierdo, elija Argo CD.
5. Elija Crear capacidad de Argo CD.
6. Para el rol de capacidad de IAM:
   - Si ya tiene un rol de capacidad de IAM, selecciónelo en el menú desplegable.
   - Si necesita crear un rol, elija Crear rol de Argo CD.

     De este modo, se abre la consola de IAM en una nueva pestaña con la política de confianza rellenada previamente y el acceso completo de lectura a Secrets Manager. No se agrega ningún otro permiso de forma predeterminada, pero puede agregarlos si es necesario. Si tiene previsto usar los repositorios de CodeCommit u otros servicios de AWS, agregue los permisos adecuados antes de crear el rol.

     Después de crear el rol, regrese a la consola de EKS y el rol se seleccionará automáticamente.

     > **Note**
     >
     > Si tiene previsto usar las integraciones opcionales con AWS Secrets Manager o AWS CodeConnections, tendrá que agregar permisos al rol. Para ver ejemplos de políticas de IAM y guías de configuración, consulte the section called “ AWS Secrets Manager” y the section called “ AWS CodeConnections”.
7. Configure la integración de AWS Identity Center:
   a. Seleccione Habilitar la integración de AWS Identity Center.
   b. Elija su instancia de Identity Center en el menú desplegable.
   c. Para configurar las asignaciones de roles para RBAC, asigne usuarios o grupos a los roles de Argo CD (ADMIN, EDITOR o VIEWER)
8. Seleccione Crear.

Comenzará el proceso de creación de la capacidad.

Comprobación de la activación de la capacidad

1. En la pestaña Capacidades, consulte el estado de la capacidad de Argo CD.
2. Espere a que el estado cambie de `CREATING` a `ACTIVE`.

3. Una vez activa, la capacidad está lista para usarse.

Para obtener información sobre los estados de la capacidad y la solución de problemas, consulte the section called “Uso de capacidades”.

Acceso a la interfaz de usuario de Argo CD

Después de que la capacidad esté activa, puede acceder a la interfaz de usuario de Argo CD:

1. En la página de la capacidad de Argo CD, seleccione Abrir la interfaz de usuario de Argo CD.
2. La interfaz de usuario de Argo CD se abre en una nueva pestaña del navegador.
3. Ahora puede crear aplicaciones y administrar las implementaciones a través de la interfaz de usuario.

Siguientes pasos

- the section called “Uso de Argo CD”: configuración de repositorios, registro de clústeres y creación de aplicaciones
- the section called “Consideraciones sobre Argo CD”: arquitectura de varios clústeres y configuración avanzada
- the section called “Uso de capacidades”: administración del recurso de la capacidad de Argo CD

Creación de una capacidad de Argo CD mediante la AWS CLI

En este tema, se describe cómo crear una capacidad de Argo CD mediante la AWS CLI.

Requisitos previos

- AWS CLI: versión `2.12.3` o posterior. Para comprobar la versión, ejecute `aws --version`. Para obtener más información, consulte Instalación en la Guía del usuario de la interfaz de la línea de comandos de AWS.
- **`kubectl`** – una herramienta de línea de comandos para trabajar con clústeres de Kubernetes. Para obtener más información, consulte the section called “Configure kubectl y eksctl”.
- AWS Identity Center configurado: Argo CD requiere AWS Identity Center para la autenticación. No se admiten usuarios locales. Si no ha configurado AWS Identity Center, consulte Introducción a AWS Identity Center para crear una instancia de Identity Center y Adición de usuarios y Adición de grupos para crear usuarios y grupos a fin de acceder a Argo CD.

Paso 1: creación de un rol de capacidad de IAM

Cree un archivo de política de confianza:

```
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
```

Cree el rol de IAM:

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

> **Note**
>
> Si tiene previsto usar las integraciones opcionales con AWS Secrets Manager o AWS CodeConnections, tendrá que agregar permisos al rol. Para ver ejemplos de políticas de IAM y guías de configuración, consulte the section called “ AWS Secrets Manager” y the section called “ AWS CodeConnections”.

Paso 2: creación de la capacidad de Argo CD

Cree el recurso de la capacidad de Argo CD en su clúster.

En primer lugar, defina las variables de entorno para la configuración de Identity Center:

```
# 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"
```

Cree la capacidad con la integración de Identity Center. Reemplace *region-code* por la región de AWS en la que se encuentra el clúster y *my-cluster* por el nombre del clúster:

```
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"
        }]
      }]
    }
  }'
```

El comando devuelve una respuesta inmediatamente, pero la capacidad tarda algún tiempo en activarse mientras EKS crea la infraestructura y los componentes de la capacidad necesarios. EKS instalará las definiciones de recursos personalizados de Kubernetes relacionadas con esta capacidad en el clúster según se vaya creando.

> **Note**
>
> Si recibe un error que indica que el clúster no existe o que no tiene permisos, compruebe lo siguiente:
>
> - El nombre del clúster es correcto
> - La AWS CLI está configurada para la región correcta
> - Dispone de los permisos de IAM necesarios

Paso 3: comprobación de la activación de la capacidad

Espere a que se active la capacidad. Reemplace *region-code* por la región de AWS en la que se encuentra el clúster y *my-cluster* por el nombre del clúster.

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-argocd \
  --query 'capability.status' \
  --output text
```

La capacidad estará lista cuando aparezca el estado `ACTIVE`. No continúe con el paso siguiente hasta que el estado sea `ACTIVE`.

También puede ver todos los detalles de la capacidad:

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-argocd
```

Paso 4: comprobación de la disponibilidad de los recursos personalizados

Una vez que la capacidad esté activa, compruebe que los recursos personalizados de Argo CD estén disponibles en el clúster:

```
kubectl api-resources | grep argoproj.io
```

Debería ver todos los tipos de recursos `Application` y `ApplicationSet` en la lista.

Siguientes pasos

- the section called “Uso de Argo CD”: configuración de repositorios, registro de clústeres y creación de aplicaciones
- the section called “Consideraciones sobre Argo CD”: arquitectura de varios clústeres y configuración avanzada
- the section called “Uso de capacidades”: administración del recurso de la capacidad de Argo CD

Creación de una capacidad de Argo CD mediante eksctl

En este tema, se describe cómo crear una capacidad de Argo CD mediante eksctl.

> **Note**
>
> Los siguientes pasos requieren la versión `0.220.0` o posterior de eksctl. Para comprobar la versión, ejecute `eksctl version`.

Paso 1: creación de un rol de capacidad de IAM

Cree un archivo de política de confianza:

```
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
```

Cree el rol de IAM:

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

> **Note**
>
> Para esta configuración básica, no se necesitan políticas de IAM adicionales. Si tiene previsto usar Secrets Manager para las credenciales del repositorio o CodeConnections, tendrá que agregar permisos al rol. Para ver ejemplos de políticas de IAM y guías de configuración, consulte the section called “ AWS Secrets Manager” y the section called “ AWS CodeConnections”.

Paso 2: obtención de la configuración de AWS Identity Center

Obtenga el ARN y el ID de usuario de su instancia de Identity Center para la configuración de RBAC:

```
# 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
```

Anote estos valores, ya que los necesitará en el siguiente paso.

Paso 3: creación de un archivo de configuración de eksctl

Cree un archivo llamado `argocd-capability.yaml` con el siguiente contenido. Sustituya los valores de los marcadores de posición por el nombre del clúster, la región, el ARN del rol de IAM, el ARN de la instancia de Identity Center, la región de Identity Center y el ID de usuario:

```
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**
>
> Puede agregar varios usuarios o grupos a las asignaciones de RBAC. Para los grupos, utilice `type: SSO_GROUP` y proporcione el ID del grupo. Los roles disponibles son `ADMIN`, `EDITOR` y `VIEWER`.

Paso 4: creación de la capacidad de Argo CD

Aplique el archivo de configuración:

```
eksctl create capability -f argocd-capability.yaml
```

El comando vuelve inmediatamente, pero la capacidad tarda algún tiempo en activarse.

Paso 5: comprobación de la activación de la capacidad

Compruebe el estado de la capacidad. Reemplace *region-code* por la región de AWS donde creó el clúster y *my-cluster* por el nombre de su clúster.

```
eksctl get capability \
  --region region-code \
  --cluster my-cluster \
  --name my-argocd
```

La capacidad estará lista cuando aparezca el estado `ACTIVE`.

Paso 6: comprobación de la disponibilidad de los recursos personalizados

Una vez que la capacidad esté activa, compruebe que los recursos personalizados de Argo CD estén disponibles en el clúster:

```
kubectl api-resources | grep argoproj.io
```

Debería ver todos los tipos de recursos `Application` y `ApplicationSet` en la lista.

Siguientes pasos

- the section called “Uso de Argo CD”: más información sobre cómo crear y administrar aplicaciones de Argo CD
- the section called “Consideraciones sobre Argo CD”: configuración del SSO y el acceso a varios clústeres
- the section called “Uso de capacidades”: administración del recurso de la capacidad de Argo CD

Conceptos de Argo CD

Para implementar GitOps, Argo CD trata a Git como el único origen de información para las implementaciones de aplicaciones. En este tema, se muestra un ejemplo práctico y, a continuación,

se explican los conceptos básicos que debe comprender al trabajar con la capacidad de EKS para Argo CD.

Introducción a Argo CD

Después de crear la capacidad de Argo CD (consulte the section called “Creación de la capacidad Argo CD”), puede comenzar a implementar aplicaciones. En este ejemplo, se explica el registro de un clúster y la creación de una aplicación.

Paso 1: Configurar

Registro del clúster (obligatorio)

Registre el clúster en el que desea implementar las aplicaciones. En este ejemplo, registraremos el mismo clúster en el que se ejecuta Argo CD (puede usar el nombre `in-cluster` por motivos de compatibilidad con la mayoría de los ejemplos de Argo CD):

```
# 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**
>
> Para obtener información sobre cómo configurar la CLI de Argo CD para que funcione con la capacidad de Argo CD en EKS, consulte the section called “Uso de la CLI de Argo CD con la capacidad administrada”.

Como alternativa, registre el clúster mediante un secreto de Kubernetes (consulte the section called “Registro de clústeres” para obtener más información).

Configuración del acceso al repositorio (opcional)

En este ejemplo, se usa un repositorio de GitHub público, por lo que no se requiere ninguna configuración del repositorio. Para los repositorios privados, configure el acceso mediante AWS Secrets Manager, CodeConnections o secretos de Kubernetes (consulte the section called "Configuración de repositorios" para obtener más información).

Para los servicios de AWS (ECR para gráficos de Helm, CodeConnections y CodeCommit), puede hacer referencia a ellos directamente en los recursos de la aplicación sin necesidad de crear un repositorio. El rol de capacidad debe tener los permisos para llamar a los permisos de IAM necesarios. Para obtener más información, consulte the section called "Configuración de repositorios".

Paso 2: Cree una aplicación

Cree este manifiesto de aplicación en `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
```

Aplique la aplicación:

```
kubectl apply -f my-app.yaml
```

Después de aplicar esta aplicación, Argo CD: 1. Sincroniza la aplicación de Git con el clúster (implementación inicial). 2. Supervisa el repositorio de Git para detectar cambios. 3. Sincroniza

automáticamente los cambios posteriores en el clúster. 4. Detecta y corrige cualquier desviación del estado deseado. 5. Proporciona el estado y el historial de sincronización en la interfaz de usuario.

Consulte el estado de la aplicación:

```
kubectl get application guestbook -n argocd
```

También puede ver la aplicación mediante la CLI de Argo CD (`argocd app get guestbook`) o la interfaz de usuario de Argo CD (a la que se puede acceder desde la consola de EKS, en la pestaña Capacidades del clúster).

> **Note**
>
> Utilice el nombre del clúster en `destination.name` (el nombre que utilizó al registrar el clúster). La capacidad administrada no admite el valor predeterminado local en el clúster (`kubernetes.default.svc`).

Conceptos clave

Principios y tipos de orígenes de GitOps

Argo CD implementa GitOps, donde el origen de la aplicación es el único origen de información para las implementaciones:

- Declarativo: el estado deseado se declara mediante manifiestos de YAML, gráficos de Helm o superposiciones de Kustomize.
- Con control de versiones: se hace un seguimiento de cada cambio con un registro de auditoría completo.
- Automatizado: Argo CD supervisa continuamente los orígenes y sincroniza automáticamente los cambios.
- Recuperación automática: detecta y corrige una desviación entre el estado deseado y el real del clúster.

Tipos de orígenes compatibles:

- Repositorios de Git: GitHub, GitLab, Bitbucket, CodeCommit (HTTPS, SSH o CodeConnections)

- Registros de Helm: registros HTTP (como `https://aws.github.io/eks-charts`) y registros OCI (como `public.ecr.aws`)
- Imágenes OCI: imágenes de contenedor que incluyen manifiestos o gráficos de Helm (como `oci://registry-1.docker.io/user/my-app`)

Esta flexibilidad permite a las organizaciones elegir orígenes que cumplan con sus requisitos de seguridad y conformidad. Por ejemplo, las organizaciones que restringen el acceso a Git desde clústeres pueden usar ECR para gráficos de Helm o imágenes OCI.

Para obtener más información, consulte Application Sources en la documentación de Argo CD.

Sincronización y conciliación

Argo CD supervisa continuamente sus orígenes y clústeres para detectar y corregir las diferencias:

1. Sondea los orígenes para ver si hay cambios (de forma predeterminada, cada 6 minutos).
2. Compara el estado deseado con el estado del clúster.
3. Marca las aplicaciones como `Synced` o `OutOfSync`.
4. Sincroniza los cambios automáticamente (si se configuró) o espera su aprobación manual.
5. Supervisa el estado de los recursos después de la sincronización.

Las ondas de sincronización controlan el orden de creación de los recursos mediante anotaciones:

```
metadata:
  annotations:
    argocd.argoproj.io/sync-wave: "0"  # Default if not specified
```

Los recursos se aplican por orden de onda (primero, los números más bajos, incluidos los negativos, como `-1`). Esto le permite crear dependencias como espacios de nombres (onda `-1`) antes de las implementaciones (onda `0`).

La recuperación automática revierte los cambios manuales:

```
spec:
  syncPolicy:
    automated:
      selfHeal: true
```

> **Note**
>
> La capacidad administrada utiliza el seguimiento de recursos basado en anotaciones (no en etiquetas) para mejorar la compatibilidad con las convenciones de Kubernetes y otras herramientas.

Para obtener información detallada sobre las fases de la sincronización, los enlaces y los patrones avanzados, consulte la documentación de sincronización de Argo CD.

Estado de la aplicación

Argo CD supervisa el estado de todos los recursos de la aplicación:

Estados: * En buen estado (todos los recursos se ejecutan según lo esperado) * En curso (los recursos se están creando o actualizando) * Degradado (algunos recursos no están en buen estado: bloqueo de pods, errores de trabajos) * Suspendido (la aplicación se pausó de forma intencionada) * Falta (los recursos definidos en Git no están en el clúster)

Argo CD tiene comprobaciones de estado integradas para los recursos comunes de Kubernetes (implementaciones, StatefulSets, trabajos, etc.) y admite comprobaciones de estado personalizadas para CRD.

El estado de la aplicación viene determinado por todos sus recursos: si hay algún recurso `Degraded`, el estado de la aplicación será `Degraded`.

Para obtener más información, consulte Resource Health en la documentación de Argo CD.

Patrones de varios clústeres

Argo CD admite dos patrones de implementación principales:

Radial: ejecute Argo CD en un clúster de administración dedicado que se implemente en varios clústeres de cargas de trabajo. * Control y visibilidad centralizados * Políticas coherentes en todos los clústeres * Una instancia de Argo CD que administrar * Separación clara entre el plano de control y las cargas de trabajo

Por clúster: ejecute Argo CD en cada clúster y administre solo las aplicaciones de ese clúster. * Separación de clústeres (un error no afecta a los demás) * Redes más sencillas (sin comunicación entre clústeres) * Configuración inicial más sencilla (sin registro de clústeres)

Elija la opción radial para los equipos de plataformas que administran varios clústeres, o la opción por clúster para equipos independientes o cuando los clústeres deban estar completamente aislados.

Para obtener información detallada sobre la configuración de varios clústeres, consulte the section called “Consideraciones sobre Argo CD”.

Proyectos

Los proyectos proporcionan agrupamiento lógico y control de acceso para las aplicaciones:

- Restricciones de origen: limite los repositorios de Git que se pueden usar.
- Restricciones de destino: limite los clústeres y espacios de nombres que se pueden usar como destino.
- Restricciones de recursos: limite los tipos de recursos de Kubernetes que se pueden implementar.
- Integración con RBAC: asigne proyectos a ID de usuarios y grupos de AWS Identity Center.

Todas las aplicaciones pertenecen a un proyecto. Si no se especifica, utilizan el proyecto `default` (que no tiene restricciones). Para la producción, cree proyectos con las restricciones adecuadas.

Para obtener información sobre la configuración de proyectos y los patrones RBAC, consulte the section called “Configuración de permisos”.

Organización de repositorios

La mayoría de los equipos utilizan una organización basada en directorios con superposiciones de Kustomize o archivos de valores de Helm para distintos entornos:

```
my-app/
### base/
#   ### deployment.yaml
#   ### service.yaml
### overlays/
    ### dev/
    #   ### kustomization.yaml
    ### staging/
    #   ### kustomization.yaml
    ### prod/
        ### kustomization.yaml
```

Este enfoque proporciona flexibilidad y claridad y, al mismo tiempo, mantiene todas las configuraciones de entornos en un único repositorio.

Para obtener información detallada sobre los patrones de estructura de repositorios y las prácticas recomendadas, consulte la documentación sobre prácticas recomendadas de Argo CD.

Opciones de sincronización

Refine el comportamiento de la sincronización con opciones comunes:

- `CreateNamespace=true`: se crea automáticamente el espacio de nombres de destino.
- `ServerSideApply=true`: se utiliza la aplicación del servidor para una mejor resolución de conflictos.
- `SkipDryRunOnMissingResource=true`: se omite la ejecución de prueba cuando las CRD aún no existan (útil para instancias de kro).

```
spec:
  syncPolicy:
    syncOptions:
    - CreateNamespace=true
    - ServerSideApply=true
```

Para obtener una lista completa de las opciones de sincronización, consulte la documentación sobre las opciones de sincronización de Argo CD.

Siguientes pasos

- the section called “Configuración de repositorios”: configuración del acceso al repositorio de Git
- the section called “Registro de clústeres”: registro de los clústeres de destino para la implementación
- the section called “Creación de aplicaciones”: creación de la primera aplicación
- the section called “Consideraciones sobre Argo CD”: patrones específicos de EKS, integración con Identity Center y configuración de varios clústeres
- Documentación de Argo CD: documentación completa de Argo CD que incluye enlaces de sincronización, comprobaciones de estado y patrones avanzados

Configuración de los permisos de Argo CD

La capacidad administrada de Argo CD se integra con AWS Identity Center para la autenticación y utiliza roles de RBAC integrados para la autorización. En este tema, se explica cómo configurar los permisos para los usuarios y los equipos.

Cómo funcionan los permisos con Argo CD

La capacidad de Argo CD utiliza AWS Identity Center para la autenticación y proporciona tres roles de RBAC integrados para la autorización.

Cuando un usuario accede a Argo CD:

1. Se autentican mediante AWS Identity Center (que puede federarse con su proveedor de identidad corporativa)
2. AWS Identity Center proporciona información sobre los usuarios y grupos a Argo CD
3. Argo CD asigna usuarios y grupos a roles de RBAC en función de su configuración
4. Los usuarios solo ven las aplicaciones y los recursos a los que tienen permiso de acceso

Roles de RBAC integrados

La capacidad de Argo CD proporciona tres roles integrados que se asignan a los usuarios y grupos de AWS Identity Center.

ADMIN

Acceso completo a todas las aplicaciones y configuraciones:

- Creación, actualización y eliminación de aplicaciones y ApplicationSets
- Administración de la configuración de Argo CD
- Registro y administración de los clústeres de destino de la implementación
- Configuración del acceso al repositorio
- Administración de proyectos
- Visualización del estado y el historial de todas las aplicaciones

EDITOR

Puede crear y modificar aplicaciones, pero no puede cambiar la configuración de Argo CD:

- Creación y actualización de aplicaciones y ApplicationSets
- Sincronización y actualización de aplicaciones
- Visualización del estado y el historial de las aplicaciones
- Imposibilidad de eliminar aplicaciones
- Imposibilidad de cambiar la configuración de Argo CD
- Imposibilidad de administrar clústeres o repositorios

VIEWER

Acceso de solo lectura a las aplicaciones:

- Visualización del estado y el historial de las aplicaciones
- Visualización de los manifiestos y los recursos de las aplicaciones
- Imposibilidad de hacer cambios
- Imposibilidad de sincronizar o actualizar aplicaciones

Configuración de las asignaciones de roles

Asigne los usuarios y grupos de AWS Identity Center a los roles de Argo CD al crear o actualizar la capacidad.

Ejemplo de asignación de roles:

```
{
  "rbacRoleMapping": {
    "ADMIN": ["AdminGroup", "alice@example.com"],
    "EDITOR": ["DeveloperGroup", "DevOpsTeam"],
    "VIEWER": ["ReadOnlyGroup", "bob@example.com"]
  }
}
```

> **Note**
>
> Los nombres de los roles distinguen entre mayúsculas y minúsculas y deben estar en mayúsculas (ADMIN, EDITOR, VIEWER).

> **Important**
>
> La integración de capacidades de EKS con AWS Identity Center admite hasta 1000 identidades por capacidad de Argo CD. Una identidad puede ser un usuario o un grupo.

Actualice las asignaciones de roles:

```
aws eksfe 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" }
            ]
          }
        ]
      }
    }
  }'
```

Uso de la cuenta de administrador

La cuenta de administrador está diseñada para la configuración inicial y las tareas administrativas, como el registro de clústeres y la configuración de repositorios.

Cuándo es adecuado usar la cuenta de administrador:

- Configuración inicial de la capacidad
- Desarrollo individual o demostraciones rápidas
- Tareas administrativas (registro de clústeres, configuración de repositorios, creación de proyectos)

Prácticas recomendadas para la cuenta de administrador:

- No asigne tokens de la cuenta al control de versiones.
- Cambie los tokens inmediatamente si se han expuesto.
- Limite el uso de los tokens de la cuenta a las tareas administrativas y de configuración.
- Establezca tiempos de vencimiento cortos (máximo 12 horas).
- Solo se pueden crear 5 tokens de la cuenta en un momento determinado.

Cuándo se usa el acceso basado en proyectos:

- Entornos de desarrollo compartidos con varios usuarios
- Cualquier entorno que se asemeje al de producción
- Cuando necesite registros de auditoría sobre quién llevó a cabo las acciones
- Cuando necesite aplicar restricciones de recursos o límites de acceso

Para entornos de producción y escenarios de varios usuarios, utilice el control de acceso basado en proyectos con roles de RBAC dedicados asignados a grupos de AWS Identity Center.

Control de acceso basado en proyectos

Utilice proyectos de Argo CD (AppProject) para proporcionar un control del acceso y un aislamiento de los recursos detallados para los equipos.

Los proyectos proporcionan lo siguiente:

- Restricciones de origen: limite los repositorios de Git que se pueden usar.
- Restricciones de destino: limite los clústeres y espacios de nombres que se pueden usar como destino.
- Restricciones de recursos: limite los tipos de recursos de Kubernetes que se pueden implementar.
- Integración con RBAC: asigne los proyectos a grupos de AWS Identity Center o roles de Argo CD.

Ejemplo de proyecto para el aislamiento de equipos:

```
apiVersion: argoproj.io/v1alpha1
kind: AppProject
metadata:
```

```
  name: team-a
  namespace: argocd
spec:
  description: Team A applications

  # 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
```

Asigne usuarios a los proyectos:

Los usuarios con los roles EDITOR o VIEWER pueden restringirse a proyectos específicos. Los usuarios ADMIN tienen acceso a todos los proyectos.

```
apiVersion: argoproj.io/v1alpha1
kind: AppProject
metadata:
  name: team-a
spec:
  # ... project configuration ...

  # Map Identity Center groups to project roles
  roles:
  - name: developer
    description: Team A developers
    policies:
```

```
    - p, proj:team-a:developer, applications, *, team-a/*, allow
    groups:
    - TeamADevelopers

  - name: viewer
    description: Team A viewers
    policies:
    - p, proj:team-a:viewer, applications, get, team-a/*, allow
    groups:
    - TeamAViewers
```

Patrones de permisos comunes

Patrón 1: equipo de administración con acceso completo

```
{
  "rbacRoleMapping": {
    "ADMIN": ["PlatformTeam", "SRETeam"]
  }
}
```

Patrón 2: los desarrolladores pueden implementar, otros pueden ver

```
{
  "rbacRoleMapping": {
    "ADMIN": ["PlatformTeam"],
    "EDITOR": ["DevelopmentTeam", "DevOpsTeam"],
    "VIEWER": ["AllEmployees"]
  }
}
```

Patrón 3: aislamiento basado en equipos con proyectos

1. Asigne todos los desarrolladores al rol EDITOR.
2. Cree AppProjects independientes para cada equipo.
3. Use los roles del proyecto para restringir el acceso a las aplicaciones específicas del equipo.

```
{
  "rbacRoleMapping": {
    "ADMIN": ["PlatformTeam"],
```

```
    "EDITOR": ["AllDevelopers"]
  }
}
```

A continuación, cree proyectos con asignaciones de roles y restricciones específicas del equipo.

Prácticas recomendadas

Utilice grupos en lugar de usuarios individuales: asigne grupos de AWS Identity Center a roles de Argo CD en lugar de usuarios individuales para facilitar la administración.

Comience con el privilegio mínimo: comience con el acceso de VIEWER y otorgue el de EDITOR o ADMIN según sea necesario.

Use los proyectos para el aislamiento de equipos: cree AppProjects independientes para diferentes equipos o entornos a fin de aplicar los límites.

Aproveche la federación de Identity Center: configure AWS Identity Center para federarse con su proveedor de identidad corporativa a fin de administrar los usuarios de forma centralizada.

Revise el acceso periódicamente: revise periódicamente las asignaciones de roles y de proyectos para garantizar los niveles de acceso adecuados.

Limite el acceso a los clústeres: recuerde que el RBAC de Argo CD controla el acceso a los recursos y las operaciones de Argo CD, pero no se corresponde con el RBAC de Kubernetes. Los usuarios con acceso de Argo CD pueden implementar aplicaciones en los clústeres a los que Argo CD tiene acceso. Limite los clústeres a los que puede acceder Argo CD y utilice las restricciones de destino del proyecto para controlar dónde se pueden implementar las aplicaciones.

AWSPermisos de servicios de

Para utilizar los servicios de AWS directamente en los recursos de la aplicación (sin crear recursos del repositorio), adjunte los permisos de IAM necesarios al rol de capacidad.

ECR para gráficos de Helm:

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

```
        "ecr:GetAuthorizationToken",
        "ecr:BatchCheckLayerAvailability",
        "ecr:GetDownloadUrlForLayer",
        "ecr:BatchGetImage"
      ],
      "Resource": "*"
    }
  ]
}
```

Repositorios de CodeCommit:

```
{
  "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"
    }
  ]
}
```

Consulte the section called “Configuración de repositorios” para obtener más información sobre el uso de estas integraciones.

Siguientes pasos

- the section called “Uso de Argo CD”: más información sobre cómo crear aplicaciones y administrar implementaciones
- the section called “Conceptos de Argo CD”: descripción de los conceptos de Argo CD, como los proyectos
- the section called “Consideraciones para las capacidades de EKS”: revisión de las prácticas recomendadas de seguridad para capacidades

Uso de Argo CD

Con Argo CD, define aplicaciones en los repositorios de Git y Argo CD las sincroniza automáticamente con los clústeres de Kubernetes. Esto permite la implementación declarativa de aplicaciones con control de versiones y una detección automática de desviaciones.

Requisitos previos

Antes de trabajar con Argo CD, necesita lo siguiente:

- Un clúster de EKS con la capacidad de Argo CD creada (consulte the section called “Creación de la capacidad Argo CD”)
- Un repositorio de Git que contiene manifiestos de Kubernetes
- `kubectl` configurado para comunicarse con el clúster

Tareas comunes

Los siguientes temas lo guían en las tareas más comunes de Argo CD:

the section called “Configuración de repositorios”: configure Argo CD para acceder a repositorios de Git mediante AWS Secrets Manager, AWS CodeConnections o secretos de Kubernetes.

the section called “Registro de clústeres”: registre los clústeres de destino en los que Argo CD implementará las aplicaciones.

the section called “Proyectos”: organice aplicaciones y aplique límites de seguridad con proyectos para los entornos de varios inquilinos.

the section called “Creación de aplicaciones”: cree aplicaciones de Argo CD que se implementan desde los repositorios de Git con sincronización automática o manual.

the section called “ApplicationSets”: utilice ApplicationSets para implementar aplicaciones en varios entornos o clústeres mediante plantillas y generadores.

Acceso a la interfaz de usuario de Argo CD

Acceda a la interfaz de usuario de Argo CD a través de la consola de EKS:

1. Abra la consola de Amazon EKS.
2. Seleccione el clúster.
3. Elija la pestaña Capacidades.
4. Elija Argo CD.
5. Elija Abrir la interfaz de usuario de Argo CD.

La interfaz de usuario proporciona la topología visual de las aplicaciones, el estado y el historial de la sincronización, los eventos y el estado de los recursos, los controles de sincronización manual y la administración de aplicaciones.

Documentación ascendente

Para obtener información detallada acerca de las características de Argo CD:

- Documentación de Argo CD: guía del usuario completa
- Application Spec: referencia completa de la API de la aplicación
- Guía de ApplicationSet: patrones y ejemplos de ApplicationSet
- GitHub de Argo CD: código fuente y ejemplos

Configuración del acceso al repositorio

Antes de implementar aplicaciones, configure Argo CD para acceder a sus repositorios de Git y registros de gráficos de Helm. Argo CD admite varios métodos de autenticación para GitHub, GitLab, Bitbucket, AWS CodeCommit y AWS ECR.

> **Note**
>
> Para las integraciones directas de servicios de AWS (gráficos de Helm de ECR, repositorios de CodeCommit y CodeConnections), puede hacer referencia a ellas directamente en

los recursos de la aplicación sin necesidad de crear un repositorio. El rol de capacidad debe tener los permisos para llamar a los permisos de IAM necesarios. Para obtener más información, consulte the section called “Configuración de permisos”.

Requisitos previos

- Un clúster de EKS con la capacidad de Argo CD creada
- Repositorios de Git que contienen manifiestos de Kubernetes
- `kubectl` configurado para comunicarse con el clúster

> **Note**
>
> Para la reutilización de credenciales en varios repositorios, puede usar plantillas de credenciales de repositorio (repocreds). Para obtener más información, consulte Private Repositories en la documentación de Argo CD.

Métodos de autenticación

Método	Caso de uso	Permisos de IAM necesarios
Integración directa con servicios de AWS		
CodeCommit	Integración directa con los repositorios de Git de AWS CodeCommit. No es necesario configurar el repositorio.	`codecommit:GitPull`
CodeConnections	Conéctese a GitHub, GitLab o Bitbucket con autenticación administrada. Requiere la configuración de la conexión.	`codeconnections:UseConnection`
Gráficos de Helm de ECR	Integración directa con AWS ECR para los gráficos de Helm de OCI. No es necesario configurar el repositorio.	`ecr:GetAuthorizationToken` , `ecr:BatchGetImage` , `ecr:GetDownloadUrlForLayer`

Método	Caso de uso	Permisos de IAM necesarios
Configuración del repositorio con credenciales		
AWS Secrets Manager (nombre de usuario o token)	Almacenamiento de claves o contraseñas de acceso personal	`secretsmanager:GetSecretValue`
AWS Secrets Manager (clave SSH)	Uso de autenticación con clave SSH	`secretsmanager:GetSecretValue`
AWS Secrets Manager (aplicación de GitHub)	Autenticación de la aplicación de GitHub con clave privada	`secretsmanager:GetSecretValue`
Secreto de Kubernetes	Método de Argo CD estándar con secretos en el clúster	Ninguno (solo política de confianza)

Acceso directo a los servicios de AWS

Para los servicios de AWS, puede hacer referencia a ellos directamente en los recursos de la aplicación sin necesidad de crear configuraciones del repositorio. El rol de capacidad debe tener los permisos para llamar a los permisos de IAM necesarios.

Repositorios de CodeCommit

Haga referencia a los repositorios de CodeCommit directamente en las aplicaciones:

```
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
```

Permisos de rol de capacidad necesarios:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codecommit:GitPull",
      "Resource": "arn:aws:codecommit:region:account-id:repository-name"
    }
  ]
}
```

CodeConnections

Haga referencia a los repositorios de GitHub, GitLab o Bitbucket a través de CodeConnections. El formato de URL del repositorio se deriva del ARN de conexión de CodeConnections.

El formato de URL del repositorio es:

```
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
```

Permisos de rol de capacidad necesarios:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codeconnections:UseConnection",
```

```
      "Resource": "arn:aws:codeconnections:region:account-id:connection/connection-id"
    }
  ]
}
```

Gráficos de Helm de ECR

ECR almacena los gráficos de Helm como artefactos OCI. Argo CD admite dos formas de hacer referencia a ellos:

Formato Helm (recomendado para gráficos de Helm):

```
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
```

Nota: No incluya el prefijo `oci://` cuando utilice el formato Helm. Utilice el campo `chart` para especificar el nombre del gráfico.

Formato OCI (para artefactos OCI con manifiestos de Kubernetes):

```
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
```

Nota: Incluya el prefijo `oci://` cuando utilice el formato OCI. Utilice el campo `path` en lugar de `chart`.

Permisos de rol de capacidad necesarios:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecr:GetAuthorizationToken",
        "ecr:BatchGetImage",
        "ecr:GetDownloadUrlForLayer"
      ],
      "Resource": "*"
    }
  ]
}
```

Uso de AWS Secrets Manager

Almacene las credenciales del repositorio en Secrets Manager y haga referencia a ellas en las configuraciones del repositorio de Argo CD.

Autenticación con nombre de usuario y token

Para los repositorios HTTPS con contraseñas o tokens de acceso personales:

Cree un secreto en 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"}'
```

Campos de certificado de cliente TLS opcionales (para servidores Git privados):

```
aws secretsmanager create-secret \
  --name argocd/my-private-repo \
  --secret-string '{
    "username":"your-username",
```

```
    "token":"your-token",

 "tlsClientCertData":"LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCi4uLgotLS0tLUVORCBDRVJUSUZJQ0FURS0tL

 "tlsClientCertKey":"LS0tLS1CRUdJTiBQUklWQVRFIEtFWS0tLS0tCi4uLgotLS0tLUVORCBQUklWQVRFIEtFWS0tLS
  }'
```

> **Note**
>
> Los valores de `tlsClientCertData` y `tlsClientCertKey` deben estar codificados en base64.

Cree un secreto de repositorio que haga referencia a Secrets Manager:

```
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
```

Autenticación con clave SSH

Para el acceso a Git basado en SSH, almacene la clave privada como texto sin formato (no JSON):

Cree el secreto con la clave SSH privada:

```
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-----"
```

Cree un secreto de repositorio para 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
```

Autenticación de la aplicación de GitHub

Para la autenticación de la aplicación de GitHub con una clave privada:

Cree el secreto con las credenciales de la aplicación de GitHub:

```
aws secretsmanager create-secret \
  --name argocd/github-app \
  --description "GitHub App credentials for Argo CD" \
  --secret-string '{

 "githubAppPrivateKeySecret":"LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQouLi4KLS0tLS1FTkQgUlNBI
    "githubAppID":"123456",
    "githubAppInstallationID":"12345678"
  }'
```

> **Note**
>
> El valor de `githubAppPrivateKeySecret` debe estar codificado en base64.

Campo opcional para GitHub Enterprise:

```
aws secretsmanager create-secret \
  --name argocd/github-enterprise-app \
  --secret-string '{
```

```
 "githubAppPrivateKeySecret":"LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQouLi4KLS0tLS1FTkQgUlNBI
    "githubAppID":"123456",
    "githubAppInstallationID":"12345678",
    "githubAppEnterpriseBaseUrl":"https://github.example.com/api/v3"
  }'
```

Cree un secreto de repositorio para la aplicación de GitHub:

```
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
```

> **Important**
>
> Asegúrese de que su rol de capacidad de IAM tenga los permisos `secretsmanager:GetSecretValue` para los secretos que cree. Consulte the section called “Consideraciones sobre Argo CD” para obtener información sobre la configuración de las políticas de IAM.

Uso de AWS CodeConnections

Para obtener información sobre la integración de CodeConnections, consulte the section called “ AWS CodeConnections”.

CodeConnections proporciona autenticación administrada para GitHub, GitLab y Bitbucket sin almacenar credenciales.

Uso de secretos de Kubernetes

Almacene las credenciales directamente en Kubernetes mediante el método de Argo CD estándar.

Para HTTPS con token de acceso personal:

```
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
```

Para 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-----
```

Repositorios públicos

No se necesita ninguna configuración adicional para los repositorios públicos:

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: public-app
  namespace: argocd
spec:
  source:
```

```
    repoURL: https://github.com/argoproj/argocd-example-apps
    targetRevision: HEAD
    path: guestbook
  # ... rest of configuration
```

Repositorios de CodeCommit

Para AWS CodeCommit, conceda permisos de CodeCommit a su rol de capacidad de IAM (`codecommit:GitPull`).

Configure el repositorio:

```
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
```

Para obtener información detallada sobre la configuración de la política de IAM, consulte the section called “Consideraciones sobre Argo CD”.

Comprobación de la conexión del repositorio

Compruebe el estado de la conexión a través de la interfaz de usuario de Argo CD, en Configuración → Repositorios. La interfaz de usuario muestra el estado de la conexión y cualquier error de autenticación.

Los secretos de repositorio no incluyen información sobre el estado.

Recursos adicionales

- the section called “Registro de clústeres”: registro de los clústeres de destino para implementaciones
- the section called “Creación de aplicaciones”: creación de la primera aplicación
- the section called “Consideraciones sobre Argo CD”: configuración de seguridad y permisos de IAM

- Private Repositories: referencia de la configuración de repositorio ascendente

Registro de clústeres de destino

Registre los clústeres para que Argo CD pueda implementar aplicaciones en ellos. Puede registrar el mismo clúster en el que se ejecuta Argo CD (clúster local) o clústeres remotos en cuentas o regiones diferentes.

Requisitos previos

- Un clúster de EKS con la capacidad de Argo CD creada
- `kubectl` configurado para comunicarse con el clúster
- Para clústeres remotos: permisos de IAM y entradas de acceso adecuados

Registro del clúster local

Para implementar aplicaciones en el mismo clúster en el que se ejecuta Argo CD, regístrelo como destino de implementación.

> **Note**
>
> La capacidad de Argo CD no registra automáticamente el clúster local. Debe registrarlo de forma explícita para implementar aplicaciones en el mismo clúster.

Uso de la CLI de Argo CD:

```
argocd cluster add <cluster-context-name> \
  --aws-cluster-name arn:aws:eks:us-west-2:111122223333:cluster/my-cluster \
  --name local-cluster
```

Uso de un secreto de Kubernetes:

```
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
```

Aplique la configuración:

```
kubectl apply -f local-cluster.yaml
```

> **Note**
>
> Use el ARN del clúster de EKS en el campo `server`, no la URL del servidor de API de Kubernetes. La capacidad administrada requiere que los ARN identifiquen los clústeres. El `kubernetes.default.svc` predeterminado no es compatible.

Registro de clústeres remotos

Para implementar en clústeres remotos, debe hacer lo siguiente:

1. Cree una entrada de acceso en el clúster remoto para el rol de capacidad de IAM de Argo CD.
2. Asocie una política de acceso a los permisos adecuados.
3. Registre el clúster en Argo CD.

Paso 1: creación de la entrada de acceso en el clúster remoto

Reemplace *`region-code`* por la región de AWS en la que está el clúster remoto, sustituya *`remote-cluster`* por el nombre del clúster remoto y el ARN por el ARN del rol de capacidad de Argo CD.

```
aws eks create-access-entry \
  --region region-code \
  --cluster-name remote-cluster \
  --principal-arn arn:aws:iam::[.replaceable]111122223333:role/ArgoCDCapabilityRole \
  --type STANDARD
```

Paso 2: asociación de una política de acceso

```
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
```

> **Note**
>
> Para los entornos de producción, considere la posibilidad de utilizar políticas de acceso más restrictivas. Consulte the section called “Consideraciones para las capacidades de EKS” para obtener las configuraciones de privilegio mínimo.

Paso 3: registro del clúster en Argo CD

Uso de la CLI de Argo CD:

```
argocd cluster add <cluster-context-name> \
  --aws-cluster-name arn:aws:eks:us-west-2:111122223333:cluster/remote-cluster \
  --name remote-cluster
```

Uso de un secreto de Kubernetes:

```
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
```

Aplique la configuración:

```
kubectl apply -f remote-cluster.yaml
```

Clústeres entre cuentas y entre regiones

Para implementar en clústeres de diferentes cuentas o regiones de AWS, siga estos pasos:

1. Agregue el rol de capacidad de Argo CD como entrada de acceso en el clúster remoto.
2. Asocie la política de acceso adecuada (por lo general, `AmazonEKSClusterAdminPolicy`).
3. Registre el clúster con su ARN completo (que incluye la región).

El formato del ARN del clúster incluye la región, por lo que no hay diferencia entre el registro entre cuentas y entre regiones, ya que ambos utilizan el mismo proceso.

Para obtener información detallada sobre la configuración entre cuentas, lo que incluye las políticas de confianza y los permisos de IAM, consulte the section called “Consideraciones sobre Argo CD”.

Comprobación del registro del clúster

Consulte los clústeres registrados:

```
kubectl get secrets -n argocd -l argocd.argoproj.io/secret-type=cluster
```

O bien, compruebe el estado del clúster en la interfaz de usuario de Argo CD, en Configuración → Clústeres.

Clústeres privados

La capacidad de Argo CD proporciona un acceso transparente a clústeres de EKS totalmente privados sin necesidad de emparejamiento de VPC ni una configuración de red especializada.

AWS administra automáticamente la conectividad entre la capacidad de Argo CD y los clústeres remotos privados.

Solo tiene que registrar el clúster privado con su ARN, sin necesidad de aplicar ninguna configuración de red adicional.

Restricción del acceso al clúster con proyectos

Use proyectos para controlar en qué clústeres se pueden implementar las aplicaciones:

```
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'
```

Para obtener más información, consulte the section called “Proyectos”.

Recursos adicionales

- the section called “Proyectos”: organización de las aplicaciones y aplicación de los límites de seguridad
- the section called “Creación de aplicaciones”: implementación de su primera aplicación
- the section called “ApplicationSets”: implementación en varios clústeres con ApplicationSets
- the section called “Consideraciones sobre Argo CD”: patrones de varios clústeres y configuración entre cuentas
- Declarative Cluster Setup: referencia de la configuración del clúster ascendente

Uso de proyectos de Argo CD

Los proyectos de Argo CD (AppProjects) proporcionan agrupamiento lógico y control de acceso para las aplicaciones. Los proyectos definen qué repositorios de Git, clústeres de destino y espacios de nombres pueden usar las aplicaciones, lo que habilita la multitenencia y los límites de seguridad en las instancias de Argo CD compartidas.

Cuándo se usan los proyectos

Use los proyectos en los siguientes casos:

- Separación de aplicaciones por equipo, entorno o unidad de negocio
- Restricción de los repositorios desde los que los equipos pueden implementar
- Limitación de los clústeres y espacios de nombres en los que los equipos pueden implementar
- Aplicación de cuotas de recursos y tipos de recursos permitidos
- Concesión de acceso a la implementación de aplicaciones de autoservicio con barreras de protección

Proyecto predeterminado

Cada capacidad de Argo CD incluye un proyecto `default` que permite el acceso a todos los repositorios, clústeres y espacios de nombres. Si bien es útil para las pruebas iniciales, cree proyectos dedicados con restricciones explícitas para su uso en producción.

Para obtener más información sobre la configuración predeterminada del proyecto y cómo restringirla, consulte The Default Project en la documentación de Argo CD.

Creación de un proyecto

Para crear un proyecto, aplique un recurso de `AppProject` al clúster.

Ejemplo: proyecto específico de un equipo

```
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
```

Aplique el proyecto:

```
kubectl apply -f team-a-project.yaml
```

Configuración del proyecto

Repositorios de código fuente

Controle qué repositorios de Git pueden usar las aplicaciones de este proyecto:

```
spec:
  sourceRepos:
    - 'https://github.com/my-org/app-*'  # Wildcard pattern
    - 'https://github.com/my-org/infra'  # Specific repo
```

Puede usar comodines y patrones de negación (prefijo `!`) para permitir o denegar repositorios específicos. Para obtener más información, consulte Managing Projects en la documentación de Argo CD.

Restricciones de destino

Limite dónde se pueden implementar las aplicaciones:

```
spec:
  destinations:
    - name: prod-cluster  # Specific cluster by name
      namespace: production
    - name: '*'  # Any cluster
      namespace: team-a-*  # Namespace pattern
```

Important

Utilice nombres de clústeres y patrones de espacios de nombres específicos en lugar de comodines para los proyectos de producción. De este modo, se impiden las implementaciones accidentales en clústeres o espacios de nombres no autorizados.

Puede utilizar comodines y patrones de negación para controlar los destinos. Para obtener más información, consulte Managing Projects en la documentación de Argo CD.

Restricciones de recursos

Controle qué tipos de recursos de Kubernetes se pueden implementar:

Recursos basados en clústeres:

```
spec:
  clusterResourceWhitelist:
    - group: ''
      kind: Namespace
    - group: 'rbac.authorization.k8s.io'
      kind: Role
```

Recursos basados en espacios de nombres:

```
spec:
  namespaceResourceWhitelist:
    - group: 'apps'
      kind: Deployment
    - group: ''
      kind: Service
    - group: ''
      kind: ConfigMap
    - group: 's3.services.k8s.aws'
      kind: Bucket
```

Use listas de elementos prohibidos para denegar recursos específicos:

```
spec:
  namespaceResourceBlacklist:
    - group: ''
      kind: Secret  # Prevent direct Secret creation
```

Asignación de aplicaciones a proyectos

Al crear una aplicación, especifique el proyecto en el campo `spec.project`:

```
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
```

Las aplicaciones sin un proyecto específico utilizan el proyecto `default`.

Roles del proyecto y RBAC

Los proyectos pueden definir roles personalizados para el control de acceso detallado. Asigne roles del proyecto a los usuarios y grupos de AWS Identity Center según la configuración de la capacidad para controlar quién puede sincronizar, actualizar o eliminar aplicaciones.

Ejemplo: proyecto con roles de desarrollador y administrador

```
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
```

Para obtener más información sobre los roles del proyecto, los tokens JWT para las canalizaciones de CI/CD y la configuración d RBAC, consulte Project Roles en la documentación de Argo CD.

Patrones comunes

Proyectos basados en entornos

Cree proyectos separados para cada entorno:

```
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
```

Proyectos basados en equipos

Aísle los equipos con proyectos específicos:

```
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: '*'
```

Proyectos de varios clústeres

Implemente en varios clústeres con políticas coherentes:

```
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
```

Prácticas recomendadas

Comience con proyectos restrictivos: comience con permisos limitados y amplíelos según sea necesario en lugar de comenzar con un acceso amplio.

Utilice patrones de espacios de nombres: use los comodines en las restricciones de los espacios de nombres (por ejemplo, `team-a-*`) para ofrecer flexibilidad mientras mantiene los límites.

Separe los proyectos de producción: utilice proyectos específicos para la producción con controles más estrictos y políticas de sincronización manual.

Documente los propósitos del proyecto: utilice el campo `description` para explicar para qué sirve cada proyecto y quién debe usarlo.

Revise los permisos del proyecto con regularidad: audite los proyectos periódicamente para asegurarse de que las restricciones sigan alineándose con las necesidades del equipo y los requisitos de seguridad.

Recursos adicionales

- the section called “Configuración de permisos”: configuración de la integración de Identity Center y RBAC
- the section called “Creación de aplicaciones”: creación de aplicaciones dentro de los proyectos
- the section called “ApplicationSets”: uso de ApplicationSets con proyectos para implementaciones de varios clústeres
- Documentación de proyectos de Argo CD: referencia ascendente completa

Creación de aplicaciones

Las aplicaciones representan implementaciones en los clústeres de destino. Cada aplicación define un origen (repositorio de Git) y un destino (clúster y espacio de nombres). Cuando se apliquen, Argo CD creará los recursos especificados en los manifiestos del repositorio de Git en el espacio de nombres del clúster. Las aplicaciones suelen especificar las implementaciones de cargas de trabajo, pero pueden administrar cualquier recurso de Kubernetes disponible en el clúster de destino.

Requisitos previos

- Un clúster de EKS con la capacidad de Argo CD creada
- Acceso al repositorio configurado (consulte the section called “Configuración de repositorios”)
- Clúster de destino registrado (consulte the section called “Registro de clústeres”)
- `kubectl` configurado para comunicarse con el clúster

Creación de una aplicación básica

Defina una aplicación que se implemente desde un repositorio de Git:

```
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:
    server: arn:aws:eks:us-west-2:111122223333:cluster/my-cluster
    namespace: default
```

Important

Use el ARN del clúster de EKS en el campo `destination.server`, no la URL del servidor de API de Kubernetes. La capacidad administrada requiere que los ARN identifiquen los clústeres.

Aplique la aplicación:

```
kubectl apply -f application.yaml
```

Consulte el estado de la aplicación:

```
kubectl get application guestbook -n argocd
```

Configuración del origen

Repositorio de Git:

```
spec:
  source:
    repoURL: https://github.com/example/my-app
    targetRevision: main
    path: kubernetes/manifests
```

Confirmación o etiqueta de Git específica:

```
spec:
```

```
  source:
    targetRevision: v1.2.0  # or commit SHA
```

Gráfico de Helm:

```
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
```

Gráfico de Helm desde ECR:

```
spec:
  source:
    repoURL: oci://account-id.dkr.ecr.region.amazonaws.com/repository-name
    targetRevision: chart-version
    chart: chart-name
```

Si el rol de capacidad tiene los permisos de ECR necesarios, el repositorio se utiliza directamente y no es necesario configurarlo. Para obtener más información, consulte the section called “Configuración de repositorios”.

Repositorio de Git desde CodeCommit:

```
spec:
  source:
    repoURL: https://git-codecommit.region.amazonaws.com/v1/repos/repository-name
    targetRevision: main
    path: kubernetes/manifests
```

Si el rol de capacidad tiene los permisos de CodeCommit necesarios, el repositorio se utiliza directamente y no es necesario configurarlo. Para obtener más información, consulte the section called “Configuración de repositorios”.

Repositorio de Git desde CodeConnections:

```
spec:
  source:
    repoURL: https://codeconnections.region.amazonaws.com/git-http/account-
id/region/connection-id/owner/repository.git
    targetRevision: main
    path: kubernetes/manifests
```

El formato de URL del repositorio se deriva del ARN de conexión de CodeConnections. Si el rol de capacidad tiene los permisos de CodeConnections necesarios y hay una conexión configurada, el repositorio se utiliza directamente y no es necesario configurarlo. Para obtener más información, consulte the section called “Configuración de repositorios”.

Kustomize:

```
spec:
  source:
    repoURL: https://github.com/example/kustomize-app
    targetRevision: main
    path: overlays/production
    kustomize:
      namePrefix: prod-
```

Políticas de sincronización

Controle la forma en que Argo CD sincroniza las aplicaciones.

Sincronización manual (predeterminada):

Las aplicaciones requieren una aprobación manual para la sincronización:

```
spec:
  syncPolicy: {}  # No automated sync
```

Active manualmente la sincronización:

```
kubectl patch application guestbook -n argocd \
  --type merge \
  --patch '{"operation": {"initiatedBy": {"username": "admin"}, "sync": {}}}'
```

Sincronización automática:

Las aplicaciones se sincronizan automáticamente cuando se detectan cambios en Git:

```
spec:
  syncPolicy:
    automated: {}
```

Recuperación automática:

Se revierten automáticamente los cambios manuales en el clúster:

```
spec:
  syncPolicy:
    automated:
      selfHeal: true
```

Cuando se activa, Argo CD revierte cualquier cambio manual hecho directamente en el clúster, lo que garantiza que Git continúe siendo el origen de información verdadera.

Poda:

Se eliminan automáticamente los recursos eliminados de Git:

```
spec:
  syncPolicy:
    automated:
      prune: true
```

> **Warning**
>
> La poda eliminará los recursos del clúster. Úsela con precaución en entornos de producción.

Sincronización automática combinada:

```
spec:
  syncPolicy:
    automated:
      prune: true
      selfHeal: true
    syncOptions:
```

```
    - CreateNamespace=true
```

Opciones de sincronización

Configuración de sincronización adicional:

Cree un espacio de nombres si no existe:

```
spec:
  syncPolicy:
    syncOptions:
    - CreateNamespace=true
```

Valide los recursos antes de aplicar:

```
spec:
  syncPolicy:
    syncOptions:
    - Validate=true
```

Aplicar solo sin sincronización:

```
spec:
  syncPolicy:
    syncOptions:
    - ApplyOutOfSyncOnly=true
```

Características de sincronización avanzadas

Argo CD admite características de sincronización avanzadas para implementaciones complejas:

- Ondas de sincronización: controlan el orden de creación de los recursos con anotaciones `argocd.argoproj.io/sync-wave`
- Enlaces de sincronización: ejecutan trabajos antes o después de la sincronización con anotaciones `argocd.argoproj.io/hook` (PreSync, PostSync, SyncFail)
- Evaluación del estado de los recursos: comprobaciones de estado personalizadas para los recursos específicos de la aplicación

Para obtener más información, consulte Sync Waves y Resource Hooks en la documentación de Argo CD.

Omisión de diferencias

Evite que Argo CD sincronice campos específicos administrados por otros controladores (como la administración de réplicas de HPA):

```
spec:
  ignoreDifferences:
  - group: apps
    kind: Deployment
    jsonPointers:
    - /spec/replicas
```

Para obtener más información sobre los patrones de omisión y las exclusiones de campos, consulte Diffing Customization en la documentación de Argo CD.

Implementación en varios entornos

Implemente la misma aplicación en varios entornos:

Desarrollo de:

```
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:
    server: arn:aws:eks:us-west-2:111122223333:cluster/dev-cluster
    namespace: my-app
```

Producción:

```
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:
    server: arn:aws:eks:us-west-2:111122223333:cluster/prod-cluster
    namespace: my-app
  syncPolicy:
    automated:
      prune: true
      selfHeal: true
```

Supervisión y administración de aplicaciones

Consulte el estado de la aplicación:

```
kubectl get application my-app -n argocd
```

Acceda a la interfaz de usuario de Argo CD:

Abra la interfaz de usuario de Argo CD a través de la consola de EKS para ver la topología de la aplicación, el estado de la sincronización, el estado de los recursos y el historial de implementación. Consulte the section called “Uso de Argo CD” para obtener instrucciones de acceso a la interfaz de usuario.

Revierta aplicaciones:

Revierta a una revisión anterior mediante la interfaz de usuario de Argo CD o al actualizar `targetRevision` en la especificación de la aplicación a una confirmación o etiqueta de Git anterior.

Recursos adicionales

- the section called “Proyectos”: organización de aplicaciones con proyectos para entornos multitenencia
- the section called “ApplicationSets”: implementación en varios clústeres con plantillas
- Application Specification: referencia completa de la API de la aplicación
- Sync Options: configuración de sincronización avanzada

Uso de ApplicationSets

Los ApplicationSets generan varias aplicaciones a partir de plantillas, lo que le permite implementar la misma aplicación en varios clústeres, entornos o espacios de nombres con una sola definición de recursos.

Requisitos previos

- Un clúster de EKS con la capacidad de Argo CD creada
- Varios clústeres de destino registrados (consulte the section called “Registro de clústeres”)
- Acceso al repositorio configurado (consulte the section called “Configuración de repositorios”)
- `kubectl` configurado para comunicarse con el clúster

Cómo funcionan los ApplicationSets

Los ApplicationSets utilizan generadores para producir parámetros y, a continuación, aplicarlos a una plantilla de aplicación. Cada conjunto de parámetros generados crea una aplicación.

Generadores comunes para implementaciones de EKS:

- Generador de listas: defina de forma explícita los clústeres y los parámetros para cada entorno.
- Generador de clústeres: implemente automáticamente en todos los clústeres registrados.
- Generador de Git: genere aplicaciones a partir de la estructura del repositorio.
- Generador de matrices: combine generadores para implementaciones multidimensionales.

Para obtener una referencia completa sobre generadores, consulte la documentación de ApplicationSet.

Generador de listas

Implemente en varios clústeres con una configuración explícita:

```
apiVersion: argoproj.io/v1alpha1
kind: ApplicationSet
metadata:
  name: guestbook-all-clusters
  namespace: argocd
```

```
spec:
  generators:
  - list:
      elements:
      - cluster: arn:aws:eks:us-west-2:111122223333:cluster/dev-cluster
        environment: dev
        replicas: "2"
      - cluster: arn:aws:eks:us-west-2:111122223333:cluster/staging-cluster
        environment: staging
        replicas: "3"
      - cluster: arn:aws:eks:us-west-2:111122223333:cluster/prod-cluster
        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:
        server: '{{cluster}}'
        namespace: guestbook
      syncPolicy:
        automated:
          prune: true
          selfHeal: true
```

> **Note**
>
> Utilice los ARN de los clústeres de EKS en el campo `server` cuando apunte a clústeres de EKS registrados. También puede utilizar los nombres de los clústeres con `destination.name` en lugar de `destination.server`.

Esto crea tres aplicaciones: `guestbook-dev`, `guestbook-staging` y `guestbook-prod`.

Generador de clústeres

Implemente automáticamente en todos los clústeres registrados:

```
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
```

Esto crea automáticamente una aplicación para cada clúster registrado.

Filtrar clústeres:

```
spec:
  generators:
  - clusters:
      selector:
        matchLabels:
          environment: production
```

Generadores de Git

Los generadores de Git crean aplicaciones basadas en la estructura del repositorio:

- Generador de directorios: implemente cada directorio como una aplicación independiente (útil para microservicios).

- Generador de archivos: genere aplicaciones a partir de archivos de parámetros (útil para implementaciones con varios inquilinos).

Ejemplo: implementación de microservicios

```
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:
        server: arn:aws:eks:us-west-2:111122223333:cluster/my-cluster
        namespace: '{{path.basename}}'
      syncPolicy:
        automated:
          prune: true
          selfHeal: true
        syncOptions:
        - CreateNamespace=true
```

Para obtener detalles sobre los generadores de Git y la configuración basada en archivos, consulte Git Generator en la documentación de Argo CD.

Generador de matrices

Combine varios generadores para implementarlos en múltiples dimensiones (entornos × clústeres):

```
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:
        server: '{{server}}'
        namespace: 'app-{{environment}}'
```

Para obtener más información sobre cómo combinar generadores, consulte Matrix Generator en la documentación de Argo CD.

Implementación progresiva

Implemente en entornos de forma secuencial con diferentes políticas de sincronización:

```
apiVersion: argoproj.io/v1alpha1
kind: ApplicationSet
metadata:
  name: progressive-rollout
  namespace: argocd
```

```
spec:
  generators:
  - list:
      elements:
      - environment: dev
        autoSync: "true"
      - environment: staging
        autoSync: "true"
      - environment: prod
        autoSync: "false"
  template:
    metadata:
      name: 'app-{{environment}}'
    spec:
      project: default
      source:
        repoURL: https://github.com/example/app
        targetRevision: HEAD
        path: 'overlays/{{environment}}'
      destination:
        server: arn:aws:eks:us-west-2:111122223333:cluster/{{environment}}-cluster
        namespace: app
      syncPolicy:
        automated:
          prune: true
          selfHeal: '{{autoSync}}'
```

El desarrollo y el uso transitorio se sincronizan automáticamente, mientras que la producción requiere una aprobación manual.

Implementación multirregional

Implemente en clústeres de varias regiones:

```
apiVersion: argoproj.io/v1alpha1
kind: ApplicationSet
metadata:
  name: global-app
  namespace: argocd
spec:
  generators:
  - list:
      elements:
```

```
      - cluster: arn:aws:eks:us-west-2:111122223333:cluster/prod-us-west
        region: us-west-2
      - cluster: arn:aws:eks:us-east-1:111122223333:cluster/prod-us-east
        region: us-east-1
      - cluster: arn:aws:eks:eu-west-1:111122223333:cluster/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:
        server: '{{cluster}}'
        namespace: app
      syncPolicy:
        automated:
          prune: true
          selfHeal: true
```

Administración de ApplicationSets

Vea los ApplicationSets y las aplicaciones generadas:

```
kubectl get applicationsets -n argocd
kubectl get applications -n argocd -l argocd.argoproj.io/application-set-
name=<applicationset-name>
```

Actualice un ApplicationSet:

Modifique la especificación de ApplicationSet y vuelva a aplicarla. Argo CD actualiza automáticamente todas las aplicaciones generadas:

```
kubectl apply -f applicationset.yaml
```

Elimine un ApplicationSet:

```
kubectl delete applicationset <name> -n argocd
```

> **Warning**
>
> Al eliminar un conjunto de aplicaciones, se eliminan todas las aplicaciones generadas. Si esas aplicaciones tienen `prune: true`, sus recursos también se eliminarán de los clústeres de destino.

Recursos adicionales

- the section called “Proyectos”: organización de ApplicationSets con proyectos
- the section called “Creación de aplicaciones”: descripción de la configuración de la aplicación
- Documentación de ApplicationSet: patrones y referencia completos sobre generadores
- Referencia sobre generadores: especificaciones detalladas sobre los generadores

Consideraciones sobre Argo CD

En este tema, se tratan aspectos importantes al usar la capacidad de EKS para Argo CD, como la planificación, los permisos, la autenticación y los patrones de implementación en varios clústeres.

Planificación

Antes de implementar Argo CD, tenga en cuenta lo siguiente:

Estrategia de repositorio: determine dónde se almacenarán los manifiestos de la aplicación (CodeCommit, GitHub, GitLab, Bitbucket). Planifique la estructura del repositorio y la estrategia de ramificación para los diferentes entornos.

Método de autenticación: decida si desea utilizar AWS Identity Center para el inicio de sesión único o administrar directamente los usuarios de Argo CD. Recomendamos el inicio de sesión único para los entornos de producción.

Estrategia de RBAC: planifique qué equipos o usuarios deben tener acceso de administrador, editor o lector. Asígnelos a los grupos de AWS Identity Center o a los roles de Argo CD.

Arquitectura de varios clústeres: determine si administrará varios clústeres desde una única instancia de Argo CD. Considere la posibilidad de utilizar un clúster de administración dedicado para Argo CD.

Organización de las aplicaciones: planifique cómo estructurará las aplicaciones y ApplicationSets. Considere la posibilidad de utilizar proyectos para organizar las aplicaciones por equipo o entorno.

Políticas de sincronización: decida si las aplicaciones deben sincronizarse automáticamente o si deben aprobarse manualmente. La sincronización automática es habitual en el desarrollo y manual en la producción.

Permisos

Para obtener información detallada sobre los roles de capacidad de IAM, las políticas de confianza y las prácticas recomendadas de seguridad, consulte the section called “Rol de IAM de capacidad” y the section called “Consideraciones para las capacidades de EKS”.

Descripción general de los roles de capacidad de IAM

Al crear un recurso de capacidad de Argo CD, proporciona un rol de capacidad de IAM. A diferencia de ACK, Argo CD administra principalmente los recursos de Kubernetes, no los recursos de AWS directamente. Sin embargo, se requiere el rol de IAM para:

- Acceder a los repositorios privados de Git en CodeCommit
- Integrar con AWS Identity Center para la autenticación
- Acceder a los secretos en AWS Secrets Manager (si está configurado)
- Implementaciones entre clústeres en otros clústeres de EKS

Integración de CodeCommit

Si utiliza repositorios de CodeCommit, adjunte una política con permisos de lectura:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codecommit:GitPull"
      ],
      "Resource": "*"
    }
  ]
```

```
}
```

> **Important**
>
> Para su uso en producción, restrinja el campo `Resource` a los ARN de repositorio específicos en lugar de usar `"*"`.
> Ejemplo:
>
> ```
> "Resource": "arn:aws:codecommit:us-west-2:111122223333:my-app-repo"
> ```
>
> De este modo, se limita el acceso de Argo CD solo a los repositorios que necesita administrar.

Integración de Secrets Manager

Si almacena las credenciales del repositorio en Secrets Manager, adjunte una política con permisos de lectura:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue",
        "secretsmanager:DescribeSecret"
      ],
      "Resource": "arn:aws:secretsmanager:*:*:secret:argocd/*"
    }
  ]
}
```

Configuración básica

Para la funcionalidad básica de Argo CD con repositorios de Git públicos, no se requieren políticas de IAM adicionales más allá de la política de confianza.

Autenticación

Integración de AWS Identity Center

La capacidad administrada de Argo CD se integra directamente con AWS Identity Center (anteriormente AWS SSO), lo que le permite utilizar su proveedor de identidad actual para la autenticación.

Al configurar la integración de AWS Identity Center:

1. Los usuarios acceden a la interfaz de usuario de Argo CD a través de la consola de EKS
2. Se autentican mediante AWS Identity Center (que puede federarse con su proveedor de identidad corporativa)
3. AWS Identity Center proporciona información sobre los usuarios y grupos a Argo CD
4. Argo CD asigna usuarios y grupos a roles de RBAC en función de su configuración
5. Los usuarios solo ven las aplicaciones y los recursos a los que tienen permiso de acceso

Simplificación del acceso con los conjuntos de permisos de Identity Center

AWS Identity Center proporciona dos rutas de autenticación distintas cuando se trabaja con Argo CD:

Autenticación mediante la API de Argo CD: Identity Center proporciona autenticación de inicio de sesión único a la interfaz de usuario y a la API de Argo CD. Se configura mediante las asignaciones de roles de RBAC de la capacidad de Argo CD.

Acceso a los clústeres de EKS: la capacidad de Argo CD utiliza el rol de IAM proporcionado por el cliente para autenticarse en los clústeres de EKS mediante las entradas de acceso. Estas entradas de acceso se pueden configurar manualmente para agregar o eliminar permisos.

Puede usar los conjuntos de permisos de Identity Center para simplificar la administración de identidades al permitir que una sola identidad acceda a los clústeres de Argo CD y EKS. De este modo, se reduce la sobrecarga, ya que se le requiere que administre solo una identidad en ambos sistemas, en lugar de mantener credenciales separadas para el acceso a Argo CD y al clúster.

Asignaciones de roles de RBAC

Argo CD cuenta con roles integrados que puede asignar a los usuarios y grupos de AWS Identity Center:

ADMIN: acceso completo a todas las aplicaciones y configuraciones. Puede crear, implementar y eliminar aplicaciones. Puede administrar la configuración de Argo CD.

EDITOR: puede crear y modificar aplicaciones. No puede cambiar la configuración de Argo CD ni eliminar aplicaciones.

VIEWER: acceso de solo lectura a las aplicaciones. Puede ver el estado y el historial de la aplicación. No puede hacer cambios.

> **Note**
>
> Los nombres de los roles distinguen entre mayúsculas y minúsculas y deben estar en mayúsculas (ADMIN, EDITOR, VIEWER).

> **Important**
>
> La integración de capacidades de EKS con AWS Identity Center admite hasta 1000 identidades por capacidad de Argo CD. Una identidad puede ser un usuario o un grupo.

Implementaciones de varios clústeres

La capacidad administrada de Argo CD admite implementaciones de varios clústeres, lo que le permite administrar las aplicaciones en los clústeres de desarrollo, prueba y producción desde una única instancia de Argo CD.

Cómo funcionan varios clústeres

Al registrar clústeres adicionales con Argo CD:

1. Crea secretos de clúster que hacen referencia a los clústeres de EKS de destino por ARN
2. Se crean aplicaciones o ApplicationSets que tienen distintos clústeres como destino
3. Argo CD se conecta a cada clúster para implementar aplicaciones y ver los recursos
4. Puede ver y administrar todos los clústeres desde una única interfaz de usuario de Argo CD

Requisitos previos para varios clústeres

Antes de registrar clústeres adicionales:

- Cree una entrada de acceso en el clúster de destino para el rol de capacidad de Argo CD
- Garantice la conectividad de red entre la capacidad de Argo CD y los clústeres de destino
- Compruebe los permisos de IAM para acceder a los clústeres de destino

Registro de un clúster

Registre los clústeres mediante Kubernetes Secrets en el espacio de nombres `argocd`.

Obtenga el ARN del clúster de destino. Reemplace *`region-code`* por la región de AWS en la que está el clúster de destino y sustituya *`target-cluster`* por el nombre del clúster de destino.

```
aws eks describe-cluster \
  --region region-code \
  --name target-cluster \
  --query 'cluster.arn' \
  --output text
```

Cree un secreto de clúster mediante el ARN del clúster:

```
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**
>
> Use el ARN del clúster de EKS en el campo `server`, no la URL del servidor de API de Kubernetes. La capacidad administrada requiere que los ARN identifiquen los clústeres de destino.

Aplique el secreto:

```
kubectl apply -f cluster-secret.yaml
```

Configuración de la entrada de acceso en el clúster de destino

El clúster de destino debe tener una entrada de acceso que conceda al rol de capacidad de Argo CD permiso para implementar aplicaciones. Reemplace *region-code* por la región de AWS en la que está el clúster de destino, sustituya *target-cluster* por el nombre del clúster de destino y el ARN por el ARN del rol de capacidad de Argo CD.

```
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**
>
> Para su uso en producción, considere la posibilidad de utilizar grupos de Kubernetes más restrictivos en lugar de `system:masters`.

Acceso a clústeres privados

La capacidad administrada de Argo CD se puede implementar en clústeres de EKS totalmente privados sin necesidad de emparejamiento de VPC ni una configuración de red especializada. AWS administra automáticamente la conectividad entre la capacidad de Argo CD y los clústeres remotos privados.

Implementaciones entre cuentas

Para implementaciones entre cuentas, agregue el rol de capacidad de IAM de Argo CD desde la cuenta de origen a la entrada de acceso de EKS del clúster de destino:

1. En la cuenta de destino, cree una entrada de acceso en el clúster de EKS de destino
2. Utilice el ARN del rol de capacidad de IAM de Argo CD de la cuenta de origen como entidad principal
3. Configure los permisos de RBAC de Kubernetes adecuados para la entrada de acceso
4. Registre el clúster de destino en Argo CD con su ARN de clúster de EKS

No es necesario crear roles de IAM adicionales ni configurar ninguna política de confianza: las entradas de acceso de EKS gestionan el acceso entre cuentas.

Prácticas recomendadas

Utilice orígenes declarativas como origen de información verdadera: almacene todos los manifiestos de aplicaciones en repositorios de Git, registros de Helm o imágenes OCI, lo que permitirá el control de versiones, los registros de auditoría y la colaboración.

Implemente un RBAC adecuado: utilice la integración de AWS Identity Center para controlar quién puede acceder a las aplicaciones de Argo CD y administrarlas. Argo CD admite un control de acceso detallado a los recursos dentro de las aplicaciones (implementaciones, pods, ConfigMaps, secretos).

Utilice ApplicationSets para implementaciones en varios entornos: utilice ApplicationSets para implementar aplicaciones en varios clústeres o espacios de nombres con diferentes configuraciones.

Administración del ciclo de vida

Políticas de sincronización de aplicaciones

Controle la forma en que Argo CD sincroniza las aplicaciones:

Sincronización manual: las aplicaciones requieren una aprobación manual para sincronizar los cambios. Recomendada para entornos de producción.

Sincronización automática: las aplicaciones se sincronizan automáticamente cuando se detectan cambios en Git. Común en entornos de desarrollo y prueba.

Recuperación automática: se revierten automáticamente los cambios manuales hechos en el clúster. Garantiza que el estado del clúster coincida con Git.

Poda: se eliminan automáticamente los recursos eliminados de Git. Úselo con precaución, ya que puede eliminar recursos.

Estado de la aplicación

Argo CD supervisa continuamente el estado de las aplicaciones:

- En buen estado: todos los recursos se ejecutan según lo esperado
- En curso: los recursos se están creando o actualizando
- Degradado: algunos recursos no están en buen estado
- Suspendido: la aplicación está en pausa

- Falta: faltan recursos en el clúster

Plazos de sincronización

Configure los plazos de sincronización para controlar cuándo se pueden sincronizar las aplicaciones:

- Permita las sincronizaciones solo durante los periodos de mantenimiento
- Bloquee las sincronizaciones durante el horario laboral
- Programe sincronizaciones automáticas para horarios específicos
- Use los plazos de sincronización para situaciones de emergencia en las que tenga que detener temporalmente todas las sincronizaciones

Configuración de webhooks para una sincronización más rápida

De forma predeterminada, Argo CD sondea los repositorios de Git cada 6 minutos para detectar cambios. Para implementaciones con mayor capacidad de respuesta, configure los webhooks de Git para que activen sincronizaciones inmediatas cuando se inserten cambios.

Los webhooks proporcionan varios beneficios:

- Respuesta de sincronización inmediata cuando se inserta el código (segundos en lugar de minutos)
- Reducción de la sobrecarga de sondeo y mejora del rendimiento del sistema
- Uso más eficiente de los límites de velocidad de la API
- Mejor experiencia de usuario con comentarios más rápidos

Punto de conexión de webhook

La capacidad de Argo CD proporciona un punto de conexión de webhook para recibir notificaciones de Git. Busque la URL del webhook en la consola de EKS, en la pestaña Capacidades del clúster, o recupérela mediante la AWS CLI:

```
aws eks describe-capability \
  --cluster-name my-cluster \
  --capability-name my-argocd \
  --query 'capability.configuration.argoCd.webhookUrl' \
  --output text \
```

```
  --region region-code
```

Configuración de webhooks por proveedor de Git

GitHub: en la configuración del repositorio, agregue un webhook con la URL del webhook de Argo CD. Establezca el tipo de contenido en `application/json` y seleccione “Solo el evento de inserción”.

GitLab: en la configuración del proyecto, agregue un webhook con la URL del webhook de Argo CD. Active “Eventos de inserción” y, opcionalmente, “Etiquetar eventos de inserción”.

Bitbucket: en la configuración del repositorio, agregue un webhook con la URL del webhook de Argo CD. Seleccione “Inserción del repositorio” como activador.

CodeCommit: cree una regla de Amazon EventBridge que active los cambios de estado del repositorio de CodeCommit y envíe notificaciones al punto de conexión del webhook de Argo CD.

Para obtener instrucciones detalladas sobre la configuración de webhooks, consulte Argo CD Webhook Configuration.

> **Note**
>
> Los webhooks complementan los sondeos, no las sustituyen. Argo CD sigue sondeando los repositorios como mecanismo alternativo en caso de que se pierdan las notificaciones de los webhooks.

Siguientes pasos

- the section called “Uso de Argo CD”: más información sobre cómo crear y administrar aplicaciones de Argo CD
- the section called “Solución de problemas”: resolución de problemas con Argo CD
- the section called “Uso de capacidades”: administración del recurso de la capacidad de Argo CD

Solución de problemas con capacidades de Argo CD

En este tema, se proporciona una guía para la solución de problemas de la capacidad de EKS para Argo CD, lo que incluye las comprobaciones de estado de la capacidad, los problemas de

sincronización de las aplicaciones, la autenticación de repositorios y las implementaciones en varios clústeres.

> **Note**
>
> Las capacidades de EKS son completamente administradas y se ejecutan fuera del clúster. No tiene acceso a los registros del servidor de Argo CD ni al espacio de nombres `argocd`. La solución de problemas se centra en el estado de la capacidad, el estado de la aplicación y la configuración.

La capacidad está ACTIVA, pero las aplicaciones no se sincronizan

Si la capacidad de Argo CD muestra el estado `ACTIVE`, pero las aplicaciones no se están sincronizando, compruebe el estado de la capacidad y de la aplicación.

Compruebe el estado de la capacidad:

Puede ver los problemas de estado de la capacidad en la consola de EKS o mediante la AWS CLI.

Consola:

1. Abra la consola de Amazon EKS en https://console.aws.amazon.com/eks/home#/clusters.
2. Seleccione el nombre del clúster.
3. Seleccione la pestaña Observabilidad.
4. Elija Supervisar clúster.
5. Seleccione la pestaña Capacidades para ver el estado de todas las capacidades.

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
```

Causas habituales:

- Repositorio no configurado: el repositorio de Git no se agregó a Argo CD.
- Error de autenticación: las credenciales de la clave SSH, el token o CodeCommit no son válidas.
- Aplicación no creada: no hay recursos de la aplicación en el clúster.
- Política de sincronización: se requiere sincronización manual (la sincronización automática no está activada).
- Permisos de IAM: faltan permisos para CodeCommit o Secrets Manager.

Compruebe el estado de la aplicación:

```
# 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}'
```

Compruebe las condiciones de la aplicación:

```
# 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}'
```

Aplicaciones bloqueadas en el estado “En curso”

Si una aplicación muestra `Progressing`, pero nunca aparece `Healthy`, compruebe el estado de los recursos y los eventos de la aplicación.

Compruebe el estado de los recursos:

```
# 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"
```

Causas habituales:

- La implementación no está lista: los pods no se inician o las sondas de preparación fallan.
- Dependencias de recursos: recursos que esperan a que otros recursos estén listos.
- Errores al extraer imágenes: no se puede acceder a las imágenes del contenedor.
- Recursos insuficientes: el clúster carece de CPU o memoria para los pods.

Verifique la configuración del clúster de destino (para configuraciones de varios clústeres):

```
# 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
```

Errores de autenticación de repositorios

Si Argo CD no puede acceder a los repositorios de Git, compruebe la configuración de autenticación.

Para repositorios de CodeCommit:

Compruebe que el rol de capacidad de IAM tenga permisos de 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
```

El rol necesita el permiso `codecommit:GitPull` para los repositorios.

Para repositorios de Git privados:

Compruebe que las credenciales del repositorio estén configuradas correctamente:

```
# Check repository secret exists
kubectl get secret -n argocd repo-secret-name -o yaml
```

Asegúrese de que el secreto contenga las credenciales de autenticación correctas (clave SSH, token o nombre de usuario y contraseña).

Para los repositorios que utilizan Secrets Manager:

```
# 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
```

Problemas en implementaciones de varios clústeres

Si las aplicaciones no se implementan en clústeres remotos, compruebe la configuración de acceso y el registro del clúster.

Compruebe el registro del clúster:

```
# 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
```

Asegúrese de que el campo `server` contenga el ARN del clúster de EKS, no la URL de la API de Kubernetes.

Compruebe la entrada de acceso del clúster de destino:

En el clúster de destino, compruebe que el rol de capacidad de Argo CD tenga una entrada de acceso:

```
# 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
```

Compruebe los permisos de IAM para varias cuentas:

Para las implementaciones entre cuentas, compruebe que el rol de capacidad de Argo CD tenga una entrada de acceso en el clúster de destino. La capacidad administrada utiliza las entradas de acceso de EKS para el acceso entre cuentas, no para asumir el rol de IAM.

Para obtener más información sobre la configuración de varios clústeres, consulte the section called “Registro de clústeres”.

Siguientes pasos

- the section called “Consideraciones sobre Argo CD”: consideraciones y prácticas recomendadas de Argo CD
- the section called “Uso de Argo CD”: creación y administración de aplicaciones de Argo CD
- the section called “Registro de clústeres”: configuración de implementaciones de varios clústeres
- the section called “Solución de problemas de capacidades”: orientación general de solución de problemas de la capacidad

Comparación de la capacidad de EKS para Argo CD con Argo CD autoadministrado

La capacidad de EKS para Argo CD proporciona una experiencia de Argo CD completamente administrada que se ejecuta en EKS. Para obtener una comparación general entre las capacidades de EKS y las soluciones autoadministradas, consulte the section called “Consideraciones”. Este tema se centra en las diferencias específicas de Argo CD, lo que incluye la autenticación, la administración de varios clústeres y la compatibilidad con características ascendentes.

Diferencias con Argo CD ascendente

La capacidad de EKS para Argo CD se basa en Argo CD ascendente, pero difiere en la forma en que se accede a él, se configura y se integra con los servicios de AWS.

RBAC y autenticación: la capacidad incluye tres roles de RBAC (administrador, editor y lector) y utiliza AWS Identity Center para la autenticación en lugar de la autenticación integrada de Argo CD. Configure las asignaciones de roles a través del parámetro `rbacRoleMapping` de la capacidad para

asignar grupos de Identity Center a roles de Argo CD, no a través del ConfigMap `argocd-rbac-cm` de Argo CD. La interfaz de usuario de Argo CD se aloja con su propia URL directa (búsquela en la consola de EKS, en la pestaña Capacidades del clúster), y el acceso a la API utiliza la autenticación y la autorización de AWS a través de IAM.

Configuración de clústeres: esta funcionalidad no configura automáticamente las topologías de clústeres locales ni radiales. Usted configura los clústeres de destino de la implementación y las entradas de acceso de EKS. La capacidad solo admite clústeres de Amazon EKS como destinos de implementación que utilizan ARN de clústeres de EKS (no URL de servidores de API de Kubernetes). La capacidad no agrega automáticamente el clúster local (`kubernetes.default.svc`) como destino de implementación. Para implementar en el mismo clúster en el que se creó la capacidad, registre ese clúster de forma explícita mediante su ARN.

Acceso remoto simplificado a los clústeres: esta capacidad simplifica las implementaciones de varios clústeres al utilizar las entradas de acceso de EKS para conceder a Argo CD acceso a los clústeres remotos, lo que elimina la necesidad de configurar roles de IAM para cuentas de servicio (IRSA) o establecer suposiciones de roles de IAM entre cuentas. La capacidad también proporciona un acceso transparente a clústeres de EKS totalmente privados sin necesidad de emparejamiento de VPC ni una configuración de red especializada. AWS administra automáticamente la conectividad entre la capacidad de Argo CD y los clústeres remotos privados.

Integración directa de servicios de AWS: esta capacidad proporciona una integración directa con los servicios de AWS mediante los permisos de IAM del rol de capacidad. Puede hacer referencia a repositorios de CodeCommit, gráficos de Helm de ECR y CodeConnections directamente en los recursos de la aplicación sin necesidad de crear configuraciones de repositorio. Esto simplifica la autenticación y elimina la necesidad de administrar credenciales independientes para los servicios de AWS. Para obtener más información, consulte the section called “Configuración de repositorios”.

Compatibilidad con espacios de nombres: inicialmente, la capacidad admite la implementación de aplicaciones en un único espacio de nombres, que especificará al crear la capacidad. Es posible que, en futuras versiones, se agregue soporte para aplicaciones en varios espacios de nombres.

Características no compatibles: las siguientes características no están disponibles en la capacidad administrada:

- Complementos de administración de configuración (CMP) para la generación de manifiestos personalizados
- Scripts de Lua personalizados para evaluar el estado de los recursos (se admiten las comprobaciones de estado integradas para recursos estándar)

- Controlador de notificaciones
- Actualizador de imágenes de Argo CD
- Proveedores de SSO personalizados (solo se admite AWS Identity Center, lo que incluye la identidad federada de terceros a través de AWS Identity Center)
- Extensiones de interfaz de usuario y banners personalizados
- Acceso directo a `argocd-cm`, `argocd-params` y otros ConfigMaps de configuración

Compatibilidad: las aplicaciones y los ApplicationSets funcionan de forma idéntica a Argo CD ascendente sin cambios en los manifiestos. La capacidad utiliza las mismas API y CRD de Kubernetes, por lo que herramientas como `kubectl` funcionan de la misma manera. La capacidad es totalmente compatible con aplicaciones y ApplicationSets, flujos de trabajo de GitOps con sincronización automática, implementaciones de varios clústeres, políticas de sincronización (automatizadas, de poda, de autorreparación), enlaces y ondas de sincronización, evaluación del estado de los recursos de Kubernetes estándar, capacidades de reversión, orígenes de repositorios de Git (HTTPS y SSH), manifiestos de Helm, Kustomize y sin formato de YAML, credenciales de la aplicación de GitHub, proyectos para multitenencia y exclusiones e inclusiones de recursos.

Uso de la CLI de Argo CD con la capacidad administrada

La CLI de Argo CD funciona igual que Argo CD ascendente para la mayoría de las operaciones, pero la autenticación y el registro de clústeres son diferentes.

Requisitos previos

Instale la CLI de Argo CD según las instrucciones de instalación ascendente.

Configuración

Configure la CLI mediante variables de entorno:

1. Obtenga la URL del servidor de Argo CD desde la consola de EKS (en la pestaña Capacidades del clúster) o mediante la AWS CLI:

```
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)
```

2. Genere un token de cuenta desde la interfaz de usuario de Argo CD (Configuración → Cuentas → Administración → Generar nuevo token) y configúrelo como una variable de entorno:

```
export ARGOCD_AUTH_TOKEN="your-token-here"
```

> **Important**
>
> Esta configuración utiliza el token de la cuenta de administrador para los flujos de trabajo de configuración y desarrollo iniciales. Para casos de uso de producción, utilice tokens y roles del ámbito del proyecto para seguir el principio de privilegio mínimo. Para obtener más información sobre la configuración de RBAC y roles del proyecto, consulte the section called "Configuración de permisos".

1. Establezca la opción de gRPC requerida:

```
export ARGOCD_OPTS="--grpc-web"
```

Con estas variables de entorno configuradas, puede utilizar la CLI de Argo CD sin el comando `argocd login`.

Diferencias clave

La capacidad administrada tiene las siguientes limitaciones de CLI:

- Los comandos `argocd admin` no son compatibles (requieren el acceso directo a pods).
- `argocd login` no es compatible (utilice tokens de proyecto o cuenta como alternativa).
- `argocd cluster add` requiere el indicador `--aws-cluster-name` con el ARN del clúster de EKS.

Ejemplo: registro de un clúster

Registre un clúster de EKS para la implementación de la aplicación:

```
# 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
```

Para obtener la documentación completa de la CLI de Argo CD, consulte la referencia de la CLI de Argo CD.

Ruta de migración

Puede migrar de Argo CD autoadministrado a la capacidad administrada:

1. Revise la configuración actual de Argo CD para comprobar si hay características no compatibles (controlador de notificaciones, CMP, comprobaciones de estado personalizadas).
2. Escale sus controladores de Argo CD autoadministrados a cero réplicas.
3. Cree un recurso de la capacidad de Argo CD en su clúster.
4. Exporte sus aplicaciones, ApplicationSets y AppProjects.
5. Migre las credenciales de repositorios, los secretos de clústeres y las plantillas de credenciales de repositorios (repocreds).
6. Si utiliza claves GPG, certificados TLS o hosts SSH conocidos, migre también estas configuraciones.
7. Actualice los campos `destination.server` para usar los nombres de los clústeres o los ARN de los clústeres de EKS.
8. Aplíquelos a la instancia de Argo CD administrado.
9. Compruebe que las aplicaciones se estén sincronizando correctamente.
10. Desinstale su instalación de Argo CD autoadministrado.

La capacidad administrada utiliza las mismas API y definiciones de recursos de Argo CD, por lo que sus manifiestos existentes funcionan con modificaciones mínimas.

Siguientes pasos

- the section called “Creación de la capacidad Argo CD”: creación de un recurso de capacidad de Argo CD
- the section called “Uso de Argo CD”: implementación de su primera aplicación
- the section called “Consideraciones sobre Argo CD”: configuración de la integración de AWS Identity Center

Composición de recursos con kro (Kube Resource Orchestrator)

kro (Kube Resource Orchestrator) es un proyecto de código abierto nativo de Kubernetes que le permite definir las API de Kubernetes personalizadas mediante una configuración sencilla y directa. Con kro, puede configurar fácilmente nuevas API personalizadas que crean un grupo de objetos de Kubernetes y las operaciones lógicas entre ellos.

Con las capacidades de EKS, AWS administra kro completamente, lo que elimina la necesidad de instalar, mantener y escalar los controladores de kro en sus clústeres.

Cómo funciona kro

kro presenta una definición de recursos personalizados (CRD) llamada `ResourceGraphDefinition` (RGD) que permite la creación sencilla y simplificada de API de Kubernetes personalizadas. Al crear una `ResourceGraphDefinition`, kro utiliza extensiones nativas de Kubernetes para crear y administrar nuevas API en el clúster. A partir de esta especificación de recurso único, kro creará y registrará una nueva CRD en función de la especificación y se adaptará para administrar los recursos personalizados recién definidos.

Las RGD pueden incluir varios recursos y kro determinará las interdependencias y el orden de los recursos para que no tenga que hacerlo usted. Puede utilizar una sintaxis sencilla para inyectar la configuración de un recurso a otro, lo que simplifica considerablemente las composiciones y elimina la necesidad de utilizar operadores de “glue” en el clúster. Con kro, los recursos personalizados pueden incluir recursos nativos de Kubernetes, así como cualquier definición de recursos personalizados (CRD) instalada en el clúster.

kro admite un solo tipo de recurso principal:

- ResourceGraphDefinition (RGD): define un recurso personalizado de Kubernetes y encapsula uno o más recursos de Kubernetes nativos o personalizados subyacentes.

Además de este recurso, kro creará y administrará el ciclo de vida de los recursos personalizados que se creen con él, así como todos los recursos que lo componen.

kro se integra sin problemas con Controladores de AWS para Kubernetes (ACK), lo que le permite componer recursos de carga de trabajo con recursos de AWS para crear abstracciones de alto nivel. Esto le permite crear sus propios componentes básicos en la nube, lo que simplifica la administración de recursos y permite crear patrones reutilizables con valores de configuración predeterminados e inmutables basados en sus estándares organizativos.

Beneficios de kro

kro permite a los equipos de plataformas crear API de Kubernetes personalizadas que componen varios recursos en abstracciones de nivel superior. Esto simplifica la administración de recursos al permitir a los desarrolladores implementar aplicaciones complejas mediante recursos personalizados sencillos, estandarizados y versionados. Definirá patrones reutilizables para las combinaciones de recursos comunes, lo que permite una creación de recursos coherente en toda la organización.

kro utiliza el lenguaje de expresión común (CEL) en Kubernetes para transferir valores entre los recursos e incorporar la lógica condicional, lo que proporciona flexibilidad en la composición de los recursos. Puede componer los recursos de Kubernetes y de AWS administrados por ACK en API personalizadas unificadas, lo que permite definir todas las aplicaciones e infraestructuras.

kro admite la configuración declarativa a través de los manifiestos de Kubernetes, lo que permite que los flujos de trabajo de GitOps y las prácticas de infraestructura como código se integren sin problemas con los procesos de desarrollo existentes. Como parte de las capacidades administradas de EKS, AWS administra kro completamente, lo que elimina la necesidad de instalar, configurar y mantener los controladores kro en sus clústeres.

Ejemplo: creación de una ResourceGraphDefinition

En el siguiente ejemplo, se muestra una `ResourceGraphDefinition` sencilla que crea una aplicación web con una implementación y un servicio:

```
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}
```

Cuando los usuarios crean instancias del recurso personalizado de `WebApplication`, kro crea automáticamente los recursos de implementación y servicio correspondientes y administra su ciclo de vida junto con el recurso personalizado.

Integración con otras capacidades administradas de EKS

kro se integra con otras capacidades administradas de EKS.

- Controladores de AWS para Kubernetes (ACK): utilice kro para componer los recursos de ACK en abstracciones de nivel superior, lo que simplifica la administración de los recursos de AWS.
- Argo CD: utilice Argo CD para administrar la implementación de los recursos personalizados de kro en varios clústeres, lo que activa los flujos de trabajo de GitOps para las pilas de aplicaciones y los componentes básicos de la plataforma.

Introducción a kro

Para comenzar a utilizar la capacidad de EKS para kro:

1. Cree un recurso de capacidad de kro en su clúster de EKS a través de la consola de AWS, la AWS CLI o su infraestructura preferida como herramienta de código.

2. Cree ResourceGraphDefinitions (RGD) que definan sus composiciones de recursos y API personalizadas.
3. Aplique instancias de los recursos personalizados para aprovisionar y administrar los recursos de AWS y Kubernetes subyacentes.

Creación de una capacidad de kro

En este tema, se explica cómo crear una capacidad de kro en un clúster de Amazon EKS.

Requisitos previos

Antes de crear una capacidad de kro, asegúrese de que disponga de lo siguiente:

- Un clúster de Amazon EKS existente que ejecute una versión de Kubernetes compatible (se admiten todas las versiones con soporte estándar y ampliado)
- Permisos de IAM suficientes para crear recursos de capacidad en los clústeres de EKS
- (Para la CLI o eksctl) La herramienta de la CLI adecuada instalada y configurada

> **Note**
>
> A diferencia de ACK y Argo CD, kro no necesita permisos de IAM adicionales más allá de la política de confianza. kro opera completamente dentro de su clúster y no hace llamadas a la API de AWS. Sin embargo, debe proporcionar igualmente un rol de capacidad de IAM con la política de confianza adecuada. Para obtener información acerca de la configuración de permisos de RBAC de Kubernetes para kro, consulte the section called “Configuración de permisos”.

Elección de la herramienta

Puede crear una capacidad de kro mediante la Consola de administración de AWS, la AWS CLI o eksctl:

- the section called “Consola”: uso de la consola para una experiencia guiada
- the section called “ AWS CLI”: uso de la AWS CLI para scripts y automatización
- the section called “eksctl”: uso de eksctl para una experiencia nativa de Kubernetes

Qué ocurre cuando se crea una capacidad de kro

Al crear una capacidad de kro:

1. EKS crea el servicio de capacidad de kro y lo configura para supervisar y administrar los recursos del clúster.
2. Las definiciones de recursos personalizados (CRD) de Kubernetes se instalan en el clúster.
3. La capacidad asume el rol de capacidad de IAM que proporcione (se utiliza solo para la relación de confianza).
4. kro comienza a supervisar los recursos de `ResourceGraphDefinition` y sus instancias.
5. El estado de la capacidad cambia de `CREATING` a `ACTIVE`.

Una vez activo, puede crear ResourceGraphDefinitions para definir las API personalizadas y crear instancias de esas API.

Siguientes pasos

Después de crear la capacidad de kro:

- the section called “Conceptos de kro”: descripción de los conceptos de kro y la composición de recursos
- the section called “Conceptos de kro”: más información sobre SimpleSchema, las expresiones de CEL y los patrones de composición de recursos

Creación de una capacidad de kro mediante la consola

En este tema, se describe cómo crear una capacidad de kro (Kube Resource Orchestrator) mediante la Consola de administración de AWS.

Creación de la capacidad de kro

1. Abra la consola de Amazon EKS en https://console.aws.amazon.com/eks/home#/clusters.
2. Seleccione el nombre del clúster para abrir la página de detalles del clúster.
3. Elija la pestaña Capacidades.
4. En el menú de navegación izquierdo, seleccione kro (Kube Resource Orchestrator).
5. Elija Crear capacidad de kro.
6. Para el rol de capacidad de IAM:

- Si ya tiene un rol de capacidad de IAM, selecciónelo en el menú desplegable.
- Si necesita crear un rol, elija Crear rol de kro.

  De este modo, se abre la consola de IAM en una nueva pestaña con la política de confianza rellenada previamente. El rol no necesita permisos de IAM adicionales, ya que kro opera completamente dentro del clúster.

  Después de crear el rol, regrese a la consola de EKS y el rol se seleccionará automáticamente.

  > **Note**
  >
  > A diferencia de ACK y Argo CD, kro no necesita permisos de IAM adicionales más allá de la política de confianza. kro opera completamente dentro de su clúster y no hace llamadas a la API de AWS.

7. Seleccione Crear.

Comenzará el proceso de creación de la capacidad.

Comprobación de la activación de la capacidad

1. En la pestaña Capacidades, consulte el estado de la capacidad de kro.
2. Espere a que el estado cambie de `CREATING` a `ACTIVE`.
3. Una vez activa, la capacidad está lista para usarse.

Para obtener información sobre los estados de la capacidad y la solución de problemas, consulte the section called “Uso de capacidades”.

Concesión de permisos para administrar los recursos de Kubernetes

De forma predeterminada, kro solo puede crear y administrar ResourceGraphDefinitions y sus instancias. Para permitir que kro cree y administre los recursos de Kubernetes subyacentes definidos en su ResourceGraphDefinitions, asocie la política de acceso `AmazonEKSClusterAdminPolicy` a la entrada de acceso de la capacidad.

1. En la consola de EKS, vaya a la pestaña Acceso del clúster.
2. En Entradas de acceso, busque la entrada para el rol de capacidad de kro (tendrá el ARN del rol que creó anteriormente).

3. Elija la entrada de acceso para abrir sus detalles.
4. En la sección Políticas de acceso, elija Asociar política de acceso.
5. Seleccione `AmazonEKSClusterAdminPolicy` en la lista de políticas.
6. En Ámbito de acceso, seleccione Clúster.
7. Elija Asociar.

> **Important**
>
> La `AmazonEKSClusterAdminPolicy` otorga amplios permisos para crear y administrar todos los recursos de Kubernetes y su objetivo es simplificar la puesta en marcha. Para su uso en producción, cree políticas de RBAC más restrictivas que otorguen solo los permisos necesarios para los recursos específicos que administrará su ResourceGraphDefinitions. Para obtener orientación sobre cómo configurar los permisos de privilegio mínimo, consulte the section called “Configuración de permisos” y the section called “Consideraciones para las capacidades de EKS”.

Comprobación de la disponibilidad de los recursos personalizados

Una vez que la capacidad esté activa, compruebe que los recursos personalizados de kro estén disponibles en el clúster.

Uso de la consola

1. Vaya a su clúster en la consola de Amazon EKS.
2. Elija la pestaña Recursos.
3. Elija Extensiones.
4. Elija CustomResourceDefinitions.

Debería ver el tipo de recursos `ResourceGraphDefinition` en la lista.

Uso de kubectl

```
kubectl api-resources | grep kro.run
```

Debería ver el tipo de recursos `ResourceGraphDefinition` en la lista.

Siguientes pasos

- the section called “Conceptos de kro”: descripción de los conceptos de kro y la composición de recursos
- the section called “Conceptos de kro”: más información sobre SimpleSchema, las expresiones de CEL y los patrones de composición
- the section called “Uso de capacidades”: administración del recurso de la capacidad de kro

Creación de una capacidad de kro mediante la AWS CLI

En este tema, se describe cómo crear una capacidad de kro (Kube Resource Orchestrator) mediante la AWS CLI.

Requisitos previos

- AWS CLI: versión `2.12.3` o posterior. Para comprobar la versión, ejecute `aws --version`. Para obtener más información, consulte Instalación en la Guía del usuario de la interfaz de la línea de comandos de AWS.
- **`kubectl`** – una herramienta de línea de comandos para trabajar con clústeres de Kubernetes. Para obtener más información, consulte the section called “Configure kubectl y eksctl”.

Paso 1: creación de un rol de capacidad de IAM

Cree un archivo de política de confianza:

```
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
```

Cree el rol de IAM:

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

> **Note**
>
> A diferencia de ACK y Argo CD, kro no necesita permisos de IAM adicionales. kro opera completamente dentro de su clúster y no hace llamadas a la API de AWS. El rol solo es necesario para establecer una relación de confianza con el servicio de capacidades de EKS.

Paso 2: creación de la capacidad de kro

Cree el recurso de la capacidad de kro en su clúster. Reemplace *region-code* por la región de AWS en la que se encuentra el clúster (como `us-west-2`) y *my-cluster* por el nombre del clúster.

```
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
```

El comando devuelve una respuesta inmediatamente, pero la capacidad tarda algún tiempo en activarse mientras EKS crea la infraestructura y los componentes de la capacidad necesarios. EKS instalará las definiciones de recursos personalizados de Kubernetes relacionadas con esta capacidad en el clúster según se vaya creando.

> **Note**
>
> Si recibe un error que indica que el clúster no existe o que no tiene permisos, compruebe lo siguiente:

- El nombre del clúster es correcto
- La AWS CLI está configurada para la región correcta
- Dispone de los permisos de IAM necesarios

Paso 3: comprobación de la activación de la capacidad

Espere a que se active la capacidad. Reemplace *region-code* por la región de AWS donde creó el clúster y *my-cluster* por el nombre de su clúster.

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-kro \
  --query 'capability.status' \
  --output text
```

La capacidad estará lista cuando aparezca el estado `ACTIVE`.

También puede ver todos los detalles de la capacidad:

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-kro
```

Paso 4: concesión de permisos para administrar los recursos de Kubernetes

De forma predeterminada, kro solo puede crear y administrar ResourceGraphDefinitions y sus instancias. Para permitir que kro cree y administre los recursos de Kubernetes subyacentes definidos en su ResourceGraphDefinitions, asocie la política de acceso `AmazonEKSClusterAdminPolicy` a la entrada de acceso de la capacidad.

Obtenga el ARN del rol de capacidad:

```
CAPABILITY_ROLE_ARN=$(aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-kro \
```

```
  --query 'capability.roleArn' \
  --output text)
```

Asocie la política de administración del clúster:

```
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

La `AmazonEKSClusterAdminPolicy` otorga amplios permisos para crear y administrar todos los recursos de Kubernetes y su objetivo es simplificar la puesta en marcha. Para su uso en producción, cree políticas de RBAC más restrictivas que otorguen solo los permisos necesarios para los recursos específicos que administrará su ResourceGraphDefinitions. Para obtener orientación sobre cómo configurar los permisos de privilegio mínimo, consulte the section called “Configuración de permisos” y the section called “Consideraciones para las capacidades de EKS”.

Paso 5: comprobación de la disponibilidad de los recursos personalizados

Una vez que la capacidad esté activa, compruebe que los recursos personalizados de kro estén disponibles en el clúster:

```
kubectl api-resources | grep kro.run
```

Debería ver el tipo de recursos `ResourceGraphDefinition` en la lista.

Siguientes pasos

- the section called “Conceptos de kro”: descripción de los conceptos de kro y la composición de recursos
- the section called “Conceptos de kro”: más información sobre SimpleSchema, las expresiones de CEL y los patrones de composición

- the section called “Uso de capacidades”: administración del recurso de la capacidad de kro

Creación de una capacidad de kro mediante eksctl

En este tema, se describe cómo crear una capacidad de kro (Kube Resource Orchestrator) mediante eksctl.

> **Note**
>
> Los siguientes pasos requieren la versión `0.220.0` o posterior de eksctl. Para comprobar la versión, ejecute `eksctl version`.

Paso 1: creación de un rol de capacidad de IAM

Cree un archivo de política de confianza:

```
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
```

Cree el rol de IAM:

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

> **Note**
>
> A diferencia de ACK y Argo CD, kro no necesita permisos de IAM adicionales más allá de la política de confianza. kro opera completamente dentro de su clúster y no hace llamadas a la API de AWS.

Paso 2: creación de la capacidad de kro

Cree la capacidad de kro mediante eksctl. Reemplace *region-code* por la región de AWS donde creó el clúster y *my-cluster* por el nombre de su clúster.

```
eksctl create capability \
  --region region-code \
  --cluster my-cluster \
  --name my-kro \
  --type KRO \
  --role-arn arn:aws:iam::[.replaceable]111122223333:role/KROCapabilityRole
```

El comando vuelve inmediatamente, pero la capacidad tarda algún tiempo en activarse.

Paso 3: comprobación de la activación de la capacidad

Compruebe el estado de la capacidad. Reemplace *region-code* por la región de AWS donde creó el clúster y *my-cluster* por el nombre de su clúster.

```
eksctl get capability \
  --region region-code \
  --cluster my-cluster \
  --name my-kro
```

La capacidad estará lista cuando aparezca el estado `ACTIVE`.

Paso 4: concesión de permisos para administrar los recursos de Kubernetes

De forma predeterminada, kro solo puede crear y administrar ResourceGraphDefinitions y sus instancias. Para permitir que kro cree y administre los recursos de Kubernetes subyacentes definidos en su ResourceGraphDefinitions, asocie la política de acceso `AmazonEKSClusterAdminPolicy` a la entrada de acceso de la capacidad.

Obtenga el ARN del rol de capacidad:

```
CAPABILITY_ROLE_ARN=$(aws eks describe-capability \
  --region region-code \
  --cluster my-cluster \
  --name my-kro \
  --query 'capability.roleArn' \
  --output text)
```

Asocie la política de administración del clúster:

```
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

La `AmazonEKSClusterAdminPolicy` otorga amplios permisos para crear y administrar todos los recursos de Kubernetes y su objetivo es simplificar la puesta en marcha. Para su uso en producción, cree políticas de RBAC más restrictivas que otorguen solo los permisos necesarios para los recursos específicos que administrará su ResourceGraphDefinitions. Para obtener orientación sobre cómo configurar los permisos de privilegio mínimo, consulte the section called “Configuración de permisos” y the section called “Consideraciones para las capacidades de EKS”.

Paso 5: comprobación de la disponibilidad de los recursos personalizados

Una vez que la capacidad esté activa, compruebe que los recursos personalizados de kro estén disponibles en el clúster:

```
kubectl api-resources | grep kro.run
```

Debería ver el tipo de recursos `ResourceGraphDefinition` en la lista.

Siguientes pasos

- the section called “Conceptos de kro”: descripción de los conceptos de kro y la composición de recursos

- the section called “Conceptos de kro”: más información sobre SimpleSchema, las expresiones de CEL y los patrones de composición
- the section called “Uso de capacidades”: administración del recurso de la capacidad de kro

Conceptos de kro

kro permite a los equipos de plataformas crear API de Kubernetes personalizadas que componen varios recursos en abstracciones de nivel superior. En este tema, se muestra un ejemplo práctico y, a continuación, se explican los conceptos básicos que debe comprender al trabajar con la capacidad de EKS para kro.

Introducción a kro

Después de crear la capacidad de kro (consulte the section called “Creación de una capacidad de kro”), puede comenzar a crear API personalizadas con ResourceGraphDefinitions en el clúster.

A continuación se muestra un ejemplo completo que crea una abstracción simple de una aplicación web:

```
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
```

Después de aplicar esta ResourceGraphDefinition, los equipos de aplicaciones pueden crear aplicaciones web mediante su API simplificada:

```
apiVersion: kro.run/v1alpha1
kind: WebApplication
metadata:
  name: my-app
spec:
  name: my-app
  replicas: 5
```

kro crea automáticamente la implementación y el servicio con la configuración adecuada. Como no se ha especificado `image`, utiliza el valor predeterminado `nginx:latest` del esquema.

Conceptos clave

> **Important**
>
> kro valida ResourceGraphDefinitions en el momento de la creación, no en tiempo de ejecución. Al crear una RGD, kro valida la sintaxis de CEL, compara las expresiones con los esquemas reales de Kubernetes, verifica la existencia de los campos y detecta las dependencias circulares. Esto significa que los errores se detectan inmediatamente al crear la RGD, antes de implementar cualquier instancia.

ResourceGraphDefinition

Una ResourceGraphDefinition (RGD) define una API de Kubernetes personalizada mediante la especificación de lo siguiente:

- Esquema: la estructura de la API con el formato SimpleSchema (nombres de campo, tipos, valores predeterminados, validación)
- Recursos: plantillas para la creación para recursos de AWS o Kubernetes subyacentes
- Dependencias: cómo se relacionan los recursos entre sí (se detectan automáticamente a partir de referencias de campo)

Al aplicar una RGD, kro registra una nueva definición de recursos personalizados (CRD) en el clúster. A continuación, los equipos de aplicaciones pueden crear instancias de la API personalizada y kro se encarga de crear y administrar todos los recursos subyacentes.

Para obtener más información, consulte la sección ResourceGraphDefinition Overview en la documentación de kro.

Formato SimpleSchema

SimpleSchema proporciona una forma simplificada de definir esquemas de API sin necesidad de conocimientos de OpenAPI:

```
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 admite los tipos `string`, `integer`, `boolean` y `number` con restricciones como `required`, `default`, `minimum`/`maximum`, `enum` y `pattern`.

Para obtener más información, consulte SimpleSchema en la documentación de kro.

Expresiones de CEL

kro utiliza el lenguaje de expresión común (CEL) para hacer referencia a valores de forma dinámica y agregar lógica condicional. Las expresiones de CEL se encapsulan en `${` y `}` y se pueden usar de dos maneras:

Expresiones independientes. Todo el valor del campo es una sola expresión:

```
spec:
  replicas: ${schema.spec.replicaCount}  # Expression returns integer
  labels: ${schema.spec.labelMap}        # Expression returns object
```

El resultado de la expresión reemplaza todo el valor del campo y debe coincidir con el tipo esperado del campo.

Plantillas de cadenas. Una o más expresiones incrustadas en una cadena:

```
metadata:
  name: "${schema.spec.prefix}-${schema.spec.name}"  # Multiple expressions
  annotation: "Created by ${schema.spec.owner}"      # Single expression in string
```

Todas las expresiones de las plantillas de cadenas deben devolver cadenas. Utilice `string()` para convertir otros tipos: `"replicas-${string(schema.spec.count)}"`.

Referencias de campos. Acceda a los valores de las especificaciones de la instancia mediante `schema.spec`:

```
template:
  metadata:
    name: ${schema.spec.name}-deployment
    namespace: ${schema.metadata.namespace}  # Can also reference metadata
  spec:
    replicas: ${schema.spec.replicas}
```

Acceso a campos opcionales. Utilice ? para campos que podrían no existir:

```
# For ConfigMaps or Secrets with unknown structure
value: ${configmap.data.?DATABASE_URL}

# For optional status fields
ready: ${deployment.status.?readyReplicas > 0}
```

Si el campo no existe, la expresión devuelve `null` en lugar de producir un error.

Recursos condicionales. Incluya los recursos solo cuando se cumplan las condiciones:

```
resources:
- id: ingress
  includeWhen:
    - ${schema.spec.enableIngress == true}
  template:
    # ... ingress configuration
```

El campo `includeWhen` acepta una lista de expresiones booleanas. Todas las condiciones deben ser verdaderas para que el recurso se cree. Actualmente, `includeWhen` solo puede hacer referencia a campos `schema.spec`.

Transformaciones. Transforme valores mediante funciones y operadores ternarios:

```
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)}
```

Referencias cruzadas entre recursos. Valores de referencia de otros recursos:

```
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}
```

Al hacer referencia a otro recurso en una expresión de CEL, se crea automáticamente una dependencia. kro garantiza que el recurso al que se hace referencia se cree primero.

Para obtener más información, consulte CEL Expressions en la documentación de kro.

Dependencias de recursos

kro infiere automáticamente las dependencias de las expresiones de CEL: no se especifica el orden, se describen las relaciones. Cuando un recurso hace referencia a otro mediante una expresión de CEL, kro crea una dependencia y determina el orden de creación correcto.

```
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
```

La expresión `${bucket.spec.name}` crea una dependencia. kro crea un gráfico acíclico dirigido (DAG) de todos los recursos y sus dependencias y, a continuación, calcula un orden topológico para la creación.

Orden de creación: los recursos se crean por orden topológico (primero las dependencias).

Creación paralela: los recursos sin dependencias se crean simultáneamente.

Orden de eliminación: los recursos se eliminan por orden topológico inverso (primero los dependientes).

Dependencias circulares: no se permiten. kro rechaza las ResourceGraphDefinitions con dependencias circulares durante la validación.

Puede ver el orden de creación de computación:

```
kubectl get resourcegraphdefinition my-rgd -o jsonpath='{.status.topologicalOrder}'
```

Para obtener más información, consulte Graph inference en la documentación de kro.

Composición con ACK

kro funciona sin problemas con la capacidad de EKS para ACK para componer recursos de AWS con recursos de Kubernetes:

```
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}
```

Este patrón le permite crear recursos de AWS, extraer sus detalles (ARN, URL, puntos de conexión) e inyectarlos en la configuración de la aplicación, todo ello administrado como una sola unidad.

Para obtener más patrones de composición y ejemplos avanzados, consulte the section called “Consideraciones”.

Siguientes pasos

- the section called “Consideraciones”: más información sobre los patrones específicos de EKS, el RBAC y la integración con ACK y Argo CD
- Documentación de kro: documentación exhaustiva de kro que incluye solución de problemas, patrones de validación y expresiones de CEL avanzadas

Configuración de permisos de kro

A diferencia de ACK y Argo CD, kro no necesita permisos de IAM. kro opera completamente dentro de su clúster de Kubernetes y no hace llamadas a la API de AWS. Controle el acceso a los recursos de kro mediante el RBAC estándar de Kubernetes.

Cómo funcionan los permisos con kro

kro utiliza dos tipos de recursos de Kubernetes con diferentes ámbitos:

ResourceGraphDefinitions: recursos del ámbito del clúster que definen las API personalizadas. Por lo general, los administran los equipos de plataformas que diseñan y mantienen los estándares organizativos.

Instancias: recursos personalizados del ámbito del espacio de nombres creados a partir de ResourceGraphDefinitions. Los equipos de aplicaciones pueden crearlos con los permisos de RBAC adecuados.

De forma predeterminada, la capacidad de kro tiene permisos para administrar ResourceGraphDefinitions y sus instancias mediante la política de entrada de acceso `AmazonEKSKROPolicy`. Sin embargo, kro requiere permisos adicionales para crear y administrar los recursos de Kubernetes subyacentes definidos en las ResourceGraphDefinitions (como implementaciones, servicios o recursos de ACK). Debe otorgar estos permisos mediante las políticas de entrada de acceso o el RBAC de Kubernetes. Para obtener más información sobre la concesión de estos permisos, consulte los permisos de recursos arbitrarios de kro.

Permisos del equipo de plataformas

Los equipos de plataformas necesitan permisos para crear y administrar ResourceGraphDefinitions.

Ejemplo de ClusterRole para equipos de plataformas:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: kro-platform-admin
rules:
- apiGroups: ["kro.run"]
  resources: ["resourcegraphdefinitions"]
  verbs: ["*"]
```

Enlace a miembros del equipo de plataformas:

```
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
```

Permisos del equipo de aplicaciones

Los equipos de aplicaciones necesitan permisos para crear instancias de recursos personalizados en sus espacios de nombres.

Ejemplo de rol para equipos de aplicaciones:

```
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"]
```

Enlace a miembros del equipo de aplicaciones:

```
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**
>
> El grupo de API del rol (`kro.run` en este ejemplo) debe coincidir con la `apiVersion` definida en el esquema de ResourceGraphDefinition.

Acceso de solo lectura

Otorgue acceso de solo lectura para ver las ResourceGraphDefinitions e instancias sin permisos de modificación.

ClusterRole de solo lectura:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: kro-viewer
rules:
- apiGroups: ["kro.run"]
  resources: ["resourcegraphdefinitions"]
  verbs: ["get", "list", "watch"]
```

Rol de solo lectura para instancias:

```
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"]
```

Acceso a varios espacios de nombres

Otorgue a los equipos de aplicaciones acceso a varios espacios de nombres mediante ClusterRoles con RoleBindings.

ClusterRole para el acceso a varios espacios de nombres:

```
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"]
```

Enlace a espacios de nombres específicos:

```
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
```

Prácticas recomendadas

Principio de privilegio mínimo: otorgue solo los permisos mínimos necesarios para las responsabilidades de cada equipo.

Uso de grupos en lugar de usuarios individuales: enlace roles a grupos en lugar de usuarios individuales para facilitar la administración.

Preocupaciones de aplicaciones y plataformas independientes: los equipos de plataformas administran ResourceGraphDefinitions, mientras que los equipos de aplicaciones administran instancias.

Aislamiento del espacio de nombres: utilice los espacios de nombres para aislar diferentes equipos o entornos, con RBAC como control de acceso a cada espacio de nombres.

Acceso de solo lectura para auditoría: proporcione acceso de solo lectura a los equipos de seguridad y cumplimiento con fines de auditoría.

Siguientes pasos

- the section called “Conceptos de kro”: descripción de los conceptos de kro y la composición de recursos
- the section called “Conceptos de kro”: descripción de SimpleSchema, las expresiones de CEL y los patrones de composición
- the section called “Consideraciones para las capacidades de EKS”: revisión de las prácticas recomendadas de seguridad para capacidades

Consideraciones sobre kro para EKS

En este tema, se tratan aspectos importantes al utilizar la capacidad de EKS para kro, lo que incluye cuándo se utilizan la composición de recursos, los patrones de RBAC y la integración con otras capacidades de EKS.

Cuándo se utiliza kro

kro está diseñado para crear patrones de infraestructura reutilizables y API personalizadas que simplifiquen la administración de recursos complejos.

Utilice kro cuando en los siguientes casos:

- Para la creación de plataformas de autoservicio con API simplificadas para los equipos de aplicaciones
- Para la estandarización de los patrones de infraestructura en todos los equipos (base de datos + copia de seguridad + supervisión)
- Para la administración de dependencias de los recursos y para la transferencia de valores entre recursos
- Para la creación de abstracciones personalizadas que oculten la complejidad de la implementación
- Para la composición de varios recursos de ACK en componentes de nivel superior
- Para la activación de flujos de trabajo de GitOps para pilas de infraestructuras complejas

No utilice kro en los siguientes casos:

- Para la administración de recursos simples e independientes (utilice directamente los recursos de ACK o Kubernetes)

- Para cuando necesite una lógica dinámica en tiempo de ejecución (kro es declarativo, no imperativo)
- Para cuando los recursos no tengan dependencias ni configuración compartida

kro destaca por crear “rutas pavimentadas”: patrones obstinados y reutilizables que facilitan a los equipos la implementación correcta de infraestructuras complejas.

Patrones de RBAC

kro permite separar las preocupaciones entre los equipos de plataformas que crean ResourceGraphDefinitions y los equipos de aplicaciones que crean instancias.

Responsabilidades del equipo de plataformas

Los equipos de plataformas crean y mantienen ResourceGraphDefinitions (RGD) que definen las API personalizadas.

Permisos necesarios:

- Creación, actualización y eliminación de ResourceGraphDefinitions
- Administración de los tipos de recursos subyacentes (implementaciones, servicios, recursos de ACK)
- Acceso a todos los espacios de nombres en los que se utilizarán las RGD

Ejemplo de ClusterRole para el equipo de plataformas:

```
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"]
```

Para obtener una configuración de RBAC detallada, consulte the section called “Configuración de permisos”.

Responsabilidades del equipo de aplicaciones

Los equipos de aplicaciones crean instancias de recursos personalizados definidas por las RGD sin necesidad de comprender la complejidad subyacente.

Permisos necesarios:

- Creación, actualización y eliminación de instancias de recursos personalizados
- Acceso de lectura a su espacio de nombres
- Sin acceso a recursos subyacentes o las RGD

Ventajas:

- Uso de API sencillas y de alto nivel por parte de los equipos
- Control de los detalles de la implementación por parte de los equipos
- Menor riesgo de errores de configuración
- Incorporación más rápida de los nuevos miembros del equipo

Integración con otras capacidades de EKS

Composición de recursos de ACK

kro es particularmente eficaz cuando se combina con ACK para crear patrones de infraestructura.

Patrones comunes:

- Aplicación con almacenamiento: bucket de S3 + cola de SQS + configuración de notificaciones
- Pila de base de datos: instancia de RDS + grupo de parámetros + grupo de seguridad + secreto de Secrets Manager
- Redes: VPC + subredes + tablas de enrutamiento + grupos de seguridad + puertas de enlace de NAT
- Computación con almacenamiento: instancia de EC2 + volúmenes de EBS + perfil de instancia de IAM

kro gestiona el orden de las dependencias, transfiere valores entre los recursos (como los ARN y las cadenas de conexión) y administra todo el ciclo de vida como una sola unidad.

Para ver ejemplos de composición de recursos de ACK, consulte the section called “Conceptos de ACK”.

GitOps con Argo CD

Utilice la capacidad de EKS para Argo CD para implementar tanto las RGD como las instancias de los repositorios de Git.

Organización de repositorios:

- Repositorio de plataformas: contiene ResourceGraphDefinitions administradas por el equipo de plataformas
- Repositorios de aplicaciones: contienen instancias de recursos personalizados administradas por los equipos de aplicaciones
- Repositorio compartido: contiene tanto RGD como instancias para organizaciones más pequeñas

Consideraciones:

- Implemente las RGD antes que las instancias (las ondas de sincronización de Argo CD pueden ayudarlo).
- Utilice proyectos de Argo CD independientes para los equipos de plataformas y aplicaciones.
- El equipo de plataformas controla el acceso al repositorio de RGD.
- Los equipos de aplicaciones tienen acceso de solo lectura a las definiciones de RGD.

Para obtener más información sobre Argo CD, consulte the section called “Uso de Argo CD”.

Organización de ResourceGraphDefinitions

Organice las RGD por propósito, complejidad y propiedad.

Por propósito:

- Infraestructura: pilas de bases de datos, redes, almacenamiento
- Aplicación: aplicaciones web, API, trabajos por lotes
- Plataforma: servicios compartidos, supervisión, registro

Por complejidad:

- Sencilla: de 2 a 3 recursos con dependencias mínimas
- Moderada: de 5 a 10 recursos con algunas dependencias
- Compleja: más de 10 recursos con dependencias complejas

Convenciones de nomenclatura:

- Utilice nombres descriptivos: `webapp-with-database`, `s3-notification-queue`.
- Incluya la versión en el nombre para los cambios bruscos: `webapp-v2`.
- Utilice prefijos coherentes para las RGD relacionadas: `platform-` , `app-`

Estrategia de espacio de nombres:

- Las RGD tienen un ámbito de clúster (no tienen espacios de nombres).
- Las instancias tienen espacios de nombres.
- Utilice selectores de espacios de nombres en las RGD para controlar dónde se pueden crear las instancias.

Control de versiones y actualizaciones

Planifique la evolución de RGD y la migración de instancias.

Actualizaciones de RGD:

- Cambios no bruscos: actualice la RGD (agregue campos opcionales y nuevos recursos con includeWhen).
- Cambios bruscos: cree una nueva RGD con un nombre diferente (webapp-v2).
- Obsolescencia: marque las RGD antiguos con anotaciones y comunique la programación de migración.

Migración de instancias:

- Cree nuevas instancias con la RGD actualizada.
- Valide que las nuevas instancias funcionen correctamente.
- Elimine las instancias antiguas.
- kro gestiona automáticamente las actualizaciones de los recursos subyacentes.

Prácticas recomendadas:

- Pruebe antes los cambios de RGD en entornos que no sean de producción.
- Utilice el control de versiones semántico en los nombres de RGD para cambios importantes.
- Documente los cambios bruscos y las rutas de migración.
- Proporcione ejemplos de migración para los equipos de aplicaciones.

Validación y pruebas

Valide las RGD antes de implementarlas en producción.

Estrategias de validación:

- Validación de esquema: kro valida la estructura de RGD automáticamente.
- Instancias de prueba: cree instancias de prueba en los espacios de nombres de desarrollo.
- Pruebas de integración: compruebe que los recursos compuestos funcionen juntos
- Aplicación de políticas: utilice controladores de admisión para hacer cumplir los estándares organizativos

Problemas comunes que se deben probar:

- Dependencias y orden de los recursos
- Transferencia de valores entre recursos (expresiones de CEL)
- Inclusión condicional de recursos (includeWhen)
- Propagación del estado a partir de los recursos subyacentes
- Permisos de RBAC para la creación de instancias

Documentación ascendente

Para obtener información detallada sobre el uso de kro:

- Introducción a kro: creación de ResourceGraphDefinitions
- Expresiones de CEL: escritura de expresiones de CEL
- Guías de kro: patrones de composición avanzados

- Solución de problemas: solución de problemas y depuración

Siguientes pasos

- the section called “Configuración de permisos”: configuración del RBAC para los equipos de plataformas y aplicaciones
- the section called “Conceptos de kro”: descripción de los conceptos de kro y el ciclo de vida de los recursos
- the section called “Solución de problemas”: solución de problemas de kro
- the section called “Conceptos de ACK”: más información sobre los recursos de ACK para la composición
- the section called “Uso de Argo CD”: implementación de instancias y RGD con GitOps

Solución de problemas con capacidades de kro

En este tema, se proporcionan instrucciones para la solución de problemas de la capacidad de EKS para kro, como las comprobaciones de estado de la capacidad, los permisos de RBAC, los errores de expresión de CEL y los problemas de composición de los recursos.

> **Note**
>
> Las capacidades de EKS son completamente administradas y se ejecutan fuera del clúster. No tiene acceso a los registros del controlador ni al espacio de nombres `kro-system`. La solución de problemas se centra en el estado de la capacidad, la configuración de RBAC y el estado de los recursos.

La capacidad está ACTIVA, pero ResourceGraphDefinitions no funciona

Si la capacidad de kro muestra el estado `ACTIVE`, pero ResourceGraphDefinitions no crea recursos subyacentes, compruebe el estado de la capacidad, los permisos de RBAC y el estado de los recursos.

Compruebe el estado de la capacidad:

Puede ver los problemas de estado de la capacidad en la consola de EKS o mediante la AWS CLI.

Consola:

1. Abra la consola de Amazon EKS en https://console.aws.amazon.com/eks/home#/clusters.
2. Seleccione el nombre del clúster.
3. Seleccione la pestaña Observabilidad.
4. Elija Supervisar clúster.
5. Seleccione la pestaña Capacidades para ver el estado de todas las capacidades.

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
```

Causas habituales:

- Faltan los permisos de RBAC: kro no tiene permisos para crear los recursos subyacentes de Kubernetes
- Expresiones de CEL no válidas: errores de sintaxis en ResourceGraphDefinition
- Dependencias de recursos: los recursos dependientes no están listos
- Validación del esquema: la instancia no cumple con los requisitos del esquema de RGD

Compruebe los permisos de RBAC:

```
# Check if capability has cluster admin policy
kubectl get accessentry -A | grep kro
```

Si la capacidad no tiene los permisos necesarios, asocie la `AmazonEKSClusterAdminPolicy` a la entrada de acceso de la capacidad de kro o cree políticas de RBAC más restrictivas para su uso en producción. Para obtener más información, consulte the section called “Configuración de permisos”.

Compruebe el estado de 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}'
```

Las ResourceGraphDefinitions tienen tres condiciones de estado clave:

- `ResourceGraphAccepted`: si la RGD ha superado la validación (sintaxis de CEL, comprobación de tipos, existencia de campos)
- `KindReady`: si la CRD de la API personalizada se generó y registró
- `ControllerReady`: si kro está supervisando activamente las instancias de la API personalizada

Si `ResourceGraphAccepted` es `False`, compruebe el mensaje de la condición para ver si hay errores de validación, como campos desconocidos, discrepancias de tipos o dependencias circulares.

Se crearon las instancias, pero los recursos subyacentes no aparecen

Si existen instancias de recursos personalizadas, pero no se crean los recursos de Kubernetes subyacentes (implementaciones, servicios, ConfigMaps), compruebe que kro tenga permisos y si hay errores de composición.

Compruebe el estado de la instancia:

```
# 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}'
```

Las instancias tienen un campo `state` que muestra el estado de alto nivel:

- `ACTIVE`: la instancia se está ejecutando correctamente.
- `IN_PROGRESS`: la instancia se está procesando o conciliando.
- `FAILED`: no se pudo conciliar la instancia.
- `DELETING`: la instancia se está eliminando.
- `ERROR`: se ha producido un error durante el procesamiento.

Las instancias también tienen cuatro condiciones de estado:

- `InstanceManaged`: los finalizadores y las etiquetas están configurados correctamente.
- `GraphResolved`: se creó un gráfico de tiempo de ejecución y se resolvieron los recursos.
- `ResourcesReady`: todos los recursos están creados y listos.
- `Ready`: estado general de la instancia (solo pasa a `True` cuando todas las subcondiciones son `True`).

Céntrese en la condición `Ready` para determinar el estado de la instancia. Si `Ready` es `False`, compruebe las subcondiciones para identificar en qué fase se ha producido el error.

Compruebe los permisos de RBAC:

La capacidad de kro necesita permisos para crear los recursos de Kubernetes subyacentes definidos en las ResourceGraphDefinitions.

```
# Check if the capability has the AmazonEKSClusterAdminPolicy
kubectl get accessentry -A | grep kro
```

Si faltan los permisos, asocie la `AmazonEKSClusterAdminPolicy` a la entrada de acceso de la capacidad de kro o cree políticas de RBAC más restrictivas para su uso en producción. Para obtener más información, consulte the section called “Configuración de permisos”.

Errores de expresión de CEL

Los errores de expresión de CEL se detectan en el momento de la creación de ResourceGraphDefinition, no cuando se crean las instancias. kro valida toda la sintaxis de CEL,

compara las expresiones con los esquemas de Kubernetes y verifica la existencia de los campos al crear la RGD.

Errores comunes de validación de CEL:

- Referencia de campo no definida: se hace referencia a un campo que no existe en el esquema o el recurso.
- Discrepancia de tipos: la expresión devuelve un tipo incorrecto (por ejemplo, una cadena cuando se espera un número entero).
- Sintaxis no válida: faltan corchetes, comillas u operadores en la expresión de CEL.
- Tipo de recurso desconocido: se hace referencia a una CRD que no existe en el clúster.

Compruebe el estado de validación de la RGD:

```
# 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
```

Si `ResourceGraphAccepted` es `False`, el mensaje de condición contiene el error de validación.

Ejemplos de expresiones de CEL válidas:

```
# 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}
```

No se resuelven las dependencias de los recursos

kro infiere automáticamente las dependencias a partir de expresiones de CEL y crea los recursos en el orden correcto. Si los recursos no se crean según lo previsto, compruebe el orden de dependencia y la disponibilidad de los recursos.

Consulte el orden de creación de computación:

```
# See the order kro will create resources
kubectl get resourcegraphdefinition my-rgd -o jsonpath='{.status.topologicalOrder}'
```

Se muestra el orden de computación en función de las referencias de expresiones de CEL entre los recursos.

Compruebe la preparación de los recursos:

```
# 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}'
```

Compruebe las condiciones readyWhen (si se utilizan):

El campo `readyWhen` es opcional. Si no se especifica, los recursos se consideran listos inmediatamente después de su creación. Si ha definido las condiciones `readyWhen`, verifique que comprueben correctamente si los recursos están listos:

```
resources:
  - id: deployment
    readyWhen:
      - ${deployment.status.availableReplicas == deployment.spec.replicas}
```

Compruebe los eventos de los recursos:

```
# View events for the underlying resources
kubectl get events -n namespace --sort-by='.lastTimestamp'
```

Errores de validación del esquema

Si las instancias no se crean debido a errores de validación del esquema, compruebe que la instancia cumpla con los requisitos del esquema de RGD.

Compruebe los errores de validación:

```
# 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"
```

Problemas de validación comunes:

- Faltan campos obligatorios: la instancia no proporciona todos los campos del esquema obligatorios.
- Discrepancia de tipos: se proporciona una cadena cuando se espera un entero.
- Valor de enumeración no válido: se utiliza un valor que no está en la lista de permitidos.
- Discrepancia de patrones: la cadena no coincide con el patrón de expresión regular.

Revise el esquema de RGD:

```
# View the schema definition
kubectl get resourcegraphdefinition my-rgd -o jsonpath='{.spec.schema}'
```

Asegúrese de que la instancia proporcione todos los campos obligatorios con los tipos correctos.

Siguientes pasos

- the section called “Consideraciones”: consideraciones y prácticas recomendadas de kro
- the section called “Configuración de permisos”: configuración del RBAC para los equipos de plataformas y aplicaciones
- the section called “Conceptos de kro”: descripción de los conceptos de kro y el ciclo de vida de los recursos
- the section called “Solución de problemas de capacidades”: orientación general de solución de problemas de la capacidad

Comparación de la capacidad de EKS para kro con kro autoadministrado

La capacidad de EKS para kro proporciona la misma funcionalidad que kro autoadministrado, pero con importantes ventajas operativas. Para obtener una comparación general entre las capacidades de EKS y las soluciones autoadministradas, consulte the section called “Consideraciones”.

La capacidad de EKS para kro utiliza los mismos controladores de kro ascendentes y es totalmente compatible con kro ascendente. Las ResourceGraphDefinitions, las expresiones de CEL y la composición de recursos funcionan de la misma forma. Para ver la documentación completa de kro y ejemplos, consulte la documentación de kro.

Ruta de migración

Puede migrar de kro autoadministrado a la capacidad administrada sin tiempo de inactividad.

> **Important**
>
> Antes de migrar, asegúrese de que el controlador de kro autoadministrado ejecute la misma versión que la capacidad de EKS para kro. Compruebe la versión de la capacidad en la consola de EKS o mediante `aws eks describe-capability` y, a continuación, actualice la instalación autoadministrada para que coincida. De este modo, se evitan problemas de compatibilidad durante la migración.

1. Actualice el controlador de kro autoadministrado para utilizar `kube-system` en las asignaciones de elección de líderes:

   ```
   helm upgrade --install kro \
     oci://ghcr.io/awslabs/kro/kro-chart \
     --namespace kro \
     --set leaderElection.namespace=kube-system
   ```

   De este modo, se traslada la asignación del controlador a `kube-system`, lo que permite que la capacidad administrada se coordine con él.
2. Cree la capacidad de kro en el clúster (consulte the section called “Creación de una capacidad de kro”).
3. La capacidad administrada reconoce las ResourceGraphDefinitions e instancias y se encarga de la conciliación.

4. Reduzca verticalmente o elimine de forma gradual las implementaciones de kro autoadministrado:

```
helm uninstall kro --namespace kro
```

Este enfoque permite que ambos controladores coexistan de forma segura durante la migración. La capacidad administrada adopta automáticamente las ResourceGraphDefinitions e instancias que antes administraba kro autoadministrado, lo que garantiza una conciliación continua sin conflictos.

Siguientes pasos

- the section called “Creación de una capacidad de kro”: creación de un recurso con la capacidad de kro
- the section called “Conceptos de kro”: descripción de los conceptos de kro y la composición de recursos

Solución de problemas de capacidades de EKS

En este tema, se proporciona una guía general de solución de problemas para las capacidades de EKS, lo que incluye las comprobaciones de estado de las capacidades, los problemas comunes y los enlaces a la solución de problemas específicos de las capacidades.

> **Note**
>
> Las capacidades de EKS son completamente administradas y se ejecutan fuera del clúster. No tiene acceso a los registros ni a los espacios de nombres de los controladores. La solución de problemas se centra en el estado de la capacidad, el estado de los recursos y la configuración.

Enfoque general para la solución de problemas

Al solucionar problemas de las capacidades de EKS, siga este enfoque general:

1. Compruebe el estado de la capacidad: utilice `aws eks describe-capability` para ver el estado de la capacidad y los problemas de estado.
2. Compruebe el estado de los recursos: compruebe los eventos y las condiciones de estado de los recursos de Kubernetes (CRD) que ha creado.

3. Revise los permisos de IAM: asegúrese de que el rol de capacidad tenga los permisos necesarios.
4. Compruebe la configuración: compruebe que la configuración específica de la capacidad sea correcta.

Comprobación de estado de la capacidad

Todas las capacidades de EKS proporcionan información sobre el estado a través de la consola de EKS y la API `describe-capability`.

Consola:

1. Abra la consola de Amazon EKS en https://console.aws.amazon.com/eks/home#/clusters.
2. Seleccione el nombre del clúster.
3. Seleccione la pestaña Observabilidad.
4. Elija Supervisar clúster.
5. Seleccione la pestaña Capacidades para ver el estado de todas las capacidades.

La pestaña Capacidades muestra lo siguiente:

- Nombre y tipo de capacidad
- Estado actual
- Problemas de estado, con descripción

AWS CLI:

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-capability-name
```

La respuesta incluye:

- estado: estado actual de la capacidad (`CREATING`, `ACTIVE`, `UPDATING`, `DELETING`, `CREATE_FAILED`, `UPDATE_FAILED`)
- estado: información sobre el estado, incluidos los problemas detectados por la capacidad

Estados de capacidad comunes

CREANDO: Se está configurando la capacidad.

ACTIVA: la capacidad está en ejecución y lista para usarse. Si los recursos no funcionan según lo esperado, compruebe el estado de los recursos y los permisos de IAM.

ACTUALIZANDO: se están aplicando cambios en la configuración. Espere a que el estado vuelva a ser `ACTIVE`.

CREATE_FAILED o UPDATE_FAILED: la configuración o la actualización detectaron un error. Consulte la sección sobre el estado para obtener más información. Causas comunes:

- Falta la política de confianza del rol de IAM o no se encuentra
- El rol de IAM no existe o no se puede acceder a él
- Problemas de acceso al clúster
- Parámetros de configuración no válidos

Comprobación del estado de los recursos de Kubernetes

Las capacidades de EKS crean y administran definiciones de recursos personalizados (CRD) de Kubernetes en el clúster. Al solucionar problemas, compruebe el estado de los recursos que creó:

```
# 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
```

Las condiciones de estado de los recursos proporcionan información sobre:

- Si el recurso está listo
- Cualquier error encontrado
- Estado de conciliación actual

Revisión de los permisos de IAM y el acceso al clúster

Muchos problemas de las capacidades se deben a problemas con los permisos de IAM o a la falta de una configuración de acceso al clúster. Compruebe los permisos del rol de capacidad y las entradas de acceso al clúster.

Comprobación de los roles de IAM

Compruebe que el rol de capacidad tenga los permisos necesarios:

```
# 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'
```

La política de confianza debe permitir la entidad principal de servicio `capabilities.eks.amazonaws.com`:

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

```
}
```

Comprobación de las entradas y políticas de acceso de EKS

Todas las capacidades requieren entradas de acceso y políticas de acceso de EKS adecuadas en el clúster en el que operan.

Compruebe que la entrada de acceso exista:

```
aws eks list-access-entries \
  --cluster-name my-cluster \
  --region region-code
```

Busque el ARN del rol de capacidad en la lista. Si falta, la capacidad no puede acceder al clúster.

Compruebe las políticas de acceso adjuntas a la entrada:

```
aws eks list-associated-access-policies \
  --cluster-name my-cluster \
  --principal-arn arn:aws:iam::111122223333:role/my-capability-role \
  --region region-code
```

Todas las capacidades requieren políticas de acceso adecuadas:

- ACK: necesita permisos para crear y administrar los recursos de Kubernetes.
- kro: necesita permisos para crear y administrar los recursos de Kubernetes.
- Argo CD: necesita permisos para crear y administrar aplicaciones, así como entradas de acceso en clústeres de destino remotos para implementaciones de varios clústeres.

Para las implementaciones de varios clústeres de Argo CD:

Si se implementa en clústeres remotos, compruebe que el rol de capacidad tenga una entrada de acceso en cada clúster de destino:

```
# 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
```

Si falta la entrada de acceso en un clúster de destino, Argo CD no podrá implementar aplicaciones en él. Consulte the section called “Registro de clústeres” para obtener información detallada sobre la configuración.

Solución de problemas específicos de la capacidad

Para obtener una guía detallada de solución de problemas específica para cada tipo de capacidad:

- the section called “Solución de problemas”: solución de problemas relacionados con la creación de recursos de ACK, los permisos de IAM y el acceso entre cuentas
- the section called “Solución de problemas”: solución de problemas de sincronización de aplicaciones, autenticación de repositorios e implementaciones de varios clústeres
- the section called “Solución de problemas”: solución de problemas de ResourceGraphDefinitions, expresiones de CEL y permisos de RBAC

Problemas comunes en todas las capacidades

La capacidad se ha bloqueado en el estado CREANDO

Si una capacidad permanece en el estado `CREATING` durante más tiempo del esperado:

1. Compruebe el estado de la capacidad para detectar problemas específicos en la consola (Observabilidad > Supervisar clúster > pestaña Capacidades) o mediante la AWS CLI:

```
aws eks describe-capability \
  --region region-code \
  --cluster-name my-cluster \
  --capability-name my-capability-name \
  --query 'capability.health'
```

2. Compruebe si el rol de IAM existe y tiene la política de confianza correcta.
3. Asegúrese de que el clúster sea accesible y esté en buen estado.
4. Compruebe si hay algún problema en el clúster que pueda impedir la configuración de la capacidad.

Los recursos no se crean o actualizan

Si la capacidad es `ACTIVE`, pero los recursos no se crean o actualizan:

1. Compruebe el estado del recurso para ver si hay condiciones de error.
2. Compruebe los permisos de IAM para los servicios de AWS (ACK) o repositorios (Argo CD) específicos.
3. Compruebe los permisos de RBAC para crear recursos subyacentes (kro).
4. Revise las especificaciones de los recursos para ver si hay errores de validación.

El estado de la capacidad muestra problemas

Si `describe-capability` muestra problemas de estado:

1. Lea atentamente las descripciones de los problemas, ya que suelen indicar el problema específico.
2. Aborde la causa raíz (permisos de IAM, errores de configuración, etc.).
3. La capacidad se recuperará automáticamente una vez que se resuelva el problema.

Siguientes pasos

- the section called “Uso de capacidades”: administración de los recursos de la capacidad
- the section called “Solución de problemas”: solución de problemas específicos de ACK
- the section called “Solución de problemas”: solución de problemas específicos de Argo CD
- the section called “Solución de problemas”: solución de problemas específicos de kro
- the section called “Consideraciones para las capacidades de EKS”: prácticas recomendadas de seguridad para capacidades

Organización y supervisión de los recursos del clúster

En este capítulo, se explican los siguientes temas para ayudarlo a administrar el clúster. También puede ver la información de sus Recursos de Kubernetes con la Consola de administración de AWS.

- El panel de Kubernetes es una interfaz de usuario de uso general basada en la web para clústeres de Kubernetes. Permite a los usuarios administrar las aplicaciones que se ejecutan en el clúster y solucionar sus problemas, así como administrar el propio clúster. Para obtener más información, consulte el repositorio GitHub del Panel de Kubernetes.
- the section called “Servidor de métricas” El servidor de métricas de Kubernetes es un agregador de datos de uso de recursos en el clúster. No se implementa de forma predeterminada en el clúster, pero lo utilizan los complementos de Kubernetes, como el panel de Kubernetes y the section called “Escalador automático de pods horizontales”. En este tema aprenderá a instalar el servidor de métricas.
- the section called “Implementación de aplicaciones con Helm” El administrador de paquetes Helm para Kubernetes lo ayuda a instalar y administrar aplicaciones en el clúster de Kubernetes. Este tema lo ayudará a instalar y ejecutar los archivos binarios de Helm para que pueda instalar y administrar gráficos mediante la CLI de Helm en su equipo local.
- the section called “Etiquetado de recursos” Para ayudarlo a administrar los recursos de Amazon EKS, puede asignar sus propios metadatos a cada recurso en forma de etiquetas. En este tema se describe qué son las etiquetas y cómo crearlas.
- the section called “Service Quotas” La cuenta de AWS tiene cuotas predeterminadas, antes denominadas límites, para cada servicio de AWS. Obtenga más información sobre las cuotas para Amazon EKS y sobre cómo aumentarlas.

Supervisión y optimización de los costos de los clústeres de Amazon EKS

La supervisión de los costos es un aspecto esencial de la administración de los clústeres de Kubernetes en Amazon EKS. Al obtener visibilidad de los costos de su clúster, puede optimizar el uso de los recursos, establecer presupuestos y tomar decisiones basadas en datos sobre sus implementaciones. Amazon EKS ofrece dos soluciones de supervisión de costos, cada una con sus propias ventajas únicas, para ayudarle a rastrear y asignar sus costos de manera efectiva:

Datos de asignación de costos divididos de facturación de AWS para Amazon EKS: esta característica nativa se integra perfectamente con la consola de facturación de AWS, lo que le permite analizar y asignar los costos mediante la misma interfaz y los mismos flujos de trabajo familiares que utiliza para otros servicios de AWS. Con la asignación de costos divididos, puede obtener información sobre sus costos de Kubernetes directamente junto con sus otros gastos de AWS, lo que le permite optimizar los costos de manera integral en todo su entorno de AWS. También puede aprovechar las características de facturación de AWS existentes, como Categorías de costos y Detección de anomalías en los costos, para mejorar aún más sus capacidades de administración de costos. Para obtener más información, consulte Understanding split cost allocation data en la Guía del usuario de facturación de AWS.

Kubecost: Amazon EKS es compatible con Kubecost, una herramienta de supervisión de costos de Kubernetes. Kubecost ofrece un enfoque nativo de Kubernetes y rico en características para la supervisión de costos, que proporciona desgloses de costos detallados por recursos de Kubernetes, recomendaciones de optimización de costos y paneles e informes listos para usar. Kubecost también recupera datos de precios precisos al integrarlos con el informe de costos y uso de AWS, lo que garantiza que tenga una visión precisa de los costos de Amazon EKS. Aprenda a instalar Kubecost. Consulte la página de Kubecost de AWS Marketplace para obtener información sobre cómo obtener una suscripción gratuita a Kubecost.

Visualización de los costos por pod en la facturación de AWS con la asignación de costos divididos

Supervisión de costos mediante datos de asignación de costos de AWS divididos para Amazon EKS

Puede utilizar los datos de asignación de costos de AWS divididos para Amazon EKS a fin de obtener una visibilidad pormenorizada de los costos de sus clústeres de Amazon EKS. Esto le permite analizar, optimizar y reembolsar los costos y el uso de sus aplicaciones de Kubernetes. Los costos de las aplicaciones se asignan a las unidades de negocio y los equipos individuales en función de los recursos de memoria y CPU de Amazon EC2 que consume la aplicación de Kubernetes. Los datos de asignación de costos divididos para Amazon EKS ofrecen visibilidad del costo por pod y le permiten agregar los datos de costos por pod mediante el espacio de nombres, el clúster y otras primitivas de Kubernetes. A continuación se muestran ejemplos de primitivas de Kubernetes que puede usar para analizar los datos de asignación de costos de Amazon EKS.

- Nombre de clúster

- Implementación
- Espacio de nombres
- Nodo
- Nombre de carga de trabajo
- Tipo de carga de trabajo

También se admiten las etiquetas de asignación de costos definidas por el usuario. Para obtener más información sobre el uso de los datos de asignación de costos divididos, consulte Understanding split cost allocation data en la Guía del usuario de facturación de AWS.

Configuración de informes de costos y usos

Puede activar los datos de asignación de costos divididos para ECS en la consola de administración de costos, la Interfaz de la línea de comandos de AWS, o en los SDK de AWS.

Utilice lo siguiente para los datos de asignación de costos divididos:

1. Active los datos de asignación de costos divididos. Para obtener más información, consulte Habilitación de los datos de asignación de costos divididos en la Guía del usuario de informe de costos y uso de AWS.
2. Incluya los datos en un informe nuevo o existente.
3. Visualización del informe. Puede utilizar la consola de administración de costo y facturación o visualizar los archivos de los informes en Amazon Simple Storage Service.

Instalación de Kubecost

Amazon EKS admite Kubecost, que puede utilizar para supervisar sus costos desglosados por los recursos de Kubernetes que incluyen pods, nodos, espacios de nombres y etiquetas. En este apartado se describe la instalación de Kubecost y el acceso al panel de control de Kubecost.

Amazon EKS ofrece un paquete optimizado por AWS de Kubecost para obtener visibilidad de los costos del clúster. Puede utilizar sus acuerdos de soporte de AWS existentes para obtener soporte. Para obtener más información sobre las versiones disponibles de Kubecost, consulte the section called “Más información sobre Kubecost”.

> **Note**
>
> Kubecost v2 presenta varias características principales nuevas. Más información sobre Kubecost v2.

Para obtener más información acerca de Kubecost, consulte la documentación de Kubecost y las preguntas frecuentes.

Instalación del paquete de Kubecost optimizado para Amazon EKS

Puede usar uno de los siguientes procedimientos para instalar el paquete de Kubecost optimizado para Amazon EKS:

- Antes de empezar, se recomienda consultar Kubecost - Architecture Overview para comprender cómo funciona Kubecost en Amazon EKS.
- Si es la primera vez que utiliza Amazon EKS, le recomendamos que utilice el complemento de Amazon EKS para la instalación, ya que simplifica la instalación del paquete de Kubecost optimizado para Amazon EKS. Para obtener más información, consulte Deploying Kubecost on an Amazon EKS cluster using Amazon EKS add-on.
- Para personalizar la instalación, puede configurar su paquete de Kubecost optimizado para Amazon EKS con Helm. Para obtener más información, consulte Deploying Kubecost on an Amazon EKS cluster using Helm en la documentación de Kubecost.

Acceso al panel de Kubecost

Una vez que se haya completado la configuración del paquete de Kubecost optimizado para Amazon EKS, debería tener acceso al panel de Kubecost. Para obtener más información, consulte the section called “Acceso al panel de Kubecost”.

Acceso al panel de Kubecost

Requisitos previos

1. Asegúrese de que el estado de los pods relacionados con Kubecost sea “En ejecución”.

```
kubectl get pods --namespace kubecost
```

Acceso al panel de Kubecost

1. En el dispositivo, habilite el reenvío de puertos para exponer el panel de control de Kubecost.
   - Si kubecost se instala con Helm:

     ```
     kubectl port-forward deployment/kubecost-cost-analyzer 9090 --namespace kubecost
     ```

   - Si kubecost se instala con el complemento de Amazon EKS:

     ```
     kubectl port-forward deployment/cost-analyzer 9090 --namespace kubecost
     ```

     Como alternativa, puede utilizar el Controlador del equilibrador de carga de AWS para exponer Kubecost y utilizar Amazon Cognito para la autenticación, autorización y administración de usuarios. Para obtener más información, consulte Cómo utilizar el equilibrador de carga de aplicación y Amazon Cognito para autenticar a usuarios de las aplicaciones web de Kubernetes.
2. En el mismo dispositivo en el que hizo el paso anterior, abra un navegador web e ingrese la siguiente dirección.

   ```
   http://localhost:9090
   ```

   Verá la página de información general de Kubecost en su navegador. Es posible que Kubecost tarde entre 5 y 10 minutos (o más) en recopilar las métricas, en función del tamaño del clúster. Puede ver sus gastos de Amazon EKS, incluidos los costos acumulados del clúster, los costos de los activos asociados de Kubernetes y los gastos mensuales agregados.
3. Para hacer un seguimiento de los costos del clúster, etiquete sus recursos de Amazon EKS para la facturación. Para obtener más información, consulte the section called “Etiquetado de los recursos para facturación”.
   - Cost allocation (Asignación de costos): vea los costos mensuales de Amazon EKS y los costos acumulados para cada uno de sus espacios de nombres y otras dimensiones durante los últimos siete días. Esto es útil para entender qué partes de su aplicación están contribuyendo al gasto de Amazon EKS.
   - Assets (Activos): vea los costos de los activos de infraestructura de AWS que se asocian a sus recursos de Amazon EKS.

Más información sobre Kubecost

Amazon EKS ofrece un paquete optimizado por AWS de Kubecost para obtener visibilidad de los costos del clúster. Amazon EKS admite Kubecost, que puede utilizar para supervisar sus costos desglosados por los recursos de Kubernetes que incluyen pods, nodos, espacios de nombres y etiquetas.

En este tema se describen las versiones disponibles de Kubecost y las diferencias entre los niveles disponibles. EKS es compatible con la versión 1 y la versión 2 de Kubecost. Cada versión está disponible en diferentes niveles. Puede utilizar el paquete de Kubecost optimizado para Amazon EKS para sus clústeres de Amazon EKS sin costo adicional. Es posible que se le cobre por el uso de los servicios de AWS asociados, como Amazon Managed Service para Prometheus. Puede utilizar sus acuerdos de soporte de AWS existentes para obtener asistencia.

Como administrador de plataforma y líder financiero de Kubernetes, puede usar Kubecost para visualizar un desglose de los cargos de Amazon EKS, asignar costos y aplicar cargos a las unidades organizativas, como los equipos de aplicaciones. Puede proporcionar a sus equipos internos y unidades de negocio datos de costos transparentes y precisos basados en su factura de AWS real. Además, también puede obtener recomendaciones personalizadas para la optimización de costos en función de su entorno de infraestructura y los patrones de uso dentro de sus clústeres. Para obtener más información acerca de Kubecost, consulte la documentación de Kubecost.

¿Cuál es la diferencia entre el paquete personalizado de Kubecost y la versión gratuita de Kubecost (también conocido como OpenCost)?

AWS y Kubecost han colaborado para ofrecer una versión personalizada de Kubecost. Esta versión incluye un subconjunto de características comerciales sin cargo adicional. Consulte las siguientes tablas para ver las características que se incluyen en el paquete personalizado de Kubecost.

Kubecost v2

¿Cuál es la diferencia entre la versión 1 y la versión 2 de Kubecost?

Kubecost 2.0 es una actualización principal con respecto a las versiones anteriores e incluye nuevas e importantes características, incluido un nuevo backend de API. Tenga en cuenta que las API de asignación y activos son totalmente compatibles con versiones anteriores. Revise la documentación de Kubecost para garantizar una transición sin problemas. Para ver la lista completa de mejoras, consulte el anuncio de Kubecost v2.0 y las notas de lanzamiento completas.

> **Important**
>
> Revise la documentación de Kubecost antes de realizar la actualización. La actualización puede afectar a la disponibilidad de los informes.

Comparación de las características principales:

Característica	Nivel 2.0 gratuito de Kubecost	Paquete Kubecost 2.0 optimizado para Amazon EKS	Kubecost Enterprise 2.0
Visibilidad de costos del clúster	Clústeres ilimitados de hasta 250 núcleos	Multiclúster unificado sin límite de núcleos cuando se integra con Amazon Managed Service para Prometheus	Número unificado e ilimitado de clústeres en un número ilimitado de entornos (es decir, multinube)
Implementación	Alojado por el usuario	Alojado por el usuario	Alojado por el usuario, alojado por Kubecost (inquilino dedicado), SaaS
Bases de datos admitidas	Prometheus local	Amazon Managed Service para Prometheus o Prometheus local	Cualquier versión de Prometheus y bases de datos personali zadas
Compatibilidad con retención de bases de datos (métricas sin procesar)	15 días	Datos históricos ilimitados	Datos históricos ilimitados
Retención de API e interfaz de usuario (ETL) de Kubecost	15 días	15 días	Sin límite

Característica	Nivel 2.0 gratuito de Kubecost	Paquete Kubecost 2.0 optimizado para Amazon EKS	Kubecost Enterprise 2.0
Visibilidad en la nube híbrida	-	Clústeres de Amazon EKS y Amazon EKS Anywhere	Multinube y nube híbrida
Alertas e informes periódicos	Solo se admite en el clúster principal, limitado a 250 núcleos	Alertas de eficiencia, alertas de presupuesto, alertas de cambio de gastos y más, compatibles con todos los clústeres	Alertas de eficiencia, alertas de presupuesto, alertas de cambio de gastos y más, compatibles con todos los clústeres
Informes guardados	-	Informes con 15 días de métricas	Informes que utilizan métricas y datos históricos ilimitados
Integración de facturación en la nube	Solo se admite en el clúster principal, limitado a 250 núcleos	Soporte de precios personalizado para AWS (incluidos varios clústeres y múltiples cuentas)	Compatibilidad con precios personalizados para cualquier nube
Recomendaciones de guardado	Solo se admite en el clúster principal, limitado a 250 núcleos	Información sobre el clúster principal, pero no hay un límite de 250 núcleos	Información sobre múltiples clústeres
Gobernanza: auditorías	-	-	Audite los eventos de costos históricos
Compatibilidad con inicio de sesión único (SSO)	-	Compatible con Amazon Cognito	Okta, Auth0, PingID, KeyCloak y todo lo demás personalizado

Característica	Nivel 2.0 gratuito de Kubecost	Paquete Kubecost 2.0 optimizado para Amazon EKS	Kubecost Enterprise 2.0
Control de acceso basado en roles (RBAC) con SAML 2.0	-	-	Okta, Auth0, PingID, KeyCloak y todo lo demás personalizado
Formación e incorporación empresarial	-	-	Servicio completo de formación e incorporación de FinOps
Equipos	-	-	Sí

Características nuevas:

Las siguientes características tienen límites de métricas:

- Agregador de Kubecost
- Monitoreo de la red
- Acciones de Kubecost
- Colecciones
- Detección de anomalías
- Solicitud de corrección de tamaño de contenedor
- Previsiones de Kubecost
- Autocompletado para filtrar y agregar

Límites de métricas:

Métrica	Nivel 2.0 gratuito de Kubecost	Paquete Kubecost 2.0 optimizado para Amazon EKS	Kubecost Enterprise 2.0
Tamaño del clúster	Clústeres ilimitados de hasta 250 núcleos	Sin límite	Sin límite
Retención de métricas	15 días	15 días	Sin límite
Compatibilidad con multiclústeres	No disponible	Disponible	Disponible
Límites de núcleos	250 núcleos por clúster	Sin límites de núcleos	Sin límites de núcleos

Kubecost v1

Característica	Nivel gratuito de Kubecost	Paquete de Kubecost optimizado para Amazon EKS	Kubecost Enterprise
Implementación	Alojado por el usuario	Alojado por el usuario	Alojado por el usuario o por Kubecost (SaaS)
Número de clústeres compatibles	Sin límite	Sin límite	Sin límite
Bases de datos admitidas	Prometheus local	Amazon Managed Service para Prometheus o Prometheus local	Prometheus, Amazon Managed Service para Prometheus, Cortex o Thanos
Soporte de retención de bases de datos	15 días	Datos históricos ilimitados	Datos históricos ilimitados

Característica	Nivel gratuito de Kubecost	Paquete de Kubecost optimizado para Amazon EKS	Kubecost Enterprise
Retención de API de Kubecost (ETL)	15 días	15 días	Datos históricos ilimitados
Visibilidad de costos del clúster	Clústeres individuales	Varios clústeres unificados	Varios clústeres unificados
Visibilidad en la nube híbrida	-	Clústeres de Amazon EKS y Amazon EKS Anywhere	Compatibilidad con multinubes y nubes híbridas
Alertas e informes periódicos	-	Soporte para alertas de eficiencia, alertas de presupuesto, alertas de cambio de gastos y más	Soporte para alertas de eficiencia, alertas de presupuesto, alertas de cambio de gastos y más
Informes guardados	-	Informes con datos de 15 días	Informes que utilizan datos históricos ilimitados
Integración de facturación en la nube	Necesario para cada clúster individual	Soporte de precios personalizado para AWS (incluidos varios clústeres y múltiples cuentas)	Soporte de precios personalizado para AWS(incluidos varios clústeres y múltiples cuentas)
Recomendaciones de guardado	Información sobre un único clúster	Información sobre un único clúster	Información sobre múltiples clústeres
Gobernanza: auditorías	-	-	Audite los eventos de costos históricos
Compatibilidad con inicio de sesión único (SSO)	-	Compatible con Amazon Cognito	Okta, Auth0, PingID, KeyCloak

Característica	Nivel gratuito de Kubecost	Paquete de Kubecost optimizado para Amazon EKS	Kubecost Enterprise
Control de acceso basado en roles (RBAC) con **2.0** SAML	-	-	Okta, Auth0, PingID, KeyCloak
Formación e incorporación empresarial	-	-	Servicio completo de formación e incorporación de FinOps

Preguntas frecuentes

Consulte las siguientes preguntas y respuestas frecuentes sobre el uso de Kubecost con Amazon EKS.

¿Qué es la característica de retención de API (ETL) de Kubecost?

La característica ETL de Kubecost agrega y organiza las métricas para mostrar la visibilidad de los costos en varios niveles de granularidad (como `namespace-level`, `pod-level`, y `deployment-level`). Con el paquete de Kubecost optimizado para Amazon EKS, los clientes obtienen datos e información de las métricas de los últimos 15 días.

¿Qué es la característica de alertas e informes periódicos? ¿Qué alertas e informes incluye?

Las alertas de Kubecost permiten a los equipos recibir actualizaciones de gasto en tiempo real de Kubernetes, así como el gasto en la nube. Los informes periódicos permiten a los equipos recibir vistas personalizadas de gastos históricos en la nube y Kubernetes. Ambos se pueden configurar mediante la UI de Kubecost o valores de Helm. Son compatibles con correos electrónicos, Slack y Microsoft Teams.

¿Qué incluyen los informes guardados?

Los informes guardados de Kubecost son vistas predefinidas de las métricas de costos y eficiencia. Incluyen el costo por clúster, espacio de nombres, etiqueta y más.

¿Qué es la integración de facturación en la nube?

La integración con las API de facturación de AWS permite a Kubecost mostrar los costos fuera del clúster (como Amazon S3). Además, permite a Kubecost conciliar las predicciones integradas de Kubecost en el clúster con datos de facturación reales para tener en cuenta el uso puntual, Savings Plans y los descuentos empresariales.

¿Qué incluyen las recomendaciones de ahorro?

Kubecost proporciona información y automatización para ayudar a los usuarios a optimizar su infraestructura y gastos de Kubernetes.

¿Se cobra por esta funcionalidad?

No. Puede usar el paquete de Kubecost optimizado para Amazon EKS sin cargo adicional. Si quiere capacidades adicionales de Kubecost que no están incluidas en este paquete, puede comprar una licencia empresarial de Kubecost a través de AWS Marketplace o directamente desde Kubecost.

¿Hay soporte disponible para el paquete de Kubecost optimizado para Amazon EKS?

Sí, solo si utiliza el paquete de Kubecost optimizado para Amazon EKS.

¿Cómo puedo obtener soporte para el paquete de Kubecost optimizado para Amazon EKS?

Puede abrir un caso de soporte con el equipo de AWS Support en Contacte con AWS.

¿Necesito una licencia para usar las características de Kubecost proporcionadas por la integración de Amazon EKS?

No.

¿Puedo integrar Kubecost con el informe de costos y uso de AWS para obtener informes más precisos?

Sí. Puede configurar Kubecost para que ingiera datos del informe de costos y uso de AWS y así obtener una vista precisa de los costos, incluidos descuentos, precios mercado, precios de instancias reservadas y otros. Para más información, consulte Integración de facturación en la nube de AWS en la documentación de Kubecost.

¿Esta versión admite la administración de costos de los clústeres de Kubernetes autoadministrados en Amazon EC2?

No. El paquete de Kubecost optimizado para Amazon EKS solo es compatible con los clústeres de Amazon EKS.

¿Kubecost puede hacer un seguimiento de los costos de Amazon EKS en AWS Fargate?

Kubecost ofrece el mejor esfuerzo para mostrar la visibilidad de los costos de los clústeres de Amazon EKS en Fargate, pero con una precisión inferior a la de Amazon EKS en Amazon EC2. Esto se debe principalmente a la diferencia en la forma en que se le factura el uso. Con Amazon EKS en Fargate, se le facturan los recursos consumidos. Con Amazon EKS en los nodos de Amazon EC2, se le facturan los recursos aprovisionados. Kubecost calcula el costo de un nodo de Amazon EC2 en función de la especificación del nodo, lo cual incluye la CPU, la RAM y el almacenamiento efímero. Con Fargate, los costos se calculan en función de los recursos solicitados para los pods de Fargate.

¿Cómo puedo obtener actualizaciones y nuevas versiones de Kubecost?

Puede actualizar su versión de Kubecost mediante procedimientos de actualización estándar de Helm. Las versiones más recientes se encuentran en la Galería pública de Amazon ECR.

¿Es el **`kubectl-cost`** compatible con la CLI? ¿Cómo se instala?

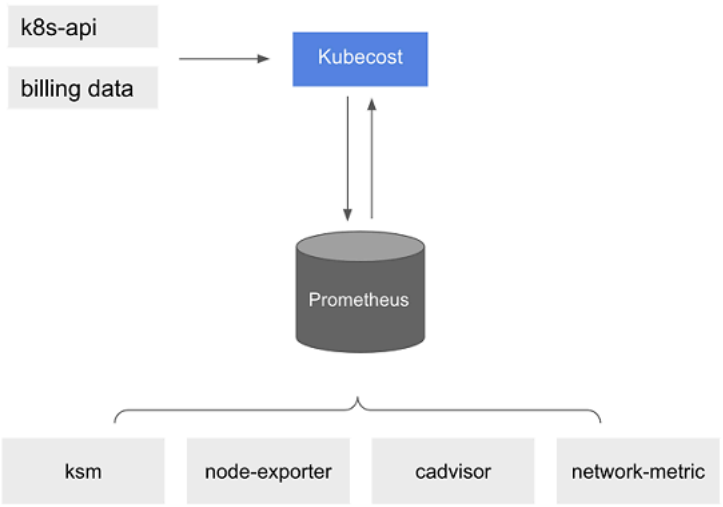
Sí. `Kubectl-cost` es una herramienta de código abierto de Kubecost (licencia Apache 2.0) que proporciona acceso de CLI a las métricas de asignación de costos de Kubernetes. Para instalar `kubectl-cost`, consulte Installation (Instalación) en GitHub.

¿Es compatible la interfaz de usuario de Kubecost? ¿Cómo puedo acceder a ella?

Kubecost proporciona un panel web al que puede acceder a través del reenvío de puertos `kubectl`, una entrada o un equilibrador de carga. También puede usar el controlador del equilibrador de carga de AWS para exponer Kubecost y usar Amazon Cognito para la autenticación, autorización y administración de usuarios. Para obtener más información, consulte Cómo usar el equilibrador de carga de aplicación y Amazon Cognito para autenticar a usuarios de las aplicaciones web de Kubernetes en el blog de AWS.

Características adicionales de Kubecost

- Las siguientes características están disponibles en ambas versiones (versión 1 y 2) de Kubecost.
  - Exportación de métricas de costos: la supervisión de costos optimizada de Amazon EKS se implementa con Kubecost y Prometheus, que es un sistema de supervisión de código abierto y una base de datos de serie temporal. Kubecost lee las métricas de Prometheus y, a continuación, hace cálculos de asignación de costos y vuelve a escribir las métricas en Prometheus. El frontend de Kubecost lee las métricas de Prometheus y las muestra en la interfaz de usuario de Kubecost. La arquitectura se ilustra en el siguiente diagrama.



Con Prometheus preinstalado, puede escribir consultas para ingerir datos de Kubecost en su actual sistema de inteligencia empresarial para su análisis posterior. También puede utilizarlo como origen de datos para su panel actual de Grafana para mostrar los costos del clúster de Amazon EKS con los que sus equipos internos están familiarizados. Para obtener más información sobre cómo escribir consultas de Prometheus, consulte el archivo https://opencost.io/docs/installation/prometheus/ `readme` en GitHub o utilice los modelos JSON de Grafana de ejemplo en el repositorio de Kubecost de GitHub como referencias.

- Integración del informe de costos y uso de AWS: para calcular la asignación de costos de un clúster de Amazon EKS, Kubecost recupera la información pública de precios de los servicios de AWS y recursos de AWS desde la API de lista de precios de AWS. También puede integrar Kubecost con el Informe de costos y uso de AWS: para mejorar la precisión de la información de precios específica de su cuenta de AWS. Esta información incluye programas de descuento para empresas, uso de instancias reservadas, Savings Plans y uso puntual. Para obtener más información sobre el funcionamiento de la integración del informe de costos y uso de AWS, consulte AWS Cloud Billing Integration en la documentación de Kubecost.

Visualización del uso de los recursos con el servidor de métricas de Kubernetes

El servidor de métricas de Kubernetes es un agregador de datos de uso de recursos en el clúster. No está implementado en los clústeres de Amazon EKS de forma predeterminada. Para obtener más información, consulte Servidor de métricas de Kubernetes en GitHub. Otros complementos

de Kubernetes suelen usar el servidor de métricas, como el escalado de las implementaciones de pods con el Escalador automático horizontal de pods o el panel de Kubernetes. Para obtener más información, consulte Resource metrics pipeline en la documentación de Kubernetes. En este tema, se explica cómo implementar el servidor de métricas de Kubernetes en el clúster de Amazon EKS.

> **Important**
>
> Las métricas están pensadas para el análisis en un momento dado y no son una fuente precisa para el análisis histórico. No se pueden utilizar como solución de monitorización ni para otros fines que no sean de escalado automático. Para obtener más información sobre las herramientas de monitorización, consulte Supervisión de clústeres.

Consideraciones

- Si implementa el Servidor de métricas de Kubernetes manualmente en los nodos de Fargate mediante el manifiesto, configure la implementación `metrics-server` para que utilice un puerto distinto al predeterminado, `10250`. Este puerto está reservado para Fargate. La versión del complemento de Amazon EKS del Servidor de métricas está preconfigurada para usar el puerto `10251`.
- Asegúrese de que los grupos de seguridad y las ACL de red permitan el puerto `10250` entre los pods `metrics-server` y todos los demás nodos y pods. El Servidor de métricas de Kubernetes sigue utilizando el puerto `10250` para recopilar métricas de otros puntos de conexión del clúster. Si implementa en nodos de Fargate, permita tanto el puerto alternativo configurado del Servidor de métricas como el puerto `10250`.

Implementación como complemento de la comunidad con complementos de Amazon EKS

Nuevo: ahora puede implementar el Servidor de métricas como un complemento de la comunidad mediante la consola de AWS o las API de Amazon EKS.

Implementación con la consola de AWS

1. Abra el clúster de EKS en la consola de AWS
2. En la pestaña “Complementos”, seleccione Obtener más complementos.

3. En la sección “Complementos de la comunidad”, seleccione Servidor de métricas y luego Siguiente
4. EKS determina la versión del complemento adecuada para el clúster. Para cambiar la versión, utilice el menú desplegable Versión.
5. Seleccione Siguiente y luego Crear para instalar el complemento.

Recursos adicionales

Obtención de más información sobre the section called “Complementos de la comunidad”.

Los complementos de la comunidad se instalan o actualizan del mismo modo que otros complementos de Amazon EKS.

- the section called “Cómo crear un complemento”
- the section called “Cómo actualizar un complemento”
- the section called “Cómo eliminar un complemento”

Implementación con manifiesto

Nuevo: ahora puede implementar el Servidor de métricas como un complemento de la comunidad mediante la consola de AWS o las API de Amazon EKS. Estas instrucciones de instalación del manifiesto se archivarán.

1. Implemente el servidor de métricas con el siguiente comando:

   ```
   kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml
   ```

   Si utiliza Fargate, deberá cambiar este archivo. En la configuración predeterminada, el servidor de métricas usa el puerto 10250. Este puerto está reservado en Fargate. Sustituya las referencias al puerto 10250 en components.yaml por otro puerto, como el 10251.
2. Compruebe que la implementación de `metrics-server` esté ejecutando la cantidad deseada de pods con el siguiente comando.

   ```
   kubectl get deployment metrics-server -n kube-system
   ```

   Un ejemplo de salida sería el siguiente.

```
NAME             READY   UP-TO-DATE   AVAILABLE   AGE
metrics-server   1/1     1            1           6m
```

3. Para probar si el servidor de métricas funciona, muestre el uso de recursos (CPU/memoria) de los nodos.

```
kubectl top nodes
```

4. Si aparece el mensaje de error `Error from server (Forbidden)`, tiene que actualizar la configuración de la RBAC de Kubernetes. Su identidad RBAC de Kubernetes necesita permisos suficientes para leer las métricas del clúster. Revise los permisos mínimos de la API de Kubernetes necesarios para leer las métricas en GitHub. Obtenga información sobre cómo conceder a las identidades de AWS IAM, como los roles, acceso a las API de Kubernetes.

Implementación de aplicaciones en Amazon EKS con Helm

El administrador de paquetes Helm para Kubernetes lo ayuda a instalar y administrar aplicaciones en el clúster de Kubernetes. Para obtener más información, consulte la documentación de Helm. Este tema lo ayudará a instalar y ejecutar los archivos binarios de Helm para que pueda instalar y administrar gráficos mediante la CLI de Helm en su sistema local.

> **Important**
>
> Antes de poder instalar gráficos de Helm en el clúster de Amazon EKS, debe configurar `kubectl` para que funcione con Amazon EKS. Si todavía no ha hecho esto, consulte the section called “Acceso al clúster con kubectl” antes de continuar. Si el siguiente comando se realiza correctamente para su clúster, entonces tiene la configuración correcta.
>
> ```
> kubectl get svc
> ```

1. Ejecute el comando apropiado para el sistema operativo del cliente.
   - Si utiliza macOS con Homebrew, instale los archivos binarios con el siguiente comando.

```
brew install helm
```

- Para obtener más opciones de instalación, consulte Instalación de Helm en los documentos de Helm.

> **Note**
>
> Si recibe un mensaje indicando que debe instalar `openssl` antes, puede instalarlo mediante el siguiente comando.

```
sudo yum install openssl
```

1. Para recoger los nuevos archivos binarios en su PATH, cierre la ventana del terminal actual y abra una nueva.
2. Consulte la versión de Helm que instaló.

   ```
   helm version --template='{{ .Version }}{{ "\n" }}'
   ```

   Un ejemplo de salida sería el siguiente.

   ```
   v3.17.2
   ```

3. Asegúrese de que la versión instalada sea compatible con la versión del clúster. Consulte la sección Sesgo de versiones compatibles para obtener más información. Por ejemplo, si ejecuta `3.17.x`, la versión de Kubernetes compatible no debería estar fuera del rango de `1.29.x ~ 1.32.x`.
4. En este momento, puede ejecutar cualquier comando de Helm (como `helm install` *`chart-name`*) para instalar, modificar, eliminar o consultar gráficos de Helm en el clúster. Si es nuevo en Helm y no tiene un gráfico específico que instalar, puede:
   - Experimentar mediante la instalación un gráfico de muestra. Consulte instalación de un gráfico de muestra en la guía de inicio rápido de Helm.
   - Cree un gráfico de ejemplo y envíelo a Amazon ECR. Para obtener más información, consulte Envío de un gráfico de Helm en la Guía del usuario de Amazon Elastic Container Registry.
   - Instale un gráfico de Amazon EKS desde el repositorio de GitHub eks-charts o desde ArtifactHub.

Organización de los recursos de Amazon EKS con etiquetas

Puede usar etiquetas para ayudarlo a administrar sus recursos de Amazon EKS. En este tema se proporciona información general sobre la función de etiquetas y se muestra cómo puede crear etiquetas.

Temas

- Conceptos básicos de etiquetas
- Etiquetado de recursos
- Restricciones de las etiquetas
- Etiquetado de los recursos para facturación
- Uso de etiquetas mediante la consola
- Uso de etiquetas mediante la CLI, la API o eksctl

> **Note**
>
> Las etiquetas son un tipo de metadatos independiente de las etiquetas y anotaciones de Kubernetes. Para obtener más información sobre estos otros tipos de metadatos, consulte las secciones siguientes de la documentación de Kubernetes:
>
> - Etiquetas y selectores
> - Annotations

Conceptos básicos de etiquetas

Una etiqueta es una marca que se asigna a un recurso de AWS. Cada etiqueta consta de una clave y un valor opcional.

Con las etiquetas, puede categorizar sus recursos de AWS. Por ejemplo, puede clasificar los recursos en categorías por objetivo, propietario o entorno. Cuando tiene muchos recursos del mismo tipo, puede utilizar las etiquetas que asignó a un recurso específico para identificarlo rápidamente. Por ejemplo, puede definir un conjunto de etiquetas para los clústeres de Amazon EKS a fin de ayudar a realizar un seguimiento del propietario y del nivel de pila de cada clúster. Le recomendamos que diseñe un conjunto coherente de claves de etiqueta para cada tipo de recurso. Puede buscar y filtrar los recursos en función de las etiquetas que agregue.

Después de agregar una etiqueta, puede editar las claves y los valores de las etiquetas o eliminar etiquetas de un recurso en cualquier momento. Si elimina un recurso, también se eliminará cualquier etiqueta asignada a dicho recurso.

Las etiquetas no tienen ningún significado semántico para Amazon EKS, por lo que se interpretan estrictamente como cadenas de caracteres. Puede establecer el valor de una etiqueta como una cadena vacía. Sin embargo, no se puede establecer el valor de una etiqueta como nulo. Si agrega una etiqueta con la misma clave que una etiqueta existente en ese recurso, el nuevo valor sobrescribirá al anterior.

Si utiliza AWS Identity and Access Management (IAM), puede controlar qué usuarios de su cuenta de AWS tienen permiso para administrar etiquetas.

Etiquetado de recursos

Las siguientes etiquetas de soporte de recursos de Amazon EKS:

- clústeres
- grupos de nodos administrados
- Perfiles de Fargate

Puede etiquetar estos recursos con lo siguiente:

- Si utiliza la consola de Amazon EKS, puede aplicar etiquetas a recursos nuevos o existentes en cualquier momento. Para ello, puede utilizar la pestaña Tags (Etiquetas) en la página de recursos pertinente. Para obtener más información, consulte the section called “Uso de etiquetas mediante la consola”.
- Si utiliza `eksctl`, puede aplicar etiquetas a los recursos cuando se crean mediante la opción `--tags`.
- Si utiliza la CLI de AWS, la API de Amazon EKS o un SDK de AWS, puede aplicar etiquetas a los recursos nuevos mediante el parámetro `tags` en la acción de la API pertinente. Puede aplicar etiquetas a recursos existentes a través de la acción de la API `TagResource`. Para obtener más información, consulte TagResource.

Cuando se utilizan algunas acciones de creación de recursos, se pueden especificar también etiquetas para el recurso al mismo tiempo que se crea. Si las etiquetas no pueden aplicarse mientras se crea el recurso, este no podrá crearse. Este mecanismo garantiza que los recursos que se

pretenden etiquetar se creen con las etiquetas que se especifican o no se creen en absoluto. Si se etiquetan los recursos al crearlos, no es necesario ejecutar scripts de etiquetado personalizados después de crear el recurso.

Las etiquetas no se propagan a otros recursos asociados al recurso que se crea. Por ejemplo, las etiquetas de perfil de Fargate no se propagan a otros recursos asociados al perfil de Fargate, como los pods que están programados con él.

Restricciones de las etiquetas

Se aplican las siguientes restricciones a las etiquetas:

- Se puede asociar un máximo de 50 etiquetas a un recurso.
- Las claves de etiquetas no se pueden repetir para un recurso. Cada clave de etiqueta debe ser única y solo puede tener un valor.
- Las claves pueden tener hasta 128 caracteres en UTF-8.
- Los valores pueden tener hasta 256 caracteres en UTF-8.
- Si hay múltiples servicios y recursos de AWS que utilizan su esquema de etiquetado, limite los tipos de caracteres que utilice. Algunos servicios pueden tener restricciones en cuanto a los caracteres permitidos. En general, los caracteres permitidos son letras, números, espacios y los siguientes caracteres: `+ - = . _ : / @`.
- Las claves y los valores de las etiquetas distinguen entre mayúsculas y minúsculas.
- No utilice `aws:`, `AWS:`, ni ninguna combinación de mayúsculas o minúsculas del mismo como prefijo para claves o valores. Estos están reservados solo para la utilización de AWS. Las claves y los valores de etiquetas que tienen este prefijo no se pueden editar. Las etiquetas con este prefijo no cuentan para el límite de etiquetas por recurso.

Etiquetado de los recursos para facturación

Cuando aplica etiquetas a los clústeres de Amazon EKS, puede utilizarlas para la asignación de costos en sus Informes de costo y uso. Los datos de medición de sus Informes de costo y uso muestran el uso en todos sus clústeres de Amazon EKS. Para obtener más información, consulte Informe de costos y usos de AWS en la Guía del usuario de AWS.

La etiqueta de asignación de costos generada por AWS, específicamente `aws:eks:cluster-name`, le permite desglosar los costos de las instancias de Amazon EC2 por clúster individual de Amazon EKS en el Explorador de costos. Sin embargo, esta etiqueta no captura los gastos del

plano de control. La etiqueta se agrega automáticamente a las instancias de Amazon EC2 que participan en un clúster de Amazon EKS. Este comportamiento se produce independientemente de si las instancias se aprovisionan mediante grupos de nodos administrados de Amazon EKS, Karpenter o directamente con Amazon EC2. Esta etiqueta específica no cuenta para el límite de 50 etiquetas. Para utilizar la etiqueta, el propietario de la cuenta debe activarla en la consola de facturación de AWS o mediante la API. Cuando el propietario de una cuenta de administración de AWS Organizations activa la etiqueta, esta también se activa para todas las cuentas miembro de la organización.

También puede organizar su información de facturación en función de los recursos que tienen los mismos valores de clave de etiqueta. Por ejemplo, puede etiquetar varios recursos con un nombre de aplicación específico y, luego, organizar su información de facturación. De esta manera, puede ver el costo total de la aplicación en distintos servicios. Para obtener más información acerca de la configuración de un informe de asignación de costos con etiquetas, consulte Informe de asignación de costos mensual en la Guía del usuario de facturación de AWS.

> **Note**
>
> Si acaba de activar los informes, los datos del mes actual estarán disponibles para su visualización después de 24 horas.

El Explorador de costos es una herramienta de informes que está disponible como parte del nivel gratuito de AWS. Puede utilizar el Explorador de costos para ver los gráficos de sus recursos de Amazon EKS de los últimos 13 meses. También puede prever cuánto va a gastar en los próximos tres meses. Puede ver los patrones de lo que gasta en recursos de AWS a lo largo del tiempo. Por ejemplo, se puede utilizar para identificar aspectos que deben estudiarse más a fondo y observar tendencias que pueden ayudar a comprender los costos. También puede especificar intervalos de tiempo para los datos y ver los datos temporales por día o por mes.

Uso de etiquetas mediante la consola

Con la consola de Amazon EKS puede administrar las etiquetas asociadas a los clústeres nuevos o existentes y a grupos de nodos administrados.

Al seleccionar una página específica de recursos en la consola de Amazon EKS, se muestra una lista de esos recursos. Por ejemplo, si selecciona Clusters (Clústeres) en el panel de navegación izquierda, la consola muestra una lista de los clústeres de Amazon EKS. Al seleccionar un recurso de

una de estas listas (por ejemplo, un clúster concreto) que admite etiquetas, puede ver y administrar sus etiquetas en la pestaña Tags (Etiquetas).

También puede utilizar Tag Editor (Editor de etiquetas) en laConsola de administración de AWS, que proporciona una forma unificada de administrar las etiquetas. Para obtener más información, consulte Etiquetar recursos de AWS con el editor de etiquetas en la Guía del usuario del editor de etiquetas de AWS.

Adición de etiquetas a un recurso al crearlo

Puede agregar etiquetas a clústeres de Amazon EKS y grupos de nodos administrados y perfiles de Fargate al crearlos. Para obtener más información, consulte the section called “Creación de un clúster ”.

Adición y eliminación de etiquetas en un recurso

Puede agregar o eliminar las etiquetas asociadas a sus clústeres directamente desde la página del recurso.

1. Abra la consola de Amazon EKS.
2. En la barra de navegación, seleccione la región de AWS que utilizará.
3. En el panel de navegación izquierdo, elija Clusters (Clústeres).
4. Elija un clúster específico.
5. Elija la pestaña Etiquetas y, a continuación, elija Administrar etiquetas.
6. En la página Manage tags (Administrar etiquetas), agregue o elimine las etiquetas según sea necesario.
   - Para agregar una etiqueta, elija Add tag (Añadir etiqueta). Especifique la clave y el valor para cada etiqueta.
   - Para eliminar una etiqueta, seleccione Remove tag (Eliminar etiqueta).
7. Repita este proceso para cada etiqueta que desee agregar o eliminar.
8. Elija Update (Actualizar) para finalizar.

Uso de etiquetas mediante la CLI, la API o **`eksctl`**

Utilice los siguientes comandos de la CLI de AWS o las operaciones de la API de Amazon EKS para agregar, actualizar, enumerar y eliminar las etiquetas de sus recursos. Solo puede utilizar `eksctl` para agregar etiquetas mientras se crean simultáneamente los nuevos recursos con un comando.

Tarea	AWS CLI	AWS Tools for Windows PowerShell	Acción de la API
Agregar o sobrescribir una o varias etiquetas .	tag-resource	Add-EKSResourceTag	TagResource
Eliminar una o varias etiquetas.	untag-resource	Remove-EKSResourceTag	UntagResource

Los siguientes ejemplos muestran cómo agregar o quitar etiquetas a los recursos mediante la CLI de AWS.

Ejemplo 1: Etiquetar un clúster existente

El siguiente comando etiqueta un clúster existente.

```
aws eks tag-resource --resource-arn resource_ARN --tags team=devs
```

Ejemplo 2: Quitar la etiqueta de un clúster existente

El siguiente comando elimina una etiqueta de un clúster existente.

```
aws eks untag-resource --resource-arn resource_ARN --tag-keys tag_key
```

Ejemplo 3: enumerar etiquetas de un recurso

El siguiente comando enumera las etiquetas que están asociadas a un recurso existente.

```
aws eks list-tags-for-resource --resource-arn resource_ARN
```

Cuando se utilizan algunas acciones de creación de recursos, se pueden especificar etiquetas al mismo tiempo que se crea el recurso. Las siguientes acciones permiten especificar una etiqueta al crear un recurso.

Tarea	AWS CLI	AWS Tools for Windows PowerShell	Acción de la API	eksctl
Creación de un clúster	create-cluster	New-EKSCluster	CreateCluster	`create cluster`
Creación de un grupo de nodos administrados*	create-nodegroup	New-EKSNodegroup	CreateNodegroup	`create nodegroup`
Creación de un perfil de Fargate	create-fargate-profile	New-EKSFargateProfile	CreateFargateProfile.html	`create fargateprofile`

- Si desea etiquetar también las instancias de Amazon EC2 cuando cree un grupo de nodos administrados, utilice una plantilla de lanzamiento para crear el grupo de nodos administrados. Para obtener más información, consulte the section called “Etiquetado de instancias de Amazon EC2”. Si las instancias ya existen, puede etiquetarlas de forma manual. Para obtener más información, consulte Etiquetado de los recursos en la Guía del usuario de Amazon EC2.

Visualización y administración de Amazon EKS y las Service Quotas de Fargate

Amazon EKS se ha integrado con Service Quotas, un servicio de AWS que le permite ver y administrar sus cuotas desde una ubicación central. Para obtener más información, consulte ¿Qué son las cuotas de servicio? en la Guía del usuario de Service Quotas. Con la integración de Service Quotas, puede buscar rápidamente el valor de sus cuotas de servicio de Amazon EKS y AWS Fargate con la Consola de administración de AWS y AWS CLI.

Consulta de las Service Quotas de EKS en la Consola de administración de AWS

1. Abra la consola de Service Quotas.
2. En el panel de navegación de la izquierda, elija Servicios de AWS.

3. En la lista servicios de AWS, busque y seleccione Amazon Elastic Kubernetes Service (Amazon EKS) o AWS Fargate.

   En la lista Service quotas, puede ver el nombre de la cuota de servicio, el valor aplicado (si está disponible), la cuota predeterminada de AWS y si el valor de cuota es ajustable.
4. Para ver información adicional sobre una cuota de servicio, como, por ejemplo, la descripción, elija el nombre de cuota.
5. (Opcional) Para solicitar un aumento de cuota, seleccione la cuota que desea aumentar, seleccione Solicitar aumento de cuota, escriba o seleccione la información necesaria y seleccione Solicitar.

Para trabajar más con cuotas de servicio mediante la Consola de administración de AWS, consulte la Guía del usuario de Service Quotas. Para solicitar un aumento de cuota, consulte Solicitud de aumento de cuota en la Guía del usuario de Service Quotas.

Consulta de las cuotas de servicio de EKS con AWS CLI

Ejecute el siguiente comando para ver las cuotas de Amazon EKS.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code eks \
    --output table
```

Ejecute el siguiente comando para ver las cuotas de Fargate.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code fargate \
    --output table
```

> **Note**
>
> La cuota devuelta es el número de tareas de Amazon ECS o pods de Amazon EKS que se ejecutan simultáneamente en Fargate en esta cuenta en la región de AWS actual.

Para trabajar más con las Service Quotas mediante AWS CLI, consulte la service-quotas en la Referencia de comandos de AWS CLI. Para solicitar un aumento de cuota, consulte el comando request-service-quota-increase en la Referencia de comandos de la AWS CLI.

Cuotas de servicio de Amazon EKS

AWS recomienda utilizar la Consola de administración de AWS para ver las cuotas actuales. Para obtener más información, consulte the section called “Consulta de las Service Quotas de EKS en la Consola de administración de AWS”.

Para consultar las cuotas de servicio predeterminadas de EKS, consulte Amazon Elastic Kubernetes Service endpoints and quotas en la Referencia general de AWS.

Estas service quotas se enumeran en Amazon Elastic Kubernetes Service (Amazon EKS), en la consola de Service Quotas. Para solicitar un aumento de cuota para los valores que se muestran como ajustables, consulte Requesting a quota increase (Solicitud de aumento de cuota) en la Guía del usuario de Service Quotas.

> **Note**
>
> Service Quotas no admite ajustes en los siguientes componentes: * Asociaciones de Pod Identity por clúster. Para conocer los límites, consulte the section called “Pod Identity”.
> * CIDR para redes de nodos remotos o redes de pods remotos para nodos híbridos. Para conocer los límites, consulte the section called “Nodos híbridos”.

Cuotas de servicio de AWS Fargate

Este servicio de Fargate AWS en la consola de Service Quotas enumeran varias cuotas de servicio. Puede configurar alarmas que le avisen cuando su uso se acerque a una Service Quota. Para obtener más información, consulte the section called “Creación de una alarma de CloudWatch para monitorear las métricas de uso de recursos de Fargate”.

Las cuentas nuevas de AWS pueden tener cuotas iniciales más bajas que pueden aumentar con el tiempo. Fargate supervisa constantemente el uso de la cuenta dentro de cada región de AWS y luego aumenta automáticamente las cuotas en función de su uso. También puede solicitar un aumento de cuota para los valores que se muestran como ajustables. Para obtener más información, consulte Solicitud de aumento de cuota en la Guía del usuario de Service Quotas.

AWS recomienda utilizar la Consola de administración de AWS para ver las cuotas actuales. Para obtener más información, consulte the section called “Consulta de las Service Quotas de EKS en la Consola de administración de AWS”.

Para ver las Service Quotas predeterminadas de AWS Fargate en EKS, consulte Service Quotas de Fargate en la Referencia general de AWS.

> **Note**
>
> Fargate aplica adicionalmente las tareas de Amazon ECS y las cuotas de la tasa de lanzamiento de pods de Amazon EKS. Para obtener más información, consulte Cuotas de limitación de AWS Fargate en la Guía de Amazon ECS.

Seguridad en Amazon EKS

La seguridad en la nube de AWS es la mayor prioridad. Como cliente de AWS, se beneficia de una arquitectura de red y un centro de datos que se han diseñado para satisfacer los requisitos de seguridad de las organizaciones más exigentes.

La seguridad es una responsabilidad compartida entre AWS y el usuario. El modelo de responsabilidad compartida la describe como seguridad de la nube y seguridad en la nube:

- Seguridad en la nube: AWS es responsable de proteger la infraestructura que ejecuta los servicios de AWS en la nube de AWS. En Amazon EKS, AWS es responsable del plano de control de Kubernetes, que incluye los nodos del plano de control y la base de datos `etcd`. Auditores externos prueban y verifican periódicamente la eficacia de nuestra seguridad en el marco de los programas de conformidad de AWS. Para obtener más información sobre los programas de conformidad que se aplican a Amazon EKS, consulte Servicios de AWS en el ámbito del programa de conformidad.
- Seguridad en la nube: las siguientes áreas son su responsabilidad.
  - La configuración de seguridad del plano de datos, incluida la configuración de los grupos de seguridad que permiten que el tráfico pase del plano de control de Amazon EKS a la VPC del cliente
  - La configuración de los nodos y los contenedores
  - El sistema operativo de los nodos (incluidas las actualizaciones y los parches de seguridad)
  - Otros software de aplicaciones asociado:
    - Configuración y administración de controles de red, como las reglas del firewall
    - Administración de identidad y acceso de nivel de plataforma, con o además de IAM
  - La confidencialidad de los datos, los requisitos de la empresa y la legislación y los reglamentos aplicables.

Amazon EKS cuenta con certificaciones de programas de cumplimiento, adecuadas para aplicaciones reguladas y confidenciales. Amazon EKS cumple con las normas SOC, PCI, ISO, FedRAMP-Moderate, IRAP, C5, K-ISMS, ENS High, OSPAR y HITRUST CSF. Además, es un servicio que cumple con los requisitos de la HIPAA. Para obtener más información, consulte Administración de acceso.

Esta documentación lo ayuda a comprender cómo aplicar el modelo de responsabilidad compartida cuando se utiliza Amazon EKS. En los siguientes temas, se mostrará cómo configurar Amazon EKS para satisfacer sus objetivos de seguridad y conformidad. También puede aprender a utilizar otros servicios de AWS que ayudan a monitorear y proteger los recursos de Amazon EKS.

> **Note**
>
> Los contenedores de Linux se componen de grupos de control (cgroups) y espacios de nombres que ayudan a limitar el acceso de un contenedor, pero todos los contenedores comparten el mismo kernel de Linux que la instancia de Amazon EC2 anfitrión. No se recomienda ejecutar un contenedor como usuario raíz (UID 0) o conceder acceso a un contenedor a recursos o espacios de nombres de anfitrión como la red de anfitrión o el espacio de nombres PID de anfitrión, ya que al hacerlo se reduce la eficacia del aislamiento que proporcionan los contenedores.

Temas

- Protección de los clústeres de Amazon EKS con las prácticas recomendadas
- Análisis de vulnerabilidades en Amazon EKS
- Validación de la conformidad para clústeres de Amazon EMR
- Consideraciones de seguridad para Amazon Elastic Kubernetes Service
- Consideraciones de seguridad para Kubernetes
- Consideraciones de seguridad para el modo automático de Amazon EKS
- Consideraciones sobre la seguridad para las capacidades de EKS
- Administración de identidades y accesos para Amazon EKS

Protección de los clústeres de Amazon EKS con las prácticas recomendadas

Las prácticas recomendadas de seguridad de Amazon EKS se encuentran en las Prácticas recomendadas para la seguridad de la Guía de prácticas recomendadas de Amazon EKS.

Análisis de vulnerabilidades en Amazon EKS

La seguridad es una consideración fundamental para configurar y mantener clústeres y aplicaciones de Kubernetes. A continuación, se enumeran los recursos disponibles para que pueda analizar la configuración de seguridad de sus clústeres de EKS y comprobar si hay vulnerabilidades, y las integraciones con servicios de AWS que pueden realizar ese análisis por usted.

Referencia del Center for Internet Security (CIS, Centro para la seguridad de Internet) para Amazon EKS

El Punto de referencia de Kubernetes del Centro para la seguridad de Internet (CIS) proporciona orientación para las configuraciones de seguridad de nodos de Amazon EKS. El punto de referencia:

- Es aplicable a los nodos de Amazon EC2 (administrados y autoadministrados) donde es responsable de las configuraciones de seguridad de los componentes de Kubernetes.
- Proporciona una forma estándar y aprobada por la comunidad de garantizar que ha configurado de forma segura el clúster y los nodos de Kubernetes al utilizar Amazon EKS.
- Consta de cuatro secciones: configuración de registro del plano de control, configuraciones de seguridad de nodos, políticas y servicios administrados.
- Admite todas las versiones de Kubernetes actualmente disponibles en Amazon EKS y se puede ejecutar con kube-bench, una herramienta estándar de código abierto para verificar la configuración mediante el punto de referencia del CIS en clústeres de Kubernetes.

Para obtener más información, consulte Presentación del punto de referencia de Amazon EKS del CIS.

Para que una canalización de `aws-sample` automatizada actualice el grupo de nodos con una AMI de referencia de CIS, consulte EKS-Optimized AMI Hardening Pipeline.

Versiones de la plataforma de Amazon EKS

Las versiones de la plataforma de Amazon EKS representan las capacidades del plano de control del clúster, lo que incluye las marcas de servidor de la API de Kubernetes que se encuentran habilitadas y la versión de parche de Kubernetes actual. Los nuevos clústeres están implementados con la versión más reciente de la plataforma. Para obtener más información, consulte las versiones de la plataforma de EKS.

Puede actualizar un clúster de Amazon EKS con las versiones más recientes de Kubernetes. Cuando haya nuevas versiones de Kubernetes disponibles en Amazon EKS, le recomendamos que actualice proactivamente los clústeres para que utilicen la versión más reciente disponible. Para obtener más información sobre las versiones de Kubernetes en EKS, consulte las versiones compatibles de Amazon EKS.

Lista de vulnerabilidades del sistema operativo

Lista de vulnerabilidades de AL2023

Realice un seguimiento de los eventos de seguridad o privacidad de Amazon Linux 2023 en el Centro de seguridad de Amazon Linux o suscríbase a la fuente RSS asociada. Los eventos de seguridad y privacidad incluyen una descripción general del problema, los paquetes afectados e instrucciones para actualizar las instancias y corregir el problema.

Lista de vulnerabilidades Amazon Linux 2

Realice un seguimiento de los eventos de seguridad o privacidad de Amazon Linux 2 en el Centro de seguridad de Amazon Linux o suscríbase a la fuente RSS asociada. Los eventos de seguridad y privacidad incluyen una descripción general del problema, los paquetes afectados e instrucciones para actualizar las instancias y corregir el problema.

Detección de nodos con Amazon Inspector

Puede utilizar Amazon Inspector para verificar la accesibilidad de la red no deseada de sus nodos y a fin de buscar vulnerabilidades en dichas instancias de Amazon EC2.

Detección de clústeres y nodos con Amazon GuardDuty

Amazon GuardDuty es un servicio de detección de amenazas que ayuda a proteger las cuentas, los contenedores, las cargas de trabajo y los datos de su entorno de AWS. Entre otras características, GuardDuty ofrece las siguientes dos características que detectan posibles amenazas para sus clústeres de EKS: Protección de EKS y Supervisión en tiempo de ejecución.

Para obtener más información, consulte the section called “Amazon GuardDuty”.

Validación de la conformidad para clústeres de Amazon EMR

Para saber si un servicio de AWS está incluido en el ámbito de programas de conformidad específicos, consulte Servicios de AWS en el ámbito del programa de conformidad y escoja el

programa de conformidad que le interese. Para obtener información general, consulte Programas de conformidad de AWS.

Puede descargar los informes de auditoría de terceros mediante AWS Artifact. Para obtener más información, consulte Descarga de informes en AWS Artifact.

Su responsabilidad de cumplimiento al utilizar servicios de AWS está determinada por la sensibilidad de sus datos, los objetivos de cumplimiento de su empresa y las leyes y las regulaciones aplicables. AWS proporciona los siguientes recursos para ayudar con el cumplimiento:

- Cumplimiento de seguridad y gobernanza: en estas guías se explican las consideraciones de arquitectura y se proporcionan pasos para implementar las características de seguridad y cumplimiento.
- Referencia de servicios válidos de HIPAA: muestra una lista con los servicios válidos de HIPAA. No todos los servicios de AWS son aptos para HIPAA.
- AWSRecursos de conformidad: este conjunto de manuales y guías podría aplicarse a su sector y ubicación.
- Guías de cumplimiento para clientes de AWS: comprenda el modelo de responsabilidad compartida desde el punto de vista del cumplimiento. Las guías resumen las prácticas recomendadas para garantizar la seguridad de los servicios de AWS y orientan los controles de seguridad en varios marcos (incluidos el Instituto Nacional de Estándares y Tecnología [NIST], el Consejo de Estándares de Seguridad de la Industria de Tarjetas de Pago [PCI] y la Organización Internacional de Normalización [ISO]).
- Evaluación de recursos con reglas en la guía para desarrolladores de AWS Config: el servicio AWS Config evalúa en qué medida las configuraciones de los recursos cumplen con las prácticas internas, las directrices del sector y las normativas.
- AWS Security Hub: este producto de AWS proporciona una visión completa de su estado de seguridad en AWS. Security Hub utiliza controles de seguridad para evaluar sus recursos de AWS y comprobar su cumplimiento con los estándares y las prácticas recomendadas del sector de la seguridad. Para obtener una lista de los servicios y controles compatibles, consulte la Referencia de controles de Security Hub.
- Amazon GuardDuty: este servicio de AWS detecta posibles amenazas para sus cuentas, cargas de trabajo, contenedores y datos de AWS mediante el monitoreo de su entorno para detectar actividades sospechosas y maliciosas. GuardDuty puede ayudarlo a satisfacer varios requisitos de conformidad, como PCI DSS, cumpliendo los requisitos de detección de intrusos que exigen determinados marcos de conformidad.

- AWS Audit Manager: este servicio de AWS le ayuda a auditar de manera continua su uso de AWS para simplificar la forma en que administra el riesgo y la conformidad con las regulaciones y los estándares del sector.

Consideraciones de seguridad para Amazon Elastic Kubernetes Service

A continuación, se presentan consideraciones de seguridad en la nube que impactan en Amazon EKS.

Temas

- Seguridad de la infraestructura de Amazon EKS
- Comprensión de la resiliencia de los clústeres de Amazon EKS
- Prevención de suplentes confusos entre servicios en Amazon EKS

Seguridad de la infraestructura de Amazon EKS

Como servicio administrado, Amazon Elastic Kubernetes Service está protegido por la seguridad de la red global AWS. Para obtener información sobre los servicios de seguridad de AWS y sobre cómo AWS protege la infraestructura, consulte Seguridad en la nube de AWS. Para diseñar su entorno de AWS siguiendo las prácticas recomendadas de seguridad de infraestructura, consulte Protección de la infraestructura en Portal de seguridad de AWS Well-Architected Framework.

Puede utilizar llamadas a la API publicadas en AWS para acceder a Amazon EKS a través de la red. Los clientes deben admitir lo siguiente:

- Seguridad de la capa de transporte (TLS). Exigimos TLS 1.2 y recomendamos TLS 1.3.
- Conjuntos de cifrado con confidencialidad directa total (PFS) como DHE (Ephemeral Diffie-Hellman) o ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). La mayoría de los sistemas modernos como Java 7 y posteriores son compatibles con estos modos.

Además, las solicitudes deben estar firmadas mediante un ID de clave de acceso y una clave de acceso secreta que esté asociada a una entidad principal de IAM. También puede utilizar el AWS Security Token Service (AWS STS) con el objeto de generar credenciales de seguridad temporales para firmar solicitudes.

Cuando se crea un clúster de Amazon EKS, se especifican las subredes de la VPC que utilizará el clúster. Amazon EKS requiere subredes en al menos dos zonas de disponibilidad. Recomendamos una VPC con subredes públicas y privadas para que Kubernetes pueda crear equilibradores de carga públicos en las subredes públicas que equilibren la carga de tráfico con los pods que se ejecutan en los nodos de las subredes privadas.

Para obtener información acerca de las consideraciones de VPC, consulte the section called “Requisitos de VPC y subred”.

Si crea la VPC y los grupos de nodos con las plantillas de AWS CloudFormation incluidas en la explicación de Introducción a Amazon EKS, los grupos de seguridad del plano de control y de los nodos se ajustan con la configuración recomendada.

Para obtener más información acerca de las consideraciones del grupo de seguridad, consulte the section called “Requisitos del grupo de seguridad”.

Cuando se crea un clúster nuevo, Amazon EKS crea un punto de enlace para el servidor de la API de Kubernetes administrado que utiliza a fin de comunicarse con su clúster (mediante herramientas de administración de Kubernetes como, por ejemplo, `kubectl`). De forma predeterminada, este punto de conexión del servidor de API es público en Internet y el acceso al servidor de API está protegido mediante una combinación de AWS Identity and Access Management (IAM) y el Control de acceso basado en rol (RBAC) nativo de Kubernetes.

Puede habilitar el acceso privado al servidor de la API de Kubernetes para que toda la comunicación entre los nodos y el servidor de la API permanezcan dentro de su VPC. Puede limitar las direcciones IP que pueden acceder a su servidor de API desde Internet o desactivar por completo el acceso a Internet al servidor de API.

Para obtener más información acerca de la modificación del acceso al punto de conexión del clúster, consulte the section called “Modificar el acceso al punto de conexión del clúster”.

Puede implementar políticas de red de Kubernetes con la CNI de Amazon VPC o con herramientas de terceros, como Project Calico. Para obtener más información sobre el CNI de Amazon VPC para las políticas de red, consulte the section called “Políticas de Kubernetes”. Project Calico es un proyecto abierto de terceros. Para obtener más información, consulte la documentación de Project Calico.

Acceso a Amazon EKS con AWS PrivateLink

Puede usar un AWS PrivateLink para crear una conexión privada entre la VPC y Amazon Elastic Kubernetes Service. Puede acceder a Amazon EKS como si estuviera en su VPC, sin el uso de una puerta de enlace de Internet, un dispositivo NAT, una conexión VPN o una conexión AWS Direct Connect. Las instancias de la VPC no necesitan direcciones IP públicas para acceder a Amazon EKS.

Esta conexión privada se establece mediante la creación de un punto de conexión de interfaz alimentado por AWS PrivateLink. Creamos una interfaz de red de punto de conexión en cada subred habilitada para el punto de conexión de interfaz. Se trata de interfaces de red administradas por el solicitante que sirven como punto de entrada para el tráfico destinado a Amazon EKS.

Para obtener más información, consulte Access AWS services through AWS PrivateLink en la Guía de AWS PrivateLink.

Antes de empezar

Antes de comenzar, asegúrese de haber realizado las siguientes tareas:

- Revise Acceda a un servicio de AWS mediante un punto de conexión de VPC de interfaz en la Guía de AWS PrivateLink

Consideraciones

- Compatibilidad y limitaciones: los puntos de conexión de la interfaz de Amazon EKS permiten el acceso seguro a todas las acciones de la API de Amazon EKS desde su VPC, pero tienen limitaciones específicas: no admiten el acceso a las API de Kubernetes, ya que estas tienen un punto de conexión privado independiente, no puede configurar Amazon EKS para que solo se pueda acceder a través del punto de conexión de la interfaz.
- Precios: el uso de puntos de conexión de la interfaz para Amazon EKS conlleva cargos estándar de AWS PrivateLink: cargos por hora por cada punto de conexión aprovisionado en cada zona de disponibilidad, cargos por procesamiento de datos por el tráfico a través del punto de conexión. Para obtener más información, consulte Precios de AWS PrivateLink.
- Seguridad y control de acceso: recomendamos mejorar la seguridad y controlar el acceso con estas configuraciones adicionales: utilice políticas de puntos de conexión de VPC para controlar el acceso a Amazon EKS a través del punto de conexión de la interfaz, asocie grupos de seguridad a las interfaces de red de los puntos de conexión para gestionar el tráfico, utilice registros de flujo

de VPC para capturar y supervisar el tráfico IP hacia y desde los puntos de conexión de la interfaz, con registros que se puedan publicar en Amazon CloudWatch o Amazon S3. Para obtener más información, consulte Uso de políticas de punto de conexión para controlar el acceso a puntos de conexión de VPC y Registro del tráfico de IP con registros de flujo de la VPC.

- Opciones de conectividad: los puntos de conexión de la interfaz ofrecen opciones de conectividad flexibles mediante el acceso local (conecte el centro de datos en las instalaciones a una VPC con el punto de conexión de la interfaz mediante AWS Direct Connect o AWS Site-to-Site VPN) o mediante la conectividad entre VPC (utilice AWS Transit Gateway o el emparejamiento de VPC para conectar otras VPC a la VPC con el punto de conexión de la interfaz y mantener el tráfico dentro de la red de AWS).
- Compatibilidad con la versión IP: los puntos de conexión creados antes de agosto de 2024 solo admiten IPv4 con eks.region.amazonaws.com. Los nuevos puntos de conexión creados después de agosto de 2024 admiten IPv4 e IPv6 de doble pila (por ejemplo, eks.region.amazonaws.com, eks.region.api.aws).
- Disponibilidad regional: AWS PrivateLink para la API de EKS no está disponible en las regiones de Asia-Pacífico (Malasia) (ap-southeast-5), Asia-Pacífico (Tailandia) (ap-southeast-7), México (centro) (mx-central-1) y Asia-Pacífico (Taipéi) (ap-east-2). AWS La compatibilidad de PrivateLink con eks-auth (EKS Pod Identity) se encuentra disponible en la región de Asia-Pacífico (Malasia) (ap-southeast-5).

Crear de un punto de conexión de interfaz para Amazon EKS

Puede crear un punto de conexión de interfaz para Amazon EKS mediante la consola de Amazon VPC o la Interfaz de la línea de comandos de AWS (AWS CLI). Para obtener más información, consulte Creación de un punto de conexión de VPC en la Guía de AWS PrivateLink.

Cree un punto de conexión de interfaz para Amazon EKS con los siguientes nombres de servicios:

API de EKS

- com.amazonaws.region-code.eks
- com.amazonaws.region-code.eks-fips (para puntos de conexión compatibles con FIPS)

API de autenticación de EKS (Pod Identity de EKS)

- com.amazonaws.region-code.eks-auth

Característica de DNS privado para los puntos de conexión de la interfaz de Amazon EKS

La característica de DNS privado, habilitada de forma predeterminada para los puntos de conexión de la interfaz de Amazon EKS y otros servicios de AWS, facilita las solicitudes de API seguras y privadas mediante nombres de DNS regionales predeterminados. Esta característica garantiza que las llamadas de API se enruten a través del punto de conexión de la interfaz a través de la red de AWS privada, lo que mejora la seguridad y el rendimiento.

La característica de DNS privado se activa automáticamente cuando crea un punto de conexión de la interfaz para Amazon EKS u otros servicios de AWS. Para habilitarla, debe configurar la VPC correctamente mediante el establecimiento de atributos específicos:

- enableDnsHostnames: Permite que las instancias dentro de la VPC tengan nombres de host DNS.
- enableDnsSupport: Habilita la resolución de DNS en toda la VPC.

Si desea obtener instrucciones paso a paso para comprobar o modificar estos ajustes, consulte Ver y actualizar los atributos de DNS de su VPC.

Nombres de DNS y tipos de direcciones IP

Con la característica de DNS privado habilitada, puede usar nombres de DNS específicos para conectarse a Amazon EKS, y estas opciones evolucionan con el tiempo:

- eks.region.amazonaws.com: El nombre de DNS tradicional, que solo se resuelve en direcciones IPv4 antes de agosto de 2024. Para los puntos de conexión existentes actualizados a doble pila, este nombre se resuelve en direcciones IPv4 e IPv6.
- eks.region.api.aws: Disponible para los nuevos puntos de conexión creados después de agosto de 2024, este nombre de DNS de doble pila se resuelve en direcciones IPv4 e IPv6.

A partir de agosto de 2024, los nuevos puntos de conexión de la interfaz vienen con dos nombres de DNS y puede optar por el tipo de dirección IP de doble pila. Para los puntos de conexión existentes, la actualización a doble pila modifica eks.region.amazonaws.com para que sea compatible con IPv4 e IPv6.

Uso de la característica de DNS privado

Una vez configurada, la característica de DNS privado se puede integrar en sus flujos de trabajo y ofrece las siguientes capacidades:

- Solicitudes de API: Utilice los nombres de DNS regionales predeterminados, ya sea `eks.region.amazonaws.com` o `eks.region.api.aws`, según la configuración de su punto de conexión, para realizar solicitudes de API a Amazon EKS.
- Compatibilidad con aplicaciones: Las aplicaciones existentes que utilizan las API de EKS no requieren cambios para aprovechar esta característica.
- AWS CLI con doble pila: Para usar los puntos de conexión de doble pila con AWS CLI, consulte la configuración de los puntos de conexión de doble pila y FIPS en la Guía de referencia de SDK y herramientas de AWS.
- Enrutamiento automático: Cualquier llamada al punto de conexión del servicio predeterminado de Amazon EKS se enruta automáticamente a través del punto de conexión de interfaz, lo que garantiza una conectividad privada y segura.

Comprensión de la resiliencia de los clústeres de Amazon EKS

La infraestructura global de AWS se compone de regiones y zonas de disponibilidad de AWS. AWS Las regiones proporcionan varias zonas de disponibilidad físicamente independientes y aisladas que se encuentran conectadas mediante redes con un alto nivel de rendimiento y redundancia, además de baja latencia. Con las zonas de disponibilidad, puedes diseñar y utilizar aplicaciones y bases de datos que realizan una conmutación por error automática entre zonas de disponibilidad sin interrupciones. Las zonas de disponibilidad tienen una mayor disponibilidad, tolerancia a errores y escalabilidad que las infraestructuras tradicionales de centros de datos únicos o múltiples.

Amazon EKS ejecuta y escala el plano de control de Kubernetes en varias zonas de disponibilidad de AWS para garantizar una alta disponibilidad. Amazon EKS escala de forma automática las instancias del plano de control en función de la carga, detecta y reemplaza instancias del plano de control en mal estado y revisa el plano de control de forma automática. Después de iniciar una actualización de versión, Amazon EKS actualiza el plano de control en su nombre y mantiene una alta disponibilidad durante la actualización.

Este plano de control consta de al menos dos instancias de servidor de la API y tres instancias `etcd` que se ejecutan en tres zonas de disponibilidad de una región de AWS. Amazon EKS:

- Monitorea de forma activa la carga en las instancias del plano de control y las escala de forma automática para garantizar un alto rendimiento.
- Detecta y reemplaza de forma automática las instancias del plano de control en mal estado y las reinicia en las zonas de disponibilidad de la región de AWS, según sea necesario.

- Aprovecha la arquitectura de las regiones de AWS con el fin de mantener una alta disponibilidad. Por este motivo, Amazon EKS puede ofrecer un acuerdo de nivel de servicio (SLA) de disponibilidad del punto de enlace del servidor de la API.

Para obtener más información sobre las zonas de disponibilidad y las regiones de AWS, consulte Infraestructura global de AWS.

Prevención de suplentes confusos entre servicios en Amazon EKS

El problema del suplente confuso es un problema de seguridad en el que una entidad que no tiene permiso para realizar una acción puede obligar a una entidad con más privilegios a realizar la acción. En AWS, la suplantación entre servicios puede dar lugar al problema de la sustitución confusa. La suplantación entre servicios puede producirse cuando un servicio (el servicio que lleva a cabo las llamadas) llama a otro servicio (el servicio al que se llama). El servicio que lleva a cabo las llamadas se puede manipular para utilizar sus permisos a fin de actuar en función de los recursos de otro cliente de una manera en la que no debe tener permiso para acceder. Para evitarlo, AWS proporciona herramientas que le ayudan a proteger sus datos para todos los servicios con entidades principales de servicio a las que se les ha dado acceso a los recursos de su cuenta.

Se recomienda utilizar las claves de contexto de condición global `aws:SourceArn` y `aws:SourceAccount` en las políticas de recursos para limitar los permisos que Amazon Elastic Kubernetes Service (Amazon EKS) concede a otro servicio para el recurso.

`aws:SourceArn`

Utilice `aws:SourceArn` para asociar solo un recurso al acceso entre servicios.

`aws:SourceAccount`

Utilice `aws:SourceAccount` para permitir que cualquier recurso de esa cuenta se asocie al uso entre servicios.

La forma más eficaz de protegerse contra el problema de la sustitución confusa es utilizar la clave de contexto de condición global de `aws:SourceArn` con el ARN completo del recurso. Si no conoce el ARN completo del recurso o si está especificando varios recursos, utilice la clave de condición de contexto global `aws:SourceArn` con caracteres comodín (*) para las partes desconocidas del ARN. Por ejemplo, `arn:aws:<servicename>:*:<123456789012>:*`.

Si el valor de `aws:SourceArn` no contiene el ID de cuenta, como un ARN de bucket de Amazon S3, debe utilizar `aws:SourceAccount` y `aws:SourceArn` para limitar los permisos.

Prevención de suplentes confusos entre servicios para el rol del clúster de Amazon EKS

Se requiere un rol de IAM de clúster de Amazon EKS para cada clúster. Los clústeres de Kubernetes administrados por Amazon EKS utilizan este rol para administrar los nodos, y el proveedor de nube heredado lo usa para crear equilibradores de carga con Elastic Load Balancing para los servicios. Estas acciones de clúster solo pueden afectar a la misma cuenta, por lo que recomendamos que limite cada rol del clúster a ese clúster y esa cuenta. Esta es una aplicación específica de la recomendación de AWS de seguir el principio de privilegios mínimos en su cuenta.

Formato del ARN de origen

El valor de `aws:SourceArn` debe ser el ARN de un clúster de EKS con el formato `arn:aws:eks:`*`region`*`:`*`account`*`:cluster/`*`cluster-name`* . Por ejemplo, ., `arn:aws:eks:us-west-2:123456789012:cluster/my-cluster` .

Formato de la política de confianza para roles de clúster de EKS

En el ejemplo siguiente se muestra cómo se pueden utilizar las claves de contexto de condición globales `aws:SourceArn` y `aws:SourceAccount` en Amazon EKS para evitar el problema de suplente confuso.

```
{
  "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"
        }
```

```
      }
    }
  ]
}
```

Consideraciones de seguridad para Kubernetes

A continuación, se presentan consideraciones de seguridad en la nube que afectan a Kubernetes en clústeres de Amazon EKS. Para obtener una revisión exhaustiva de los controles y prácticas de seguridad de Kubernetes, consulte Cloud Native Security and Kubernetes en la documentación de Kubernetes.

Temas

- Protección de las cargas de trabajo con certificados de Kubernetes
- Comprensión de roles y usuarios de RBAC creados por Amazon EKS
- Cifrado de los secretos de Kubernetes con KMS en los clústeres existentes
- Uso de secretos de AWS Secrets Manager con pods de Amazon EKS
- Cifrado de sobre predeterminado para todos los datos de la API de Kubernetes

Protección de las cargas de trabajo con certificados de Kubernetes

La API de certificados de Kubernetes automatiza el aprovisionamiento de credenciales X.509. La API cuenta con una interfaz de línea de comandos para que los clientes API de Kubernetes soliciten y obtengan certificados X.509 de una entidad de certificación (CA). Se puede usar un recurso `CertificateSigningRequest` (CSR) para solicitar que un firmante indicado firme el certificado. Las solicitudes se aprueban o se rechazan antes de que se firmen. Kubernetes admite firmantes integrados y personalizados con comportamientos bien definidos. De esta forma, los clientes pueden predecir qué sucede con sus CSR. Para obtener más información sobre la firma de certificados, consulte acerca de la firma de solicitudes.

Uno de los firmantes integrados es `kubernetes.io/legacy-unknown`. La API `v1beta1` del recurso de CSR honró a este firmante desconocido de legado. Sin embargo, la API estable `v1` de CSR no permite que el `signerName` se establezca en `kubernetes.io/legacy-unknown`.

Si desea utilizar la CA de Amazon EKS para generar certificados en sus clústers, debe usar un firmante personalizado. Para utilizar la versión de la API de CSR `v1` y generar un nuevo certificado,

debe migrar cualquier manifiesto y cliente API existentes. Los certificados existentes creados con las API `v1beta1` anteriores son válidas y funcionan hasta que caduque el certificado. Esta incluye lo siguiente:

- Distribución de confianza: ninguna. No hay confianza o distribución estándar para este firmante en un clúster de Kubernetes.
- Temas permitidos: cualquiera
- Extensiones x509 permitidas: honra las extensiones de uso de subjectAltName y clave y descarta otras extensiones
- Usos de clave permitidos: no debe incluir usos más allá de [“cifrado de clave”, “firma digital”, “autenticación del servidor”]

> **Note**
>
> No se admite la firma de certificados de cliente.

- Vida útil del certificado/caducidad: 1 año (predeterminado y máximo)
- Bit CA permitido/no permitido: no permitido

Generación de CSR de ejemplo con signerName

En estos pasos se muestra cómo generar un certificado de publicación para el nombre de DNS `myserver.default.svc` con `signerName: beta.eks.amazonaws.com/app-serving`. Úselo como guía para su propio entorno.

1. Ejecute el comando `openssl genrsa -out myserver.key 2048` para generar una clave privada RSA.

```
openssl genrsa -out myserver.key 2048
```

2. Utilice el siguiente comando para generar una solicitud de certificado.

```
openssl req -new -key myserver.key -out myserver.csr -subj "/CN=myserver.default.svc"
```

3. Genere un valor `base64` para la CSR y almacénelo en una variable para utilizarlo en un paso posterior.

```
base_64=$(cat myserver.csr | base64 -w 0 | tr -d "
```

```
")
```

4. Ejecute el siguiente comando para crear un archivo llamado `mycsr.yaml`. En el siguiente ejemplo, `beta.eks.amazonaws.com/app-serving` es el `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. Envíe la CSR.

```
kubectl apply -f mycsr.yaml
```

6. Apruebe el certificado de entrega.

```
kubectl certificate approve myserver
```

7. Compruebe que se emitió el certificado.

```
kubectl get csr myserver
```

   Un ejemplo de salida sería el siguiente.

```
NAME       AGE     SIGNERNAME                            REQUESTOR          CONDITION
myserver   3m20s   beta.eks.amazonaws.com/app-serving   kubernetes-admin
 Approved,Issued
```

8. Exporte el certificado emitido.

```
kubectl get csr myserver -o jsonpath='{.status.certificate}'| base64 -d >
 myserver.crt
```

Comprensión de roles y usuarios de RBAC creados por Amazon EKS

Al crear un clúster de Kubernetes, se crean varias identidades de Kubernetes predeterminadas en ese clúster para el correcto funcionamiento de Kubernetes. Amazon EKS crea identidades de Kubernetes para cada uno de sus componentes predeterminados. Las identidades proporcionan un control de autorización basado en roles (RBAC) de Kubernetes para los componentes del clúster. Para obtener más información, consulte Utilización de la autorización de RBAC en la documentación de Kubernetes.

Al instalar complementos opcionales en el clúster, es posible que se agreguen identidades de Kubernetes adicionales al clúster. Para obtener más información sobre las identidades no abordadas en este tema, consulte la documentación del complemento.

Puede ver la lista de identidades de Kubernetes creadas por Amazon EKS en su clúster mediante la Consola de administración de AWS o la herramienta de línea de comandos de `kubectl`. Todas las identidades de usuario aparecen en los registros de auditoría de `kube` disponibles para los clientes a través de Amazon CloudWatch.

Consola de administración de AWS

Requisito previo

La entidad principal de IAM que utilice debe tener los permisos que se describen en Permisos necesarios.

Para ver las identidades creadas por Amazon EKS mediante la Consola de administración de AWS

1. Abra la consola de Amazon EKS.
2. En la lista Clusters (Clústeres), seleccione el clúster que contiene las identidades que desea ver.
3. Elija la pestaña Recursos.
4. En Resource types (Tipos de recursos), elija Authorization (Autorización).
5. Elija ClusterRoles, ClusterRoleBindings, Roles o RoleBindings. Amazon EKS crea todos los recursos con el prefijo eks. Los recursos de identidad adicionales creados por Amazon EKS son los siguientes:
   - El ClusterRole y el ClusterRoleBinding, que se denominan aws-node. Los recursos de aws-node admiten el complemento de la CNI de Amazon VPC para Kubernetes, que Amazon EKS instala en todos los clústeres.

- Un ClusterRole llamado vpc-resource-controller-role y un ClusterRoleBinding llamado vpc-resource-controller-rolebinding. Estos recursos son compatibles con el controlador de recursos de Amazon VPC, que Amazon EKS instala en todos los clústeres.

Además de los recursos que ve en la consola, existen las siguientes identidades de usuario especiales en su clúster, aunque no están visibles en la configuración del clúster:

- **`eks:cluster-bootstrap`** : se utiliza para operaciones de `kubectl` durante el arranque del clúster.
- **`eks:support-engineer`** – se utiliza para operaciones de administración de clústeres.

6. Elija un recurso específico para ver detalles sobre él. De forma predeterminada, se muestra la información en la Vista estructurada. En la esquina superior derecha de la página de detalles de la, elija la vista de sin procesar para ver toda la información del recurso.

Kubectl

Requisito previo

La entidad que utilice (AWS Identity and Access Management [IAM] u OpenID Connect [OIDC]) para enumerar los recursos de Kubernetes del clúster debe estar autenticada por IAM o por su proveedor de identidad de OIDC. Se deben conceder permisos a la entidad para usar los verbos de `get` y `list` de Kubernetes de los recursos del clúster `Role`, `ClusterRole`, `RoleBinding` y `ClusterRoleBinding` con los que desea que trabaje la entidad. Para obtener más información sobre cómo conceder acceso de entidades de IAM a su clúster, consulte the section called “Acceso a la API de Kubernetes”. Para obtener más información sobre cómo conceder acceso de entidades autenticadas mediante su propio proveedor de OIDC a su clúster, consulte the section called “Vinculación del proveedor de OIDC”.

Para ver las identidades creadas por Amazon EKS mediante **`kubectl`**

Ejecute el comando para el tipo de recurso que desea ver. Todos los recursos devueltos que llevan el prefijo eks son creados por Amazon EKS. Además de los recursos devueltos en la salida de los comandos, existen las siguientes identidades de usuario especiales en su clúster, aunque no están visibles en la configuración del clúster:

- **`eks:cluster-bootstrap`** : se utiliza para operaciones de `kubectl` durante el arranque del clúster.
- **`eks:support-engineer`** – se utiliza para operaciones de administración de clústeres.

ClusterRoles: `ClusterRoles` están dentro del ámbito de su clúster, por lo que cualquier permiso otorgado a un rol se aplica a los recursos de cualquier espacio de nombres de Kubernetes del clúster.

El siguiente comando devuelve todos los `ClusterRoles` de Kubernetes de Amazon EKS creados en su clúster.

```
kubectl get clusterroles | grep eks
```

Además de los `ClusterRoles` devueltos en la salida con el prefijo, existen los siguientes `ClusterRoles`.

- **`aws-node`** : este `ClusterRole` es compatible con el complemento de la CNI de Amazon VPC para Kubernetes, que Amazon EKS instala en todos los clústeres.
- **`vpc-resource-controller-role`** : este `ClusterRole` es compatible con el Controlador de recursos de Amazon VPC, que Amazon EKS instala en todos los clústeres.

Para ver la especificación de un `ClusterRole`, sustituya *`eks:k8s-metrics`* en el siguiente comando por el `ClusterRole` devuelto en el resultado del comando anterior. El siguiente ejemplo devuelve la especificación de `ClusterRole` *`eks:k8s-metrics`*.

```
kubectl describe clusterrole eks:k8s-metrics
```

Un ejemplo de salida sería el siguiente.

```
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` está dentro del ámbito de su clúster.

El siguiente comando devuelve todos los `ClusterRoleBindings` de Kubernetes de Amazon EKS creados en su clúster.

```
kubectl get clusterrolebindings | grep eks
```

Además de los `ClusterRoleBindings` devueltos en la salida, existen los siguientes `ClusterRoleBindings`.

- **`aws-node`** : este `ClusterRoleBinding` es compatible con el complemento de la CNI de Amazon VPC para Kubernetes, que Amazon EKS instala en todos los clústeres.
- **`vpc-resource-controller-rolebinding`** : este `ClusterRoleBinding` es compatible con el Controlador de recursos de Amazon VPC, que Amazon EKS instala en todos los clústeres.

Para ver la especificación de un `ClusterRoleBinding`, sustituya *`eks:k8s-metrics`* en el siguiente comando por el `ClusterRoleBinding` devuelto en el resultado del comando anterior. El siguiente ejemplo devuelve la especificación de `ClusterRoleBinding` *`eks:k8s-metrics`*.

```
kubectl describe clusterrolebinding eks:k8s-metrics
```

Un ejemplo de salida sería el siguiente.

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  ClusterRole
  Name:  eks:k8s-metrics
Subjects:
  Kind  Name             Namespace
  ----  ----             ---------
  User  eks:k8s-metrics
```

Roles: `Roles` se limitan a un espacio de nombres de Kubernetes. Todos los `Roles` de Amazon EKS creados están incluidos en el espacio de nombres de `kube-system`.

El siguiente comando devuelve todos los `Roles` de Kubernetes de Amazon EKS creados en su clúster.

```
kubectl get roles -n kube-system | grep eks
```

Para ver la especificación de un `Role`, sustituya *`eks:k8s-metrics`* en el siguiente comando por el nombre del `Role` devuelto en el resultado del comando anterior. El siguiente ejemplo devuelve la especificación de `Role` *`eks:k8s-metrics`*.

```
kubectl describe role eks:k8s-metrics -n kube-system
```

Un ejemplo de salida sería el siguiente.

```
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` se circunscriben a un espacio de nombres de Kubernetes. Todos los `RoleBindings` de Amazon EKS creados están incluidos en el espacio de nombres de `kube-system`.

El siguiente comando devuelve todos los `RoleBindings` de Kubernetes de Amazon EKS creados en su clúster.

```
kubectl get rolebindings -n kube-system | grep eks
```

Para ver la especificación de un `RoleBinding`, sustituya *`eks:k8s-metrics`* en el siguiente comando por el `RoleBinding` devuelto en el resultado del comando anterior. El siguiente ejemplo devuelve la especificación de `RoleBinding` *`eks:k8s-metrics`*.

```
kubectl describe rolebinding eks:k8s-metrics -n kube-system
```

Un ejemplo de salida sería el siguiente.

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
Role:
```

```
  Kind:  Role
  Name:  eks:k8s-metrics
Subjects:
  Kind  Name             Namespace
  ----  ----             ---------
  User  eks:k8s-metrics
```

Cifrado de los secretos de Kubernetes con KMS en los clústeres existentes

> **Important**
>
> Este procedimiento se aplica únicamente a clústeres de EKS que ejecutan la versión 1.27 de Kubernetes o una inferior. Si se ejecuta Kubernetes versión 1.28 o superior, los secretos de Kubernetes están protegidos con cifrado de sobre de forma predeterminada. Para obtener más información, consulte the section called “Cifrado de sobre predeterminado para todos los datos de la API de Kubernetes”.

Si habilita el cifrado de secretos, los secretos de Kubernetes se cifran con la clave de AWS KMS que seleccione. La clave de KMS debe cumplir las siguientes condiciones:

- Simétrica
- Debe poder cifrar y descifrar datos
- Creada en la misma región de AWS que el clúster
- Si la clave de KMS se creó en una cuenta diferente, la entidad principal de IAM debe tener acceso a la clave de KMS.

Para obtener más información, consulte Permitir que las entidades principales de IAM de otras cuentas utilicen una clave de KMS en la Guía para desarrolladores de AWS Key Management Service.

> **Warning**
>
> No puede desactivar el cifrado de secretos después de habilitarlo. Esta acción es irreversible.

eksctl

Este procedimiento se aplica únicamente a clústeres de EKS que ejecutan la versión 1.27 de Kubernetes o una inferior. Para obtener más información, consulte the section called “Cifrado de sobre predeterminado para todos los datos de la API de Kubernetes”.

Puede habilitar el cifrado de dos formas:

- Agregue cifrado a su clúster con un solo comando.

  Para volver a cifrar los secretos de forma automática, ejecute el siguiente comando.

```
eksctl utils enable-secrets-encryption \
    --cluster my-cluster \
    --key-arn arn:aws:kms:region-code:account:key/key
```

  Para optar por no volver a cifrar los secretos de forma automática, ejecute el siguiente comando.

```
eksctl utils enable-secrets-encryption
    --cluster my-cluster \
    --key-arn arn:aws:kms:region-code:account:key/key \
    --encrypt-existing-secrets=false
```

- Agregue cifrado al clúster con un archivo `kms-cluster.yaml`.

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

metadata:
  name: my-cluster
  region: region-code

secretsEncryption:
  keyARN: arn:aws:kms:region-code:account:key/key
```

  Para que los secretos se vuelvan a cifrar automáticamente, ejecute el siguiente comando.

```
eksctl utils enable-secrets-encryption -f kms-cluster.yaml
```

  Para optar por no volver a cifrar los secretos de forma automática, ejecute el siguiente comando.

```
eksctl utils enable-secrets-encryption -f kms-cluster.yaml --encrypt-existing-
secrets=false
```

Consola de administración de AWS

a. Este procedimiento se aplica únicamente a clústeres de EKS que ejecutan la versión 1.27 de Kubernetes o una inferior. Para obtener más información, consulte the section called “Cifrado de sobre predeterminado para todos los datos de la API de Kubernetes”.

b. Abra la consola de Amazon EKS.

c. Elija el clúster al que desea agregar el cifrado de KMS.

d. Elija la pestaña Overview (Resumen) (está seleccionada de manera predeterminada).

e. Desplácese hasta la sección Secrets encryption (Cifrado de secretos) y elija Enable (Habilitar).

f. Seleccione una clave en el menú desplegable y elija el botón Enable (Habilitar). Si no aparece ninguna clave, primero debe crear una. Para obtener más información, consulte Creación de claves.

g. Elija el botón Confirm (Confirmar) para utilizar la clave elegida.

AWS CLI

a. Este procedimiento se aplica únicamente a clústeres de EKS que ejecutan la versión 1.27 de Kubernetes o una inferior. Para obtener más información, consulte the section called “Cifrado de sobre predeterminado para todos los datos de la API de Kubernetes”.

b. Asocie la configuración del cifrado de secretos con el clúster mediante el siguiente comando de la AWS CLI. Sustituya los valores de ejemplo por sus propios valores.

```
aws eks associate-encryption-config \
    --cluster-name my-cluster \
    --encryption-config '[{"resources":["secrets"],"provider":
{"keyArn":"arn:aws:kms:region-code:account:key/key"}}]'
```

Un ejemplo de salida sería el siguiente.

```
{
  "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. Puede monitorear el estado de la actualización de cifrado con el siguiente comando. Utilice la especificación `cluster name` y `update ID` que se devolvió en la salida anterior. Cuando se muestra el estado `Successful`, la actualización se ha completado.

```
aws eks describe-update \
    --region region-code \
    --name my-cluster \
    --update-id 3141b835-8103-423a-8e68-12c2521ffa4d
```

Un ejemplo de salida sería el siguiente.

```
{
  "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. Para verificar que el cifrado se encuentra habilitado en el clúster, ejecute el comando `describe-cluster`. La respuesta contiene una cadena de `EncryptionConfig`.

```
aws eks describe-cluster --region region-code --name my-cluster
```

Después de habilitar el cifrado en el clúster, deberá cifrar todos los secretos existentes con la clave nueva:

> **Note**
>
> Si usa `eksctl`, solo es necesario ejecutar el siguiente comando si opta por no volver a cifrar sus secretos automáticamente.

```
kubectl get secrets --all-namespaces -o json | kubectl annotate --overwrite -f - kms-
encryption-timestamp="time value"
```

> **Warning**
>
> Si habilita el cifrado de secretos para un clúster existente y, en algún momento, se elimina la clave de KMS que utiliza, no habrá forma de recuperar el clúster. Si elimina la clave de KMS, coloca el clúster permanentemente en un estado degradado. Para obtener más información, consulte Eliminación de claves de AWS KMS.

> **Note**
>
> De forma predeterminada, el comando `create-key` crea una clave KMS de cifrado simétrica con una política de claves que da al administrador raíz de la cuenta acceso a las acciones y los recursos de AWS KMS. Si desea reducir los permisos, asegúrese de que las acciones `kms:DescribeKey` y `kms:CreateGrant` estén permitidas en la política para la entidad principal que llama a la API `create-cluster`.
>
> Para clústeres que utilizan cifrado de sobres KMS, se requieren permisos `kms:CreateGrant`. La condición `kms:GrantIsForAWSResource` no es compatible con la acción CreateCluster y no debe utilizarse en las políticas de KMS para controlar permisos `kms:CreateGrant` para los usuarios que realizan CreateCluster.

Uso de secretos de AWS Secrets Manager con pods de Amazon EKS

A fin de mostrar secretos de Secrets Manager y parámetros del Almacén de parámetros como archivos montados en pods de Amazon EKS, puede utilizar el proveedor de secretos y configuración de AWS (ASCP) para el Controlador CSI del almacén de secretos de Kubernetes.

Con el ASCP, puede almacenar y administrar sus secretos en Secrets Manager y recuperarlos a través de sus cargas de trabajo que se ejecutan en Amazon EKS. Puede utilizar roles y políticas de IAM para limitar el acceso a sus secretos a pods de Kubernetes específicos de un clúster. El ASCP recupera la identidad del pod e intercambia la identidad por un rol de IAM. El ASCP asume el rol de IAM del pod y, a continuación, puede recuperar secretos de Secrets Manager autorizados para ese rol.

Si utiliza la rotación automática de Secrets Manager para sus secretos, también puede utilizar la característica de conciliador de rotación del controlador de CSI del almacén de secretos a fin de asegurarse de que está recuperando el último secreto de Secrets Manager.

> **Note**
>
> No se admiten los grupos de nodos de AWS Fargate (Fargate).

Para obtener más información, consulte Uso de secretos de Secrets Manager en Amazon EKS en la Guía del usuario de AWS Secrets Manager.

Cifrado de sobre predeterminado para todos los datos de la API de Kubernetes

Amazon Elastic Kubernetes Service (Amazon EKS) aplica cifrado de sobre de forma predeterminada a todos los datos de la API de Kubernetes en los clústeres que ejecutan la versión 1.28 o superior.

El cifrado de sobre protege los datos almacenados en el servidor de la API de Kubernetes. Por ejemplo, el cifrado de sobre se aplica a la configuración del clúster de Kubernetes, como `ConfigMaps`. El cifrado de sobre no se aplica a los datos almacenados en los nodos ni en los volúmenes de EBS. Anteriormente, EKS permitía cifrar los secretos de Kubernetes; ahora, este cifrado de sobre se extiende a la totalidad de los datos de la API de Kubernetes.

Esto proporciona una experiencia administrada y predeterminada que implementa un enfoque de defensa en profundidad para las aplicaciones de Kubernetes, sin requerir intervención adicional.

Amazon EKS utiliza AWS Key Management Service (KMS) junto con el proveedor KMS v2 de Kubernetes para ofrecer esta capa adicional de seguridad, mediante una clave administrada por Amazon Web Services, con la opción de emplear una clave administrada por el cliente (CMK) de AWS KMS.

Comprensión del cifrado de sobre

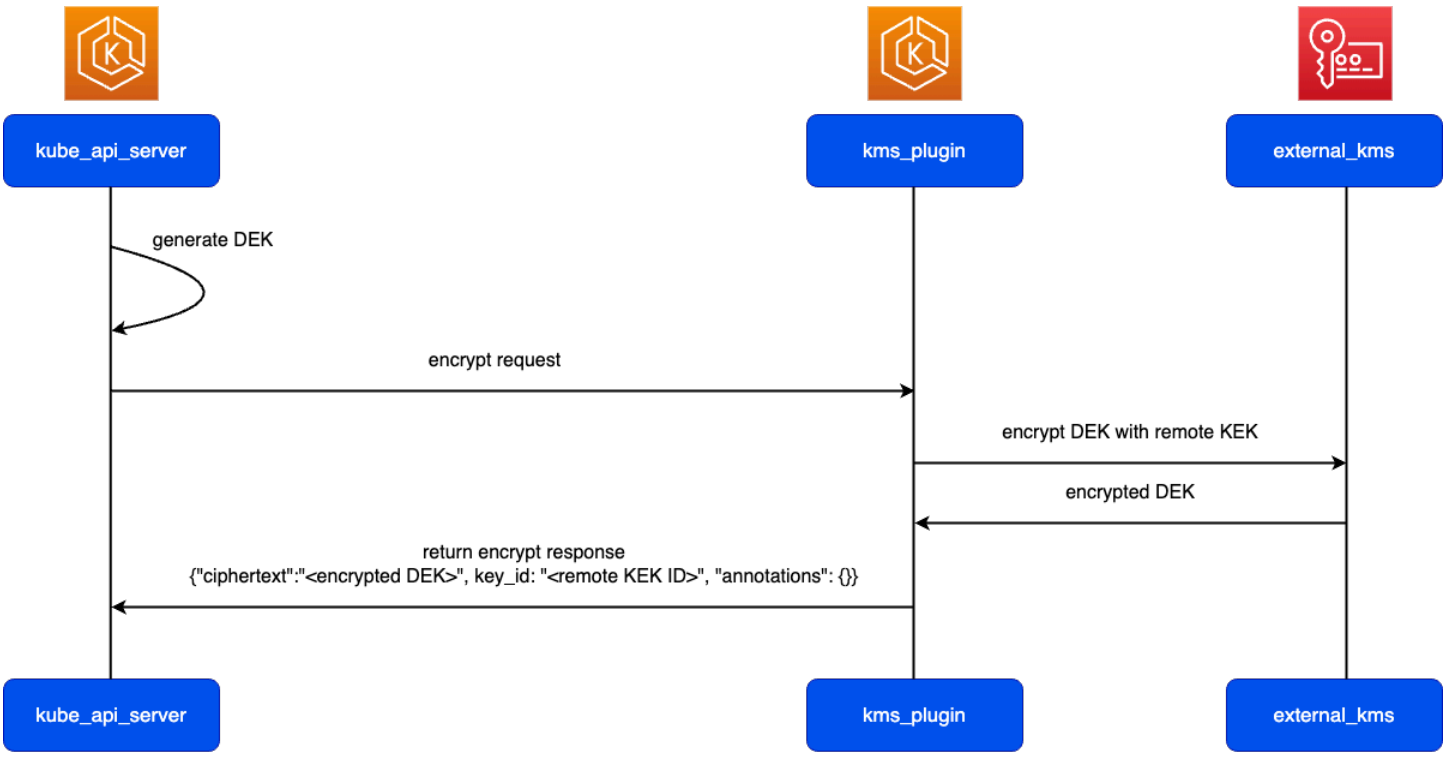
El cifrado de sobre consiste en cifrar los datos en texto plano mediante una clave de cifrado de datos (DEK) antes de enviarlos al almacén de datos (etcd), y luego cifrar la DEK con una clave raíz de KMS, la cual se almacena en un sistema KMS remoto y administrado de forma centralizada (AWS KMS). Esta es una estrategia de defensa en profundidad, ya que protege los datos mediante una clave de cifrado (DEK) y agrega una capa adicional de seguridad al proteger dicha DEK con una clave de cifrado independiente y almacenada de forma segura, conocida como clave de cifrado de clave (KEK).

Cómo Amazon EKS habilita el cifrado de sobre predeterminado con KMS v2 y AWS KMS

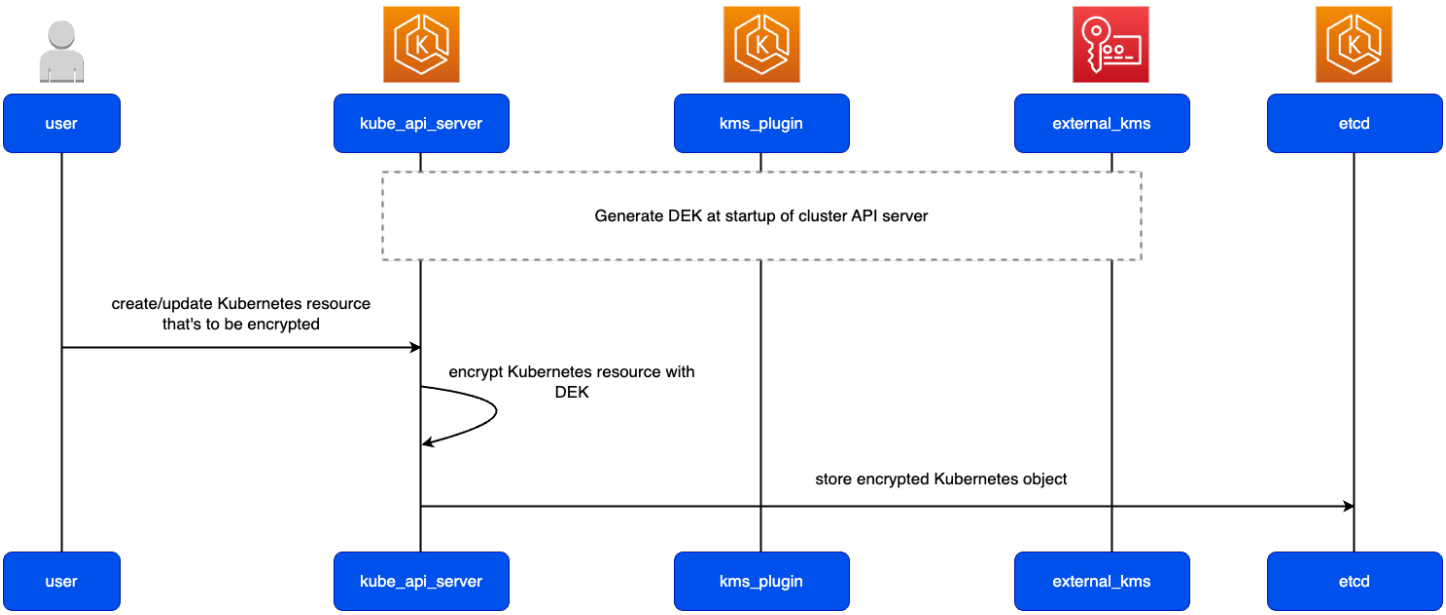
Amazon EKS utiliza KMS v2 para aplicar cifrado de sobre predeterminado a todos los datos de la API en el plano de control administrado de Kubernetes, antes de que se almacenen de forma persistente en la base de datos etcd. Al iniciar, el servidor de la API del clúster genera una clave de cifrado de datos (DEK) a partir de una semilla secreta combinada con datos generados aleatoriamente. Además, al inicio, el servidor de la API realiza una llamada al complemento de KMS para cifrar la semilla de la DEK mediante una clave de cifrado de clave (KEK) remota proporcionada por AWS KMS. Se trata de una llamada única que se ejecuta al inicio del servidor de la API y durante la rotación de la KEK. A continuación, el servidor de la API almacena en caché la semilla DEK cifrada. Después de esto, el servidor de la API utiliza la semilla de la DEK almacenada en caché para generar otras claves de cifrado de datos de un solo uso, basadas en una función de derivación de claves (KDF). Posteriormente, cada una de estas DEK generadas se utiliza una sola vez para cifrar un recurso individual de Kubernetes antes de que se almacene en etcd. El uso de una semilla de la DEK cifrada y almacenada en caché a través de KMS v2 permite que el cifrado de los recursos de Kubernetes en el servidor de la API sea más eficiente y rentable.

De forma predeterminada, esta KEK es administrada por AWS; no obstante, es posible utilizar una propia de AWS KMS.

El siguiente diagrama ilustra el proceso de generación y cifrado de una DEK durante el inicio del servidor de la API.



El siguiente diagrama de alto nivel ilustra el cifrado de un recurso de Kubernetes antes de su almacenamiento en etcd.



Preguntas frecuentes

¿Cómo el cifrado de sobre predeterminado mejora la postura de seguridad del clúster de EKS?

Esta característica reduce tanto la superficie de exposición como el tiempo durante el cual los metadatos y el contenido del cliente permanecen sin cifrado. Con el cifrado de sobre predeterminado, los metadatos y el contenido del cliente permanecen sin cifrado únicamente de forma temporal en la memoria del kube-apiserver, antes de almacenarse en etcd. La memoria del kube-apiserver cuenta con protección a través del sistema Nitro. Amazon EKS solo utiliza instancias de EC2 basadas en Nitro para el plano de control administrado de Kubernetes. Estas instancias incorporan mecanismos de control de seguridad diseñados para impedir que cualquier sistema o persona acceda a su memoria.

¿Qué versión de Kubernetes debo ejecutar para disponer de esta característica?

Para habilitar el cifrado de sobre de forma predeterminada, el clúster de Amazon EKS debe ejecutar Kubernetes en la versión 1.28 o superior.

¿Los datos permanecen seguros si se ejecuta una versión del clúster de Kubernetes que no admite esta característica?

Sí. En AWS, la seguridad es nuestra máxima prioridad. Fundamentamos nuestra transformación digital e innovación en las prácticas recomendadas operativas en términos de seguridad, con el firme compromiso de elevar ese estándar consistentemente.

Todos los datos almacenados en etcd están cifrados a nivel de disco para cada clúster de EKS, sin importar la versión de Kubernetes que se ejecuta. EKS utiliza claves raíz para generar claves de cifrado por volumen, las cuales son administradas por el servicio de EKS. Además, cada clúster de Amazon EKS se ejecuta dentro de una VPC aislada, mediante máquinas virtuales dedicadas al clúster. Gracias a esta arquitectura y a nuestras prácticas de seguridad operativa, Amazon EKS cumple con múltiples estándares y marcos de conformidad, incluidos SOC 1, 2 y 3, PCI-DSS, ISO y los requisitos de HIPAA. Estos estándares y marcos de cumplimiento se garantizan para todos los clústeres de EKS, independientemente de si utilizan el cifrado de sobre predeterminado.

¿Cómo funciona el cifrado de sobre en Amazon EKS?

Al iniciar, el servidor de la API del clúster genera una clave de cifrado de datos (DEK) a partir de una semilla secreta combinada con datos generados aleatoriamente. Asimismo, al iniciar, el servidor de la API llama al complemento de KMS para cifrar la clave de cifrado de datos (DEK) con una clave de cifrado remota (KEK) de AWS KMS. Se trata de una llamada única que se ejecuta al inicio del servidor de la API y durante la rotación de la KEK. A continuación, el servidor de la API almacena

en caché la semilla DEK cifrada. Después de esto, el servidor de la API utiliza la semilla de la DEK almacenada en caché para generar otras claves de cifrado de datos de un solo uso, basadas en una función de derivación de claves (KDF). Posteriormente, cada una de estas DEK generadas se utiliza una sola vez para cifrar un recurso individual de Kubernetes antes de que se almacene en etcd.

Es importante tener en cuenta que el servidor de la API realiza llamadas adicionales para verificar el estado y el correcto funcionamiento de la integración con AWS KMS. Estas comprobaciones de estado adicionales se pueden ver en AWS CloudTrail.

¿Debo realizar alguna acción o modificar permisos para que esta característica opere en el clúster de EKS?

No, no es necesario realizar ninguna acción. El cifrado de sobre en Amazon EKS es ahora una configuración predeterminada que se encuentra habilitada en todos los clústeres que ejecutan Kubernetes en la versión 1.28 o superior. La integración con AWS KMS se establece mediante el servidor de la API de Kubernetes administrado por AWS. Esto significa que no es necesario configurar permisos para comenzar a utilizar el cifrado con KMS en el clúster.

¿Cómo se puede saber si el cifrado de sobre predeterminado está habilitado en el clúster?

Si migra al uso de una CMK propia, podrá visualizar el ARN de la clave de KMS asociada al clúster. Además, puede consultar los registros de eventos de AWS CloudTrail relacionados con el uso de la CMK del clúster.

Si el clúster utiliza una clave propiedad de AWS, esta información se detalla en la consola de EKS (sin incluir el ARN de la clave).

¿Puede AWS acceder a la clave propiedad de AWS utilizada para el cifrado de sobre predeterminado en Amazon EKS?

No. AWS aplica controles de seguridad estrictos en Amazon EKS que impiden que cualquier persona acceda a claves de cifrado en texto plano utilizadas para proteger los datos en la base de datos de etcd. Estas medidas de seguridad también se aplican a la clave de KMS propiedad de AWS.

¿El cifrado de sobre predeterminado está habilitado en el clúster de EKS existente?

Si ejecuta un clúster de Amazon EKS con Kubernetes en la versión 1.28 o superior, el cifrado de sobre de todos los datos de la API de Kubernetes estará habilitado de forma predeterminada. Para los clústeres existentes, Amazon EKS utiliza el ClusterRole de RBAC denominado `eks:kms-storage-migrator` para migrar los datos que anteriormente no estaban cifrados mediante cifrado de sobre en etcd hacia este nuevo estado de cifrado.

¿Qué significa esto si ya habilité el cifrado de sobre para secretos en el clúster de EKS?

Si ya se cuenta con una clave administrada por el cliente (CMK) en KMS que se utilizó para cifrar mediante cifrado de sobre los secretos de Kubernetes, esa misma clave se usará como clave de cifrado (KEK) para el cifrado de sobre de todos los tipos de datos de la API de Kubernetes en el clúster.

¿Existe algún costo adicional por ejecutar un clúster de EKS con el cifrado de sobre predeterminado?

No hay ningún costo adicional asociado al plano de control administrado de Kubernetes si se utiliza una clave propiedad de Amazon Web Services para el cifrado de sobre predeterminado. De forma predeterminada, todo clúster de EKS que ejecute Kubernetes en la versión 1.28 o posterior utiliza una clave propiedad de Amazon Web Services. Sin embargo, si se utiliza una clave propia de AWS KMS, se aplicarán los precios estándar de KMS.

¿Cuánto cuesta utilizar una clave propia de AWS KMS para cifrar los datos de la API de Kubernetes en el clúster?

Se paga 1 USD al mes por almacenar cualquier clave personalizada que se cree o se importe a KMS. KMS cobra por las solicitudes de cifrado y descifrado. Existe un nivel gratuito de 20 000 solicitudes por mes y por cuenta, y se paga 0,03 USD por cada 10 000 solicitudes adicionales al mes que superen ese límite. Esto se aplica a todo el uso de KMS dentro de una cuenta, por lo que el costo de utilizar una clave propia de AWS KMS en el clúster se verá afectado por el uso de esa misma clave en otros clústeres o recursos de AWS dentro de la cuenta.

¿Aumentarán los cargos de KMS ahora que la clave administrada por el cliente (CMK) se utiliza para cifrar mediante cifrado de sobre todos los datos de la API de Kubernetes y no solo los secretos?

No. Nuestra implementación con KMS v2 reduce significativamente la cantidad de llamadas realizadas a AWS KMS. Esto, a su vez, reducirá los costos relacionados con la CMK, independientemente de los datos adicionales de Kubernetes que se cifren o descifren en el clúster de EKS.

Como se detalla arriba, la semilla de la DEK generada, utilizada para el cifrado de los recursos de Kubernetes, se almacena localmente en la caché del servidor de la API de Kubernetes después de haber sido cifrada con la KEK remota. Si la semilla de la DEK cifrada no se encuentra en la caché del servidor de la API, el servidor de la API realizará una llamada a AWS KMS para cifrar la semilla de DEK. El servidor de la API luego almacena en caché la semilla de la DEK cifrada para su uso futuro en el clúster sin realizar más llamadas a KMS. De manera similar, para las solicitudes de descifrado, el servidor de la API realizará una llamada a AWS KMS para la primera solicitud de

descifrado, después de lo cual la semilla de la DEK descifrada se almacenará en caché y se utilizará para futuras operaciones de descifrado.

Para obtener más información, consulte KEP-3299: KMS v2 Improvements en las Mejoras de Kubernetes en GitHub.

¿Puedo utilizar la misma clave de CMK para varios clústeres de Amazon EKS?

Sí. Para utilizar una clave nuevamente, puede vincularla a un clúster en la misma región al asociar el ARN con el clúster durante su creación. Sin embargo, si utiliza la misma clave de CMK para varios clústeres de EKS, debe tomar las medidas necesarias para evitar la desactivación arbitraria de la clave de CMK. De lo contrario, una clave de CMK desactivada asociada a varios clústeres de EKS tendrá un impacto más amplio en los clústeres que dependan de esa clave.

¿Qué ocurre con un clúster de EKS si la CMK queda inaccesible después de habilitar el cifrado de sobre predeterminado?

Si desactiva una clave KMS, no se podrá utilizar en ninguna operación criptográfica. Sin acceso a una CMK existente, el servidor de la API no podrá cifrar ni almacenar de forma persistente objetos de Kubernetes recién creados, ni descifrar objetos de Kubernetes previamente cifrados almacenados en etcd. Si la CMK está desactivada, el clúster se colocará inmediatamente en mal estado o estado degradado, momento en el cual no podremos cumplir con nuestro compromiso de servicio hasta que reactivé la CMK asociada.

Cuando una CMK esté desactivada, recibirá notificaciones sobre el estado degradado del clúster de EKS y la necesidad de volver a habilitar la CMK dentro de los 30 días posteriores a su desactivación para garantizar la correcta restauración de los recursos del plano de control de Kubernetes.

¿Cómo puedo proteger un clúster de EKS del impacto de una CMK desactivada o eliminada?

Para proteger los clústeres de EKS de una situación como esta, los administradores de claves deben administrar el acceso a las operaciones de claves de KMS mediante políticas de IAM basadas en el principio de privilegio mínimo, con el fin de reducir el riesgo de desactivación o eliminación arbitraria de las claves asociadas con los clústeres de EKS. Además, puede configurar una alarma de CloudWatch para recibir notificaciones sobre el estado de la CMK.

¿Se restaurará el clúster de EKS si vuelvo a habilitar la CMK?

Para garantizar la restauración correcta de un clúster de EKS, se recomienda encarecidamente volver a habilitar la CMK dentro de los primeros 30 días de su desactivación. Sin embargo, la restauración exitosa de un clúster de EKS también dependerá de si se producen cambios

incompatibles en la API como resultado de una actualización automática de Kubernetes que pueda tener lugar mientras el clúster se encuentre en mal estado o en estado degradado.

¿Por qué un clúster de EKS queda en mal estado o en estado degradado después de desactivar la CMK?

El servidor de la API del plano de control de EKS utiliza una clave DEK cifrada y almacenada en caché en la memoria del servidor de la API para cifrar todos los objetos durante las operaciones de creación o actualización, antes de almacenarlos en etcd. Al recuperar un objeto existente desde etcd, el servidor de la API utiliza la misma clave DEK almacenada en caché y descifra el objeto del recurso de Kubernetes. Si se desactiva la CMK, el servidor de la API no experimentará un impacto inmediato debido a la clave DEK almacenada en caché en la memoria del servidor de la API. Sin embargo, al reiniciar la instancia del servidor de la API, no contará con una clave DEK almacenada en caché y deberá llamar a AWS KMS para realizar las operaciones de cifrado y descifrado. Sin una CMK, este proceso fallará con el código de error KMS_KEY_DISABLED, lo que impedirá que el servidor de la API inicie correctamente.

¿Qué ocurre con un clúster de EKS si se elimina la CMK?

Eliminar la clave de CMK asociada con un clúster de EKS degradará su estado de forma irreversible. Sin la CMK del clúster, el servidor de la API ya no podrá cifrar ni almacenar de forma persistente nuevos objetos de Kubernetes, ni descifrar los objetos previamente cifrados almacenados en la base de datos etcd. Debe proceder con la eliminación de una clave de CMK de un clúster de EKS solo cuando tenga la certeza de que ya no se necesita utilizar ese clúster.

Tenga en cuenta que, si no se encuentra la CMK (KMS_KEY_NOT_FOUND) o si se revocan las concesiones de la CMK asociada al clúster (KMS_GRANT_REVOKED), el clúster no se podrá recuperar. Para obtener más información sobre el estado del clúster y los códigos de error, consulte las Preguntas frecuentes sobre el estado del clúster y los códigos de error con sus rutas de resolución.

¿Aún se aplicarán cargos por un clúster de EKS en mal estado o degradado debido a la desactivación o eliminación de la CMK?

Sí. Aunque el plano de control de EKS no se podrá utilizar en caso de desactivación de la CMK, AWS seguirá ejecutando los recursos de infraestructura dedicados asignados al clúster de EKS hasta que el cliente lo elimine. Además, nuestro compromiso de servicio no se aplicará en tal circunstancia, ya que se tratará de una acción u omisión voluntaria del cliente la que afecta el estado y funcionamiento normal del clúster de EKS.

¿Se puede actualizar un clúster de EKS cuando se encuentra en mal estado o en un estado degradado debido a una CMK desactivada?

Sí. Sin embargo, si el clúster tiene una CMK desactivada, tendrá un periodo de 30 días para volver a activarla. Durante este período de 30 días, el clúster de Kubernetes no se actualizará automáticamente. No obstante, si transcurre el período sin que se vuelva a habilitar la CMK, el clúster se actualizará automáticamente a la siguiente versión (n+1) con soporte estándar, de acuerdo con el ciclo de vida de versiones de Kubernetes en EKS.

Recomendamos encarecidamente volver a habilitar lo antes posible una CMK desactivada al identificar un clúster afectado. Es importante tener en cuenta que, aunque EKS actualizará automáticamente los clústeres afectados, no se garantiza que estos se recuperen correctamente, especialmente si atraviesan múltiples actualizaciones automáticas, ya que esto puede implicar cambios en la API de Kubernetes y generar comportamientos inesperados en el proceso de arranque del servidor de la API.

¿Se puede usar un alias de clave de KMS?

Sí. Amazon EKS admite el uso de alias de claves de KMS. Un alias es un nombre descriptivo asignado a una clave de KMS de Amazon Web Services. Por ejemplo, un alias permite hacer referencia a una clave de KMS como my-key en lugar de usar **`1234abcd-12ab-34cd-56ef-1234567890ab`**.

¿Es posible seguir con la copia de seguridad y restauración de los recursos del clúster mediante una solución propia de copia de seguridad para Kubernetes?

Sí. Se puede utilizar una solución de copia de seguridad para Kubernetes (como Velero) con fines de recuperación ante desastres, migración de datos y protección de datos. Si utiliza una solución de copia de seguridad para Kubernetes que accede a los recursos del clúster a través del servidor de la API, los datos que recupere la aplicación se descifrarán antes de llegar al cliente. Esto permite recuperar los recursos del clúster en otro clúster de Kubernetes.

Consideraciones de seguridad para el modo automático de Amazon EKS

En este tema se describe la arquitectura de seguridad, los controles y las prácticas recomendadas para el modo automático de Amazon EKS. A medida que las organizaciones implementan aplicaciones en contenedores a escala, resulta cada vez más complejo mantener una postura de seguridad sólida. El modo automático de EKS implementa controles de seguridad automatizados

y se integra con servicios de seguridad de AWS para ayudar a proteger la infraestructura del clúster, las cargas de trabajo y los datos. Gracias a las características de seguridad integradas, como la administración forzada del ciclo de vida de los nodos y la implementación automatizada de revisiones, el modo automático de EKS ayuda a seguir las prácticas recomendadas de seguridad a la vez que reduce la sobrecarga operativa.

Antes de continuar con este tema, asegúrese de que conoce los conceptos básicos del modo automático de EKS y de que ha revisado los requisitos previos para habilitar el modo automático de EKS en los clústeres. Para obtener información general sobre la seguridad de Amazon EKS, consulte Seguridad.

El modo automático de Amazon EKS se basa en los fundamentos de seguridad existentes de Amazon EKS al tiempo que introduce controles de seguridad automatizados adicionales para las instancias administradas de EC2.

Seguridad y autenticación de la API

El modo automático de Amazon EKS utiliza mecanismos de seguridad de la plataforma de AWS para proteger y autenticar las llamadas a la API de Amazon EKS.

- El acceso a la API de Kubernetes está protegido mediante entradas de acceso de EKS, que se integran con las identidades de AWS IAM.
  - Para obtener más información, consulte the section called “Entradas de los registros”.
- Los clientes pueden implementar un control de acceso detallado al punto de conexión de la API de Kubernetes mediante la configuración de entradas de acceso de EKS.

Seguridad de la red

El modo automático de Amazon EKS admite varias capas de seguridad de red:

- Integración de la VPC
  - Funciona dentro de la Amazon Virtual Private Cloud (VPC)
  - Admite configuraciones de VPC y diseños de subred personalizados
  - Habilita las redes privadas entre los componentes del clúster
  - Para obtener más información, consulte Administración de las responsabilidades de seguridad para Amazon Virtual Private Cloud
- Políticas de red

- Compatibilidad nativa con las políticas de red de Kubernetes
- Capacidad para definir reglas detalladas de tráfico de red
- Para obtener más información, consulte the section called “Políticas de Kubernetes”

Seguridad de las instancias administradas por EC2

El modo automático de Amazon EKS opera instancias administradas de EC2 con los siguientes controles de seguridad:

Seguridad de EC2

- Las instancias administradas por EC2 mantienen las características de seguridad de Amazon EC2.
- Para obtener más información sobre las instancias administradas por EC2, consulte Seguridad en Amazon EC2.

Administración del ciclo de vida de la instancia

Las instancias administradas de EC2 que funcionan con el modo automático de EKS tienen una vida útil máxima de 21 días. El modo automático de Amazon EKS termina automáticamente las instancias que sobrepasan esta vida útil. Este límite del ciclo de vida ayuda a evitar desviaciones en la configuración y mantiene la postura de seguridad.

Protección de los datos

- El almacenamiento de instancias de Amazon EC2 está cifrado. Se trata de almacenamiento directamente conectado a la instancia. Para obtener más información, consulte Protección de datos en Amazon EC2.
- El modo automático de EKS administra los volúmenes asociados a las instancias de EC2 en el momento de la creación, incluidos los volúmenes raíz y de datos. El modo automático de EKS no administra completamente los volúmenes de EBS creados mediante las características de almacenamiento persistente de Kubernetes.

Administración de parches

- El modo automático de Amazon EKS aplica revisiones automáticamente a las instancias administradas.

- Dentro de las revisiones se incluyen:
  - Actualizaciones del sistema operativo
  - Parches de seguridad
  - Componentes del modo automático de Amazon EKS

> **Note**
>
> Los clientes mantienen la responsabilidad de proteger y actualizar las cargas de trabajo que se ejecutan en estas instancias.

Controles de acceso

- El acceso directo a la instancia está restringido:
  - El acceso SSH no está disponible.
  - El acceso al Administrador de sesiones de AWS Systems Manager (SSM) no está disponible.
- Las operaciones de administración se realizan a través de la API de Amazon EKS y la API de Kubernetes.

Administración automatizada de los recursos

El modo automático de Amazon EKS no administra completamente los volúmenes de Amazon Elastic Block Store (Amazon EBS) creados mediante las características de almacenamiento persistente de Kubernetes. El modo automático de EKS tampoco administra los equilibradores de carga elásticos (ELB). El modo automático de Amazon EKS automatiza las tareas rutinarias de estos recursos.

Seguridad de almacenamiento

- AWS recomienda habilitar el cifrado para los volúmenes de EBS aprovisionados por las características de almacenamiento persistente de Kubernetes. Para obtener más información, consulte the section called “Creación de una StorageClass”.
- Cifrado en reposo mediante AWS KMS

- Puede configurar su cuenta de AWS para aplicar el cifrado de los nuevos volúmenes de EBS y copias de instantáneas que cree. Para obtener más información, consulte Cómo habilitar el cifrado de Amazon EBS de forma predeterminada en la Guía del usuario de Amazon EBS.
- Para obtener más información, consulte Seguridad en Amazon EBS.

Seguridad del equilibrador de carga

- Configuración automatizada de equilibradores de carga elásticos
- Administración de certificados SSL/TLS mediante la integración de AWS Certificate Manager
- Automatización de grupos de seguridad para el control de acceso al equilibrador de carga
- Para obtener más información, consulte Seguridad en Elastic Load Balancing.

Prácticas recomendadas de seguridad

La siguiente sección describe las prácticas recomendadas de seguridad aplicables al modo automático de Amazon EKS.

- Revise periódicamente las políticas de AWS IAM y las entradas de acceso de EKS.
- Implemente patrones de acceso de privilegio mínimo para las cargas de trabajo.
- Supervise la actividad del clúster a través de AWS CloudTrail y Amazon CloudWatch. Para obtener más información, consulte the section called “ AWS CloudTrail” y the section called “Amazon CloudWatch”.
- Utilice AWS Security Hub para evaluar la postura de seguridad.
- Aplique los estándares de seguridad de pod apropiados para las cargas de trabajo.

Consideraciones sobre la seguridad para las capacidades de EKS

En este tema, se tratan aspectos de seguridad importantes para las capacidades de EKS, como la configuración de roles de IAM, los permisos de Kubernetes y los patrones de arquitectura para las implementaciones de varios clústeres y la administración de recursos de AWS entre cuentas.

Las capacidades de EKS utilizan una combinación de roles de IAM, entradas de acceso de EKS y RBAC de Kubernetes para proporcionar un acceso seguro a los servicios de AWS, los recursos de Kubernetes en el clúster y las integraciones con AWS CodeConnections, AWS Secrets Manager y otros servicios de AWS.

Rol de IAM de capacidad

Al crear una capacidad, proporcionará un rol de capacidad de IAM que EKS utiliza para llevar a cabo acciones en su nombre. Este rol debe cumplir con lo siguiente:

- Debe estar en la misma cuenta de AWS que el clúster y el recurso de capacidad.
- Debe tener una política de confianza que permita a la entidad principal del servicio de `capabilities.eks.amazonaws.com` asumir el rol.
- Debe tener los permisos de IAM adecuados para el tipo de capacidad y el caso de uso, en función de sus requisitos. Para obtener información detallada sobre los permisos de IAM necesarios, consulte the section called “ AWS CodeConnections”, the section called “ AWS Secrets Manager” y the section called “Configuración de permisos”.

Una práctica recomendada es tener en cuenta el ámbito de los privilegios necesarios para su caso de uso específico y otorgar solo los permisos necesarios para cumplir sus requisitos. Por ejemplo, al utilizar la capacidad de EKS para Kube Resource Orchestrator, es posible que no se requieran permisos de IAM, mientras que, si se utiliza la capacidad de EKS para Controladores de AWS para Kubernetes, puede otorgar acceso completo a uno o más servicios de AWS.

> **Important**
>
> Si bien algunos casos de uso pueden justificar el uso de amplios privilegios administrativos, siga el principio de privilegio mínimo y otorgue solo los permisos de IAM mínimos necesarios para su caso de uso específico y restrinja el acceso a recursos específicos mediante ARN y claves de condición en lugar de utilizar permisos comodín.

Para obtener más información sobre cómo crear y configurar roles de IAM de capacidad, consulte the section called “Rol de IAM de capacidad”.

Entradas de acceso de EKS

Al crear una capacidad con un rol de IAM, Amazon EKS crea automáticamente una entrada de acceso para ese rol en el clúster. Esta entrada de acceso otorga a la capacidad los permisos básicos de Kubernetes para funcionar.

> **Note**
>
> Las entradas de acceso se crean para el clúster en el que se crea la capacidad. Para las implementaciones de Argo CD en clústeres remotos, debe crear entradas de acceso en esos clústeres con los permisos adecuados para que Argo CD pueda implementar y administrar aplicaciones.

La entrada de acceso incluye lo siguiente:

- El rol de IAM (ARN) como entidad principal
- Políticas de entrada de acceso específicas por capacidad que otorgan permisos de línea de base de Kubernetes
- Ámbito adecuado (en todo el clúster o en el ámbito del espacio de nombres) en función del tipo de capacidad

> **Note**
>
> En el caso de Argo CD, los permisos relacionados con el espacio de nombres se otorgan al espacio de nombres especificado en la configuración de la capacidad (el valor predeterminado es `argocd`).

Políticas de entrada de acceso predeterminadas por capacidad

Cada tipo de capacidad otorga al rol de capacidad los permisos necesarios y establece diferentes políticas de entrada de acceso predeterminadas, de la siguiente manera:

kro

- `arn:aws:eks::aws:cluster-access-policy/AmazonEKSKROPolicy` (en el ámbito del clúster)

  Otorga permisos para ver y administrar ResourceGraphDefinitions y crear instancias de recursos personalizados definidos por las RGD.

ACK

- `arn:aws:eks::aws:cluster-access-policy/AmazonEKSACKPolicy` (en el ámbito del clúster)

Otorga permisos para crear, leer, actualizar y eliminar recursos personalizados de ACK en todos los espacios de nombres.

Argo CD

- `arn:aws:eks::aws:cluster-access-policy/AmazonEKSArgoCDClusterPolicy` (en el ámbito del clúster)

  Otorga permisos de clúster para que Argo CD pueda detectar recursos y administrar objetos del ámbito del clúster.

- `arn:aws:eks::aws:cluster-access-policy/AmazonEKSArgoCDPolicy` (en el ámbito del espacio de nombres)

  Otorga permisos de espacio de nombres para que Argo CD pueda implementar y administrar aplicaciones. Su ámbito es el espacio de nombres especificado en la configuración de la capacidad (el valor predeterminado es `argocd`).

Consulte the section called “Revisión de las políticas de acceso” para obtener información más detallada.

Permisos de Kubernetes adicionales

Algunas capacidades pueden requerir permisos de Kubernetes adicionales además de las políticas de acceso y entrada predeterminadas. Puede otorgar estos permisos de una de las siguientes maneras:

- Políticas de entrada de acceso: asocie políticas administradas adicionales a la entrada de acceso.
- RBAC de Kubernetes: cree enlaces de `Role` o `ClusterRole` para el usuario de Kubernetes de la capacidad.

Permisos de lector de secretos de ACK

Algunos controladores de ACK necesitan leer secretos de Kubernetes para recuperar información confidencial, como las contraseñas de las bases de datos. Los siguientes controladores de ACK requieren un acceso de lectura de secretos:

- `acm`, `acmpca`, `documentdb`, `memorydb`, `mq`, `rds`, `secretsmanager`

Haga lo siguiente para conceder permisos de lectura de secretos:

1. Asocie la política de entrada de acceso `arn:aws:eks::aws:cluster-access-policy/AmazonEKSSecretReaderPolicy` a la entrada de acceso de la capacidad
2. Limite la política a espacios de nombres específicos en los que los recursos de ACK hagan referencia a secretos u otorguen acceso a todo el clúster

> **Important**
>
> Los permisos de lectura de secretos se limitan a los espacios de nombres que especifique al asociar la política de entrada de acceso. Esto le permite limitar los secretos a los que puede acceder la capacidad.

Permisos de recursos arbitrarios de kro

kro necesita permisos para crear y administrar los recursos definidos en sus ResourceGraphDefinitions. De forma predeterminada, kro solo puede ver y administrar las RGD por sí mismo.

Puede hacer lo siguiente para otorgar permisos de creación de recursos a kro:

Opción 1: políticas de entrada de acceso

Asocie políticas de entrada de acceso predefinidas, como `AmazonEKSAdminPolicy` o `AmazonEKSEditPolicy`, a la entrada de acceso de la capacidad.

Opción 2: RBAC de Kubernetes

Cree un `ClusterRoleBinding` que otorgue los permisos necesarios al usuario de Kubernetes de la capacidad:

```
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**
>
> El nombre de usuario de Kubernetes para kro sigue este patrón: `arn:aws:sts::ACCOUNT_ID:assumed-role/ROLE_NAME/kro`.
> La capacidad de kro de EKS establece automáticamente el nombre de la sesión `/kro`.

Permisos de IAM que necesita la capacidad

kro (Kube Resource Orchestrator)

No se necesita ningún permiso de IAM. Puede crear un rol de capacidad sin políticas adjuntas. kro solo necesita los permisos de RBAC de Kubernetes.

ACK (Controladores de AWS para Kubernetes)

Necesita permisos para administrar los recursos de AWS que ACK creará y administrará. Debe limitar los permisos a servicios, acciones y recursos específicos en función de sus requisitos. Para obtener información detallada sobre la configuración de los permisos de ACK, lo que incluye las prácticas recomendadas de producción con selectores de roles de IAM, consulte the section called “Configuración de permisos”.

Argo CD

No se necesita ningún permiso de IAM de forma predeterminada. Es posible que se necesiten permisos opcionales para:

- AWS Secrets Manager: si se almacenan las credenciales del repositorio de Git en Secrets Manager
- AWS CodeConnections: si utiliza CodeConnections para la autenticación del repositorio de Git
- Amazon ECR: si utiliza gráficos de Helm almacenados en formato OCI en Amazon ECR

Prácticas recomendadas de seguridad

Privilegio mínimo de IAM

Otorgue a sus recursos de capacidad solo los permisos necesarios para su caso de uso. Esto no significa que no pueda conceder amplios permisos administrativos a las capacidades si es necesario. En esos casos, debe gobernar el acceso a esos recursos de manera adecuada.

Roles de capacidad:

- ACK: siempre que sea posible, limite los permisos de IAM a recursos y servicios de AWS específicos que necesiten sus equipos, según el caso de uso y los requisitos.
- Argo CD: restrinja el acceso a repositorios de Git y espacios de nombres de Kubernetes específicos.
- kro: necesita un rol de capacidad para la política de confianza, pero no se necesitan permisos de IAM (solo usa RBAC del clúster).

Ejemplo: en lugar de `"Resource": "*"`, especifique patrones para recursos o grupos de recursos específicos.

```
"Resource": [
  "arn:aws:s3:::my-app-*",
  "arn:aws:rds:us-west-2:111122223333:db:prod-*"
]
```

Utilice claves de condición de IAM para restringir aún más el acceso:

```
"Condition": {
  "StringEquals": {
    "aws:ResourceTag/Environment": "production"
  }
}
```

Para obtener información adicional sobre la configuración de IAM, consulte la sección de consideraciones de cada capacidad.

Aislamiento de espacios de nombres para los secretos de Argo CD

La capacidad de Argo CD administrada tiene acceso a todos los secretos de Kubernetes dentro del espacio de nombres configurado (el valor predeterminado es `argocd`). Para mantener una posición de seguridad óptima, siga estas prácticas de aislamiento de espacios de nombres:

- Guarde solo los secretos pertinentes para Argo CD en el espacio de nombres de Argo CD.
- Evite almacenar secretos de aplicaciones no relacionados en el mismo espacio de nombres que Argo CD.
- Utilice espacios de nombres separados para los secretos de aplicaciones que no sean necesarios para las operaciones de Argo CD.

Este aislamiento garantiza que el acceso a los secretos de Argo CD se limite solo a las credenciales que necesita para la autenticación del repositorio de Git y otras operaciones específicas de Argo CD.

RBAC de Kubernetes

Controle qué usuarios y cuentas de servicio pueden crear y administrar recursos de capacidad. Una práctica recomendada consiste en implementar recursos de capacidad en espacios de nombres dedicados con políticas de RBAC adecuadas.

Ejemplo: el rol de RBAC funciona con ACK, lo que permite administrar los recursos del bucket de S3 en el espacio de nombres `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"]
```

Registro de auditoría

CloudTrail: todas las operaciones de la API de la capacidad de EKS (crear, actualizar, eliminar) se registran en AWS CloudTrail.

Active el registro de CloudTrail para hacer un seguimiento de lo siguiente:

- Quién creó o modificó las capacidades
- Cuándo se cambiaron las configuraciones de las capacidades
- Qué roles de capacidad se utilizan

Acceso a la red y puntos de conexión de VPC

Acceso privado a la API de Argo CD

Para restringir el acceso al servidor de la API de Argo CD, puede asociar uno o más puntos de conexión de VPC con el punto de conexión de Argo CD alojado. Esto permite la conectividad privada con la interfaz de usuario y la API de Argo CD desde la VPC sin tener que atravesar la Internet pública.

> **Note**
>
> Los puntos de conexión de VPC conectados a los puntos de conexión de la API de Argo CD alojados eks-capabilities.*region*.amazonaws.com) no admiten políticas de puntos de conexión de VPC.

Implementación en clústeres privados

La capacidad de Argo CD permite implementar aplicaciones en clústeres de EKS completamente privados, lo que proporciona una ventaja operativa significativa al eliminar la necesidad de emparejamiento de VPC o configuraciones de red complejas. Sin embargo, al diseñar esta arquitectura, tenga en cuenta que Argo CD extraerá la configuración de los repositorios de Git (que pueden ser públicos) y la aplicará en clústeres privados.

Asegúrese de lo siguiente:

- Utilice repositorios de Git privados para cargas de trabajo confidenciales.
- Implemente los controles de acceso y la autenticación adecuados al repositorio de Git.
- Revise y apruebe los cambios mediante solicitudes de extracción antes de fusionarlos.
- Considere la posibilidad de usar los plazos de sincronización de Argo CD para controlar cuándo se pueden producir las implementaciones.
- Supervise los registros de auditoría de Argo CD para detectar cambios no autorizados en la configuración.

Conformidad

Las capacidades de EKS están completamente administradas y cuentan con las certificaciones de cumplimiento de Amazon EKS.

Para obtener información actualizada sobre el cumplimiento, consulte Servicios de AWS en el ámbito del programa de conformidad.

Siguientes pasos

- the section called “Configuración de permisos”: configuración de los permisos de IAM para ACK
- the section called “Configuración de permisos”: configuración del RBAC de Kubernetes para kro
- the section called “Configuración de permisos”: configuración de la integración de Identity Center para Argo CD
- the section called “Solución de problemas de capacidades”: solución de problemas de seguridad y permisos

Administración de identidades y accesos para Amazon EKS

AWS Identity and Access Management (IAM) es un servicio de AWS que ayuda a los administradores a controlar de forma segura el acceso a los recursos de AWS. Los administradores de IAM controlan quién está autenticado (ha iniciado sesión) y autorizado (tiene permisos) para utilizar recursos de Amazon EKS. IAM es un servicio de AWS que puede utilizar sin cargo adicional.

Público

El modo de utilizar AWS Identity and Access Management (IAM) varía en función del trabajo que se realice en Amazon EKS.

Usuario de servicio: si utiliza el servicio Amazon EKS para realizar su trabajo, su administrador proporciona las credenciales y los permisos que necesita. A medida que utilice más características de Amazon EKS para realizar su trabajo, es posible que necesite permisos adicionales. Entender cómo se administra el acceso puede ayudarle a solicitar los permisos correctos al administrador. Si no puede acceder a una característica de Amazon EKS, consulte the section called “Solución de problemas”.

Administrador de servicio: si está a cargo de los recursos de Amazon EKS de su empresa, probablemente tenga acceso completo a Amazon EKS. Su trabajo consiste en determinar a qué

características y recursos de Amazon EKS deben acceder los usuarios del servicio. Luego, debe enviar solicitudes a su administrador de IAM para cambiar los permisos de los usuarios de su servicio. Revise la información de esta página para conocer los conceptos básicos de IAM. Para obtener más información sobre cómo la empresa puede utilizar IAM con Amazon EKS, consulte the section called “Amazon EKS e IAM”.

Administrador de IAM: si es un administrador de IAM, es posible que desee obtener información sobre cómo escribir políticas para administrar el acceso a Amazon EKS. Para consultar ejemplos de políticas de Amazon EKS basadas en identidades que puede utilizar en IAM, consulte the section called “Políticas basadas en identidades”.

Autenticación con identidades

La autenticación es la manera de iniciar sesión en AWS mediante credenciales de identidad. Debe estar autenticado (haber iniciado sesión en AWS) como usuario raíz de la cuenta de AWS, como un usuario de IAM o asumiendo un rol de IAM.

Puede iniciar sesión en AWS como una identidad federada mediante las credenciales proporcionadas a través de una fuente de identidad. AWS Los usuarios del Centro de identidades de IAM, la autenticación de inicio de sesión único de su empresa y sus credenciales de Google o Facebook son ejemplos de identidades federadas. Al iniciar sesión como una identidad federada, su administrador habrá configurado previamente la federación de identidades mediante roles de IAM. Cuando accede a AWS mediante la federación, está asumiendo un rol de forma indirecta.

Según el tipo de usuario que sea, puede iniciar sesión en Consola de administración de AWS o en el portal de acceso AWS. Para obtener más información sobre el inicio de sesión en AWS, consulte Cómo iniciar sesión en su cuenta de AWS en la Guía del usuario de inicio de sesión de AWS.

Si accede a AWS mediante programación, AWS proporciona un kit de desarrollo de software (SDK) y una interfaz de la línea de comandos (CLI) para firmar criptográficamente las solicitudes mediante el uso de las credenciales. Si no usa las herramientas de AWS, debe firmar las solicitudes. Para obtener más información sobre la firma de solicitudes, consulte Firma de solicitudes API de AWS en la Guía del usuario de IAM.

Independientemente del método de autenticación que use, es posible que deba proporcionar información de seguridad adicional. Por ejemplo, AWS le recomienda el uso de la autenticación multifactor (MFA) para aumentar la seguridad de su cuenta. Para obtener más información, consulte Autenticación multifactor en la Guía del usuario de AWS IAM Identity Center y Uso de la autenticación multifactor (MFA) en AWS en la Guía del usuario de IAM.

Usuario raíz de la cuenta de AWS

Cuando se crea una cuenta de AWS, se comienza con una identidad de inicio de sesión que tiene acceso completo a todos los servicios y recursos de AWS de la cuenta. Esta identidad recibe el nombre de usuario raíz de la cuenta de AWS y se accede a ella iniciando sesión con la dirección de email y la contraseña que utilizó para crear la cuenta. Recomendamos encarecidamente que no utilice el usuario raíz para sus tareas diarias. Proteja las credenciales del Usuario raíz y utilícelas solo para las tareas que solo el Usuario raíz pueda realizar. Para ver la lista completa de las tareas que requieren que inicie sesión como usuario raíz, consulte Tareas que requieren credenciales de usuario raíz en la Guía del usuario de IAM.

Usuarios y grupos de IAM

Un usuario de IAM es una identidad dentro de su cuenta de AWS que dispone de permisos específicos para una sola persona o aplicación. Siempre que sea posible, recomendamos emplear credenciales temporales, en lugar de crear usuarios de IAM que tengan credenciales de larga duración como contraseñas y claves de acceso. No obstante, si tiene casos de uso específicos que requieran credenciales de larga duración con usuarios de IAM, recomendamos rotar las claves de acceso. Para más información, consulte Rotar las claves de acceso periódicamente para casos de uso que requieran credenciales de larga duración en la Guía del usuario de IAM.

Un grupo de IAM es una identidad que especifica un conjunto de usuarios de IAM. No puede iniciar sesión como grupo. Puede usar grupos para especificar permisos para varios usuarios a la vez. Los grupos facilitan la administración de los permisos de grandes conjuntos de usuarios. Por ejemplo, podría tener un grupo cuyo nombre fuese IAMAdmins y conceder permisos a dicho grupo para administrar los recursos de IAM.

Los usuarios son diferentes de los roles. Un usuario se asocia exclusivamente a una persona o aplicación, pero la intención es que cualquier usuario pueda asumir un rol que necesite. Los usuarios tienen credenciales permanentes a largo plazo y los roles proporcionan credenciales temporales. Para más información, consulte Cuándo crear un usuario de IAM (en lugar de un rol) en la Guía del usuario de IAM.

IAM roles

Un rol de IAM es una identidad dentro de su cuenta de AWS que dispone de permisos específicos. Es similar a un usuario de IAM, pero no está asociado a una determinada persona. Puede asumir temporalmente un rol de IAM en la Consola de administración de AWS cambiando de roles. Puede asumir un rol realizando una llamada a una operación de la CLI de AWS o de la API de AWS, o bien

utilizando una URL personalizada. Para más información sobre los métodos para el uso de roles, consulte Uso de roles de IAM en la Guía del usuario de IAM.

Los roles de IAM con credenciales temporales son útiles en las siguientes situaciones:

- Acceso de usuario federado: para asignar permisos a una identidad federada, puede crear un rol y definir sus permisos. Cuando se autentica una identidad federada, se asocia la identidad al rol y se le conceden los permisos define el rol. Para obtener información acerca de roles para federación, consulte Creación de un rol para un proveedor de identidades de terceros en la Guía del usuario de IAM. Si utiliza el IAM Identity Center, debe configurar un conjunto de permisos. IAM Identity Center correlaciona el conjunto de permisos con un rol en IAM para controlar a qué puede acceder las identidades después de autenticarse. Para obtener más información sobre los conjuntos de permisos, consulte Conjuntos de permisos en la Guía del usuario de AWS IAM Identity Center.
- Permisos de usuario de IAM temporales: un usuario de IAM puede asumir un rol de IAM para recibir temporalmente permisos distintos que le permitan realizar una tarea concreta.
- Acceso entre cuentas: puede utilizar un rol de IAM para permitir que alguien (una entidad principal de confianza) de otra cuenta acceda a los recursos de la cuenta. Los roles son la forma principal de conceder acceso entre cuentas. Sin embargo, con algunos servicios de AWS, puede asociar una política directamente a un recurso (en lugar de utilizar un rol como proxy). Para obtener información acerca de la diferencia entre los roles y las políticas basadas en recursos para el acceso entre cuentas, consulta Acceso a recursos entre cuentas en IAM en la Guía del usuario de IAM.
- Acceso entre servicios: algunos servicios de AWS utilizan características de otros servicios de AWS. Por ejemplo, cuando realiza una llamada en un servicio, es común que ese servicio ejecute aplicaciones en Amazon EC2 o almacene objetos en Amazon S3. Es posible que un servicio haga esto usando los permisos de la entidad principal, usando un rol de servicio o usando un rol vinculado a un servicio.
  - Reenviar sesiones de acceso (FAS): cuando utiliza un rol o un usuario de IAM para llevar a cabo las acciones en AWS, se le considera una entidad principal. Cuando utiliza algunos servicios, es posible que realice una acción que desencadene otra acción en un servicio diferente. FAS utiliza los permisos de la entidad principal para llamar a un servicio deAWS, combinados con el servicio de AWS solicitante para realizar solicitudes a servicios posteriores. Las solicitudes de FAS solo se realizan cuando un servicio recibe una solicitud que requiere interacciones con otros servicios o recursos de AWS para completarse. En este caso, debe tener permisos para realizar ambas acciones. Para obtener información sobre las políticas a la hora de realizar solicitudes de FAS, consulte Reenviar sesiones de acceso.

  - Rol de servicio: un rol de servicio es un rol de IAM que adopta un servicio para realizar acciones en su nombre. Un administrador de IAM puede crear, modificar y eliminar un rol de servicio desde IAM. Para obtener más información, consulte Creación de un rol para delegar permisos a un servicio de AWS en la Guía del usuario de IAM.
  - Rol vinculado a un servicio: un rol vinculado a un servicio es un tipo de función del servicio que está vinculado a un servicio de AWS. El servicio puede asumir el rol para realizar una acción en su nombre. Los roles vinculados a servicios aparecen en la cuenta de AWS y son propiedad del servicio. Un administrador de IAM puede ver, pero no editar, los permisos de los roles vinculados a servicios.
- Aplicaciones que se ejecutan en Amazon EC2: puede utilizar un rol de IAM para administrar credenciales temporales para las aplicaciones que se ejecutan en una instancia de EC2 y realizan solicitudes a la CLI de AWS o a la API de AWS. Es preferible hacerlo de este modo a almacenar claves de acceso en la instancia de EC2. Para asignar un rol de AWS a una instancia de EC2 y ponerla a disposición de todas las aplicaciones, cree un perfil de instancia adjuntado a la instancia. Un perfil de instancia contiene el rol y permite a los programas que se ejecutan en la instancia de EC2 obtener credenciales temporales. Para más información, consulte Uso de un rol de IAM para conceder permisos a aplicaciones que se ejecutan en instancias Amazon EC2 en la Guía del usuario de IAM.

Para obtener información sobre el uso de los roles de IAM, consulte Cuándo crear un rol de IAM (en lugar de un usuario) en la Guía del usuario de IAM.

Administración del acceso con políticas

Para controlar el acceso en AWS, se crean políticas y se adjuntan a identidades o recursos de AWS. Una política es un objeto de AWS que, cuando se asocia a una identidad o un recurso, define sus permisos. AWS evalúa estas políticas cuando una entidad principal (sesión de rol, usuario o usuario raíz) realiza una solicitud. Los permisos en las políticas determinan si la solicitud se permite o se deniega. La mayoría de las políticas se almacenan en AWS como documentos JSON. Para obtener más información sobre la estructura y el contenido de los documentos de política JSON, consulte Información general de políticas JSON en la Guía del usuario de IAM.

Los administradores pueden utilizar las políticas JSON de AWS para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

De forma predeterminada, los usuarios y los roles no tienen permisos. Un administrador de IAM puede crear políticas de IAM para conceder permisos a los usuarios para realizar acciones en los

recursos que necesitan. A continuación, el administrador puede añadir las políticas de IAM a roles y los usuarios pueden asumirlos.

Las políticas de IAM definen permisos para una acción independientemente del método que se utiliza para realizar la operación. Por ejemplo, suponga que dispone de una política que permite la acción `iam:GetRole`. Un usuario con esa política puede obtener información sobre roles de la Consola de administración de AWS, la AWS CLI o la API de AWS.

Políticas basadas en identidades

Las políticas basadas en identidad son documentos de políticas de permisos JSON que puede asociar a una identidad, como un usuario de IAM, un grupo de usuarios o un rol. Estas políticas controlan qué acciones pueden realizar los usuarios y los roles, en qué recursos y en qué condiciones. Para obtener más información sobre cómo crear una política basada en identidad, consulte Creación de políticas de IAM en la Guía del usuario de IAM.

Las políticas basadas en identidades pueden clasificarse además como políticas insertadas o políticas administradas. Las políticas insertadas se integran directamente en un único usuario, grupo o rol. Las políticas administradas son políticas independientes que puede asociar a varios usuarios, grupos y roles de su cuenta de AWS. Las políticas administradas incluyen las políticas administradas por AWS y las políticas administradas por el cliente. Para más información sobre cómo elegir una política administrada o una política insertada, consulte Elegir entre políticas administradas y políticas insertadas en la Guía del usuario de IAM.

Políticas basadas en recursos

Las políticas basadas en recursos son documentos de política JSON que se asocian a un recurso. Los ejemplos de políticas basadas en recursos son las políticas de confianza de roles de IAM y las políticas de bucket de Amazon S3. En los servicios que admiten políticas basadas en recursos, los administradores de servicios pueden utilizarlos para controlar el acceso a un recurso específico. Para el recurso al que se asocia la política, la política define qué acciones puede realizar una entidad principal especificada en ese recurso y en qué condiciones. Debe especificar una entidad principal en una política basada en recursos. Las entidades principales pueden incluir cuentas, usuarios, roles, usuarios federados o servicios de AWS.

Las políticas basadas en recursos son políticas insertadas que se encuentran en ese servicio. No se puede utilizar políticas de IAM administradas por AWS en una política basada en recursos.

Listas de control de acceso (ACL)

Las listas de control de acceso (ACL) controlan qué entidades principales (miembros de cuentas, usuarios o roles) tienen permisos para acceder a un recurso. Las ACL son similares a las políticas basadas en recursos, aunque no utilizan el formato de documento de políticas JSON.

Amazon S3, AWS WAF y Amazon VPC son ejemplos de servicios que admiten las ACL. Para obtener más información sobre las ACL, consulte Información general de Lista de control de acceso (ACL) en la Guía para desarrolladores de Amazon Simple Storage Service.

Otros tipos de políticas

AWS admite otros tipos de políticas adicionales menos frecuentes. Estos tipos de políticas pueden establecer el máximo de permisos que los tipos de políticas más frecuentes le conceden.

- Límites de permisos: un límite de permisos es una característica avanzada que le permite establecer los permisos máximos que una política basada en identidad puede conceder a una entidad de IAM (usuario o rol de IAM). Puede establecer un límite de permisos para una entidad. Los permisos resultantes son la intersección de las políticas basadas en la identidad de la entidad y los límites de permisos. Las políticas basadas en recursos que especifiquen el usuario o rol en el campo `Principal` no estarán restringidas por el límite de permisos. Una denegación explícita en cualquiera de estas políticas anulará el permiso. Para obtener más información sobre los límites de los permisos, consulte Límites de permisos para las entidades de IAM en la Guía del usuario de IAM.
- Políticas de control de servicios (SCP): las SCP son políticas JSON que especifican los permisos máximos para una organización o unidad organizativa (OU) en AWS Organizations. AWS Organizations es un servicio que le permite agrupar y administrar de forma centralizada varias cuentas de AWS que posee su negocio. Si habilita todas las características en una organización, entonces podrá aplicar políticas de control de servicio (SCP) a una o a todas sus cuentas. Las SCP limitan los permisos de las entidades de las cuentas miembro, incluido cada usuario raíz de la cuenta de AWS. Para obtener más información acerca de SCP y Organizations, consulte Políticas de control de servicios en la Guía del usuario de AWS.
- Políticas de sesión: las políticas de sesión son políticas avanzadas que se pasan como parámetro cuando se crea una sesión temporal mediante programación para un rol o un usuario federado. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidad del rol y las políticas de la sesión. Los permisos también pueden proceder de una política basada en recursos. Una denegación explícita en cualquiera de estas políticas anulará el permiso. Para más información, consulte Políticas de sesión en la Guía del usuario de IAM.

Varios tipos de políticas

Cuando se aplican varios tipos de políticas a una solicitud, los permisos resultantes son más complicados de entender. Para obtener información acerca de cómo AWS decide si permitir o no una solicitud cuando hay varios tipos de políticas implicados, consulte Lógica de evaluación de políticas en la Guía del usuario de IAM.

Cómo funciona Amazon EKS con IAM

Antes de utilizar IAM para administrar el acceso a Amazon EKS, debe conocer qué características de IAM se encuentran disponibles con Amazon EKS. Para obtener una perspectiva general sobre cómo funcionan Amazon EKS y otros servicios de AWS con IAM, consulte Servicios de AWS que funcionan con IAM en la Guía del usuario de IAM.

Temas

- Políticas de Amazon EKS basadas en identidades
- Políticas basadas en recursos de Amazon EKS
- Autorización basada en etiquetas de Amazon EKS
- Roles de IAM de Amazon EKS

Políticas de Amazon EKS basadas en identidades

Con las políticas basadas en identidades de IAM, puede especificar las acciones y los recursos permitidos o denegados, así como las condiciones en las que se permiten o deniegan las acciones. Amazon EKS admite acciones, claves de condiciones y recursos específicos. Para obtener más información acerca de los elementos que utiliza en una política de JSON, consulte Referencia de los elementos de las políticas de JSON de IAM en la Guía del usuario de IAM.

Acciones

Los administradores pueden utilizar las políticas JSON de AWS para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Action` de una política JSON describe las acciones que puede utilizar para conceder o denegar el acceso en una política. Las acciones de la política generalmente tienen el mismo nombre que la operación de API de AWS asociada. Hay algunas excepciones, como acciones de solo permiso que no tienen una operación de API coincidente. También hay algunas operaciones

que requieren varias acciones en una política. Estas acciones adicionales se denominan acciones dependientes.

Incluya acciones en una política para conceder permisos y así llevar a cabo la operación asociada.

Las acciones de políticas de Amazon EKS utilizan el siguiente prefijo antes de la acción: `eks:`. Por ejemplo, para conceder permiso a alguien para conseguir información descriptiva sobre un clúster de Amazon EKS, incluya la acción `DescribeCluster` en su política. Las instrucciones de la política deben incluir un elemento `Action` o un elemento `NotAction`.

Para especificar varias acciones en una única instrucción, sepárelas con comas del siguiente modo:

```
"Action": ["eks:action1", "eks:action2"]
```

Puede utilizar caracteres comodín para especificar varias acciones (*). Por ejemplo, para especificar todas las acciones que comiencen con la palabra `Describe`, incluya la siguiente acción:

```
"Action": "eks:Describe*"
```

Para ver una lista de las acciones de Amazon EKS, consulte Acciones definidas por Amazon Elastic Kubernetes Service en la Referencia de autorizaciones de servicio.

Recursos

Los administradores pueden utilizar las políticas JSON de AWS para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Resource` de la política JSON especifica el objeto u objetos a los que se aplica la acción. Las instrucciones deben contener un elemento `Resource` o `NotResource`. Como práctica recomendada, especifique un recurso utilizando el Nombre de recurso de Amazon (ARN). Puede hacerlo para acciones que admitan un tipo de recurso específico, conocido como permisos de nivel de recurso.

Para las acciones que no admiten permisos de nivel de recurso, como las operaciones de descripción, utilice un carácter asterisco (*) para indicar que la instrucción se aplica a todos los recursos.

```
"Resource": "*"
```

El recurso del clúster de Amazon EKS tiene el siguiente ARN.

```
arn:aws:eks:region-code:account-id:cluster/cluster-name
```

Para obtener más información acerca del formato de los ARN, consulte Nombres de recursos de Amazon (ARN) y espacios de nombres de servicios de AWS.

Por ejemplo, para especificar el clúster de con el nombre *my-cluster* en su instrucción, utilice el siguiente ARN:

```
"Resource": "arn:aws:eks:region-code:111122223333:cluster/my-cluster"
```

Para especificar todos los clústeres que pertenecen a una cuenta y una región de AWS específicas, utilice el carácter comodín (*):

```
"Resource": "arn:aws:eks:region-code:111122223333:cluster/*"
```

Algunas acciones de Amazon EKS, como las que se utilizan para crear recursos, no se pueden llevar a cabo en un recurso específico. En dichos casos, debe utilizar el carácter comodín (*).

```
"Resource": "*"
```

Para ver una lista de los tipos de recursos de Amazon EKS y sus ARN, consulte Recursos definidos por Amazon Elastic Kubernetes Service en la Referencia de autorizaciones de servicio. Para obtener información sobre las acciones con las que puede especificar el ARN de cada recurso, consulte Acciones definidas por Amazon Elastic Kubernetes Service.

Claves de condición

Amazon EKS define su propio conjunto de claves de condición y también admite el uso de algunas claves de condición globales. Para ver todas las claves de condición globales de AWS, consulte Claves de contexto de condición globales de AWS en la Guía del usuario de IAM.

Puede establecer claves de condición al asociar un proveedor de OpenID Connect al clúster. Para obtener más información, consulte the section called “Política de IAM de ejemplo”.

Todas las acciones de Amazon EC2 admiten las claves de condición `aws:RequestedRegion` y `ec2:Region`. Para obtener más información, consulte Ejemplo: restricción del acceso a una región de AWS específica.

Para obtener una lista de las claves de condición de Amazon EKS, consulte Condiciones de Amazon Elastic Kubernetes Service en la Referencia de autorizaciones de servicio. Para obtener más información sobre las acciones y los recursos con los que puede utilizar una clave de condición, consulte Acciones definidas por Amazon Elastic Kubernetes Service.

Ejemplos

Para ver ejemplos de políticas de Amazon EKS basadas en identidades, consulte the section called "Políticas basadas en identidades".

Cuando se crea un clúster de Amazon EKS, la entidad principal de IAM que crea el clúster recibe permisos de `system:masters` de forma automática en la configuración del role-based access control (RBAC, control de acceso basado en roles) del clúster en el plano de control de Amazon EKS. Esta entidad principal no aparece en ninguna configuración visible, así que asegúrese de realizar un seguimiento de la entidad principal que creó el clúster originalmente. Para conceder a entidades principales adicionales de IAM la capacidad de interactuar con el clúster, edite el `aws-auth ConfigMap` dentro de Kubernetes y cree un `rolebinding` o `clusterrolebinding` de Kubernetes con el nombre de un `group` que especifique en el `aws-auth ConfigMap`.

Para obtener más información sobre cómo trabajar con el ConfigMap, consulte the section called "Acceso a la API de Kubernetes".

Políticas basadas en recursos de Amazon EKS

Amazon EKS no admite las políticas basadas en recursos.

Autorización basada en etiquetas de Amazon EKS

Puede asociar etiquetas a los recursos de Amazon EKS o transferirlas en una solicitud a Amazon EKS. Para controlar el acceso en función de etiquetas, debe proporcionar información de las etiquetas en el elemento de condición de una política utilizando las claves de condición `aws:ResourceTag/key-name` , `aws:RequestTag/key-name` o `aws:TagKeys`. Para obtener más información sobre el etiquetado de recursos de Amazon EKS, consulte the section called "Etiquetado de recursos". Para obtener más información sobre qué acciones puede usar las etiquetas en las claves de condición de, consulte Acciones definidas por Amazon EKS en la Referencia de autorizaciones de servicio.

Roles de IAM de Amazon EKS

Un rol de IAM es una entidad de la cuenta de AWS que dispone de permisos específicos.

Uso de credenciales temporales con Amazon EKS

Puede utilizar credenciales temporales para iniciar sesión con federación, asumir un rol de IAM o asumir un rol de acceso entre cuentas. Para obtener credenciales de seguridad temporales debe realizar una llamada a operaciones de la API de AWS STS como AssumeRole o GetFederationToken.

Amazon EKS admite el uso de credenciales temporales.

Roles vinculados a servicios

Los roles vinculados a servicios permiten a los servicios de AWS obtener acceso a los recursos de otros servicios para completar una acción en su nombre. Los roles vinculados a servicios aparecen en la cuenta de IAM y son propiedad del servicio. Un administrador puede ver, pero no editar, los permisos de los roles vinculados a servicios.

Amazon EKS admite roles vinculados a servicios. Para obtener más información sobre cómo crear o administrar roles vinculados a servicios de Amazon EKS, consulte the section called “Roles vinculados a servicios”.

Roles de servicio

Esta característica permite que un servicio asuma un rol de servicio en su nombre. Este rol permite que el servicio obtenga acceso a los recursos de otros servicios para completar una acción en su nombre. Los roles de servicio aparecen en su cuenta de IAM y son propiedad de la cuenta. Esto significa que un administrador de IAM puede cambiar los permisos de este rol. Sin embargo, hacerlo podría deteriorar la funcionalidad del servicio.

Amazon EKS admite roles de servicio. Para obtener más información, consulte the section called “Rol de IAM de clúster” y the section called “Rol de IAM de nodo”.

Elegir un rol de IAM en Amazon EKS

Cuando se crea un recurso de clúster en Amazon EKS, debe elegir un rol para permitir a Amazon EKS acceder a otros recursos de AWS en su nombre. Si ya ha creado una función del servicio, Amazon EKS proporciona una lista de roles para elegir. Es importante que elija un rol que cuente con políticas administradas de Amazon EKS asociadas a él. Para obtener más información, consulte the section called “Comprobar si existe un rol de clúster existente” y the section called “Verificar un rol de nodo existente”.

Ejemplos de políticas de Amazon EKS basadas en identidades

De forma predeterminada, los usuarios y roles de IAM no tienen permiso para crear ni modificar recursos de Amazon EKS. Tampoco pueden realizar tareas mediante Consola de administración de AWS, AWS CLI o la API de AWS. Un administrador de IAM debe crear políticas de IAM que concedan permisos a los usuarios y a los roles para realizar operaciones de la API concretas en los recursos especificados que necesiten. El administrador debe adjuntar esas políticas a los usuarios o grupos de IAM que necesiten esos permisos.

Para obtener más información acerca de cómo crear una política basada en identidad de IAM con estos documentos de políticas de JSON de ejemplo, consulte Creación de políticas en la pestaña JSON en la Guía del usuario de IAM.

Cuando se crea un clúster de Amazon EKS, la entidad principal de IAM que crea el clúster recibe permisos de `system:masters` de forma automática en la configuración del role-based access control (RBAC, control de acceso basado en roles) del clúster en el plano de control de Amazon EKS. Esta entidad principal no aparece en ninguna configuración visible, así que asegúrese de realizar un seguimiento de la entidad principal que creó el clúster originalmente. Para conceder a entidades principales adicionales de IAM la capacidad de interactuar con el clúster, edite el `aws-auth ConfigMap` dentro de Kubernetes y cree un `rolebinding` o `clusterrolebinding` de Kubernetes con el nombre de un `group` que especifique en el `aws-auth ConfigMap`.

Para obtener más información sobre cómo trabajar con el ConfigMap, consulte the section called “Acceso a la API de Kubernetes”.

Temas

- Prácticas recomendadas sobre las políticas
- Utilizar la consola de Amazon EKS
- Permitir a los usuarios de IAM ver sus propios permisos
- Creación de un clúster de Kubernetes en la nube de AWS
- Creación de un clúster local de Kubernetes en un Outpost
- Actualización de un clúster de Kubernetes
- Enumeración o descripción de todos los clústeres

Prácticas recomendadas sobre las políticas

Las políticas basadas en identidades determinan si alguien puede crear, eliminar o acceder a los recursos de Amazon EKS de su cuenta. Estas acciones pueden generar costos adicionales para su cuenta de AWS. Siga estas directrices y recomendaciones al crear o editar políticas basadas en identidades:

- Comience con las políticas administradas por AWS y continúe con los permisos de privilegio mínimo: a fin de comenzar a conceder permisos a los usuarios y las cargas de trabajo, utilice las políticas administradas por AWS, que conceden permisos para muchos casos de uso comunes. Están disponibles en su cuenta de AWS. Se recomienda definir políticas administradas por el cliente de AWS específicas para sus casos de uso a fin de reducir aún más los permisos. Con el fin de obtener más información, consulte las políticas administradas por AWS o las políticas administradas por AWS para funciones de tarea en la Guía de usuario de IAM.
- Aplique permisos de privilegio mínimo: cuando establezca permisos con políticas de IAM, conceda solo los permisos necesarios para realizar una tarea. Para ello, debe definir las acciones que se pueden llevar a cabo en determinados recursos en condiciones específicas, también conocidos como permisos de privilegios mínimos. Con el fin de obtener más información sobre el uso de IAM para aplicar permisos, consulte Políticas y permisos en IAM en la Guía del usuario de IAM.
- Utilice condiciones en las políticas de IAM para restringir aún más el acceso: puede agregar una condición a sus políticas para limitar el acceso a las acciones y los recursos. Por ejemplo, puede escribir una condición de políticas para especificar que todas las solicitudes deben enviarse utilizando SSL. También puede usar condiciones para conceder acceso a acciones de servicios si se emplean a través de un servicio determinado de AWS como, por ejemplo, AWS CloudFormation. Para obtener más información, consulte Elementos de la política de JSON de IAM: Condición en la Guía del usuario de IAM.
- Utiliza el analizador de acceso de IAM para validar las políticas de IAM con el fin de garantizar la seguridad y funcionalidad de los permisos: el analizador de acceso de IAM valida políticas nuevas y existentes para que respeten el lenguaje (JSON) de las políticas de IAM y las prácticas recomendadas de IAM. El analizador de acceso de IAM proporciona más de 100 verificaciones de políticas y recomendaciones procesables para ayudar a crear políticas seguras y funcionales. Para más información, consulte Política de validación de Analizador de acceso de IAM en la Guía de usuario de IAM.
- Solicite la autenticación multifactor (MFA): si se encuentra en una situación en la que necesita un usuario raíz o usuarios de IAM en su cuenta de AWS, active la MFA para mayor seguridad. Para solicitar la MFA cuando se invocan las operaciones de la API, agregue las condiciones de la MFA

a sus políticas. Para más información, consulte Configuración del acceso a una API protegido por MFA en la Guía de usuario de IAM.

Para obtener más información sobre las prácticas recomendadas de IAM, consulte las Prácticas recomendadas de seguridad en IAM en la Guía del usuario de IAM.

Utilizar la consola de Amazon EKS

Para acceder a la consola de Amazon EKS, una entidad principal de IAM debe tener un conjunto mínimo de permisos. Estos permisos permiten que la entidad principal enumere y vea detalles sobre los recursos de Amazon EKS en su cuenta de AWS. Si crea una política basada en identidad que sea más restrictiva que el mínimo de permisos necesarios, la consola no funcionará del modo esperado para las entidades principales que tengan esa política adjunta.

Para asegurarse de que las entidades principales de IAM puedan seguir utilizando la consola de Amazon EKS, cree una política con su propio nombre, como `AmazonEKSAdminPolicy`. Adjunte la política a las entidades principales. Para obtener más información, consulte Adición y eliminación de permisos de identidad de IAM en la Guía del usuario de IAM.

> **Important**
>
> La siguiente política de ejemplo permite que una entidad principal vea información en la pestaña Configuración en la consola. Para ver información en las pestañas Resumen y Recursos en la Consola de administración de AWS, la entidad principal también necesita permisos de Kubernetes. Para obtener más información, consulte the section called “Permisos necesarios”.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks:*"
            ],
            "Resource": "*"
        },
        {
```

```
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "eks.amazonaws.com"
                }
            }
        }
    ]
}
```

No es necesario conceder permisos mínimos para la consola a las entidades principales que solo realizan llamadas a la AWS CLI o a la API de AWS. En su lugar, permite acceso únicamente a las acciones que coincidan con la operación de API que intenta realizar.

Permitir a los usuarios de IAM ver sus propios permisos

En este ejemplo, se muestra cómo podría crear una política que permita a los usuarios de IAM ver las políticas administradas e insertadas que se asocian a la identidad de sus usuarios. Esta política incluye permisos para completar esta acción en la consola o mediante programación con la CLI de AWS o la API de AWS.

```
{
    "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": "*"
        }
    ]
}
```

Creación de un clúster de Kubernetes en la nube de AWS

En esta política de ejemplo se incluyen los permisos mínimos necesarios para crear un clúster de Amazon EKS denominado *my-cluster* en la región de AWS *us-west-2*. Puede reemplazar la región de AWS por la región de AWS en la que desea implementar un clúster. Si ve una advertencia que diga Las acciones de su política no admiten permisos de nivel de recursos y requieren que elija **All resources** en la Consola de administración de AWS, puede omitirla con toda tranquilidad. Si su cuenta ya tiene el rol *AWSServiceRoleForAmazonEKS*, puede quitar la acción `iam:CreateServiceLinkedRole` de la política. Si alguna vez creó un clúster de Amazon EKS en su cuenta, este rol ya existe, a menos que lo haya eliminado.

```
{
    "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"
        }
    ]
}
```

Creación de un clúster local de Kubernetes en un Outpost

En esta política de ejemplo se incluyen los permisos mínimos necesarios para crear un clúster local de Amazon EKS denominado *my-cluster* en un Outpost en la región de AWS *us-west-2*. Puede reemplazar la región de AWS por la región de AWS en la que desea implementar un clúster. Si ve una advertencia que diga Las acciones de su política no admiten permisos de nivel de recursos y requieren que elija **All resources** en la Consola de administración de AWS, puede omitirla con toda tranquilidad. Si su cuenta ya tiene el rol `AWSServiceRoleForAmazonEKSLocalOutpost`, puede quitar la acción `iam:CreateServiceLinkedRole` de la política. Si alguna vez creó un clúster local de Amazon EKS en un Outpost en su cuenta, este rol ya existe, a menos que lo haya eliminado.

```
{
    "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"
        }
    ]
}
```

Actualización de un clúster de Kubernetes

En esta política de ejemplo se incluyen los permisos mínimos necesarios para actualizar un clúster de Amazon EKS denominado *my-cluster* en la región de AWS us-west-2.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "eks:UpdateClusterVersion",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-cluster"
        }
    ]
}
```

Enumeración o descripción de todos los clústeres

En esta política de ejemplo se incluyen los permisos mínimos necesarios para enumerar y describir todos los clústeres de su cuenta. Una entidad principal de IAM tiene que ser capaz de enumerar y describir clústeres para usar el comando de `update-kubeconfig` de AWS CLI.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks:DescribeCluster",
                "eks:ListClusters"
            ],
            "Resource": "*"
        }
    ]
}
```

Utilizar roles vinculados a servicios para Amazon EKS

Amazon Elastic Kubernetes Service utiliza roles vinculados a servicios de AWS Identity and Access Management (IAM). Un rol vinculado a servicios es un tipo único de rol de IAM que se encuentra vinculado directamente a Amazon EKS. Los roles vinculados a servicios se encuentran predefinidos por Amazon EKS e incluyen todos los permisos que el servicio requiere para llamar a otros servicios de AWS en su nombre.

Temas

- Utilizar roles para clústeres de Amazon EKS
- Utilizar roles para grupos de nodos de Amazon EKS
- Utilizar roles para perfiles de Fargate de Amazon EKS
- Uso de roles para conectar un clúster de Kubernetes a Amazon EKS
- Uso de roles para clústeres locales de Amazon EKS en Outpost
- Uso de roles para el panel de Amazon EKS

Utilizar roles para clústeres de Amazon EKS

Amazon Elastic Kubernetes Service utiliza roles vinculados a servicios de AWS Identity and Access Management (IAM). Un rol vinculado a servicios es un tipo único de rol de IAM que se encuentra vinculado directamente a Amazon EKS. Los roles vinculados a servicios se encuentran predefinidos por Amazon EKS e incluyen todos los permisos que el servicio requiere para llamar a otros servicios de AWS en su nombre.

Un rol vinculado a servicios simplifica la configuración de Amazon EKS porque ya no tendrá que agregar de forma manual los permisos necesarios. Amazon EKS define los permisos de sus roles vinculados a servicios y, a menos que esté definido de otra manera, solo Amazon EKS puede asumir sus roles. Los permisos definidos incluyen las políticas de confianza y de permisos, y que la política de permisos no se pueda adjuntar a ninguna otra entidad de IAM.

Solo es posible eliminar un rol vinculado a un servicio después de eliminar sus recursos relacionados. De esta forma, se protegen los recursos de Amazon EKS, ya que se evita que se puedan eliminar accidentalmente permisos de acceso a los recursos.

Para obtener información sobre otros servicios que admiten roles vinculados al servicio, consulte Servicios de AWS que funcionan con IAM y busque los servicios que muestran Yes (Sí) en la columna Service-linked role (Rol vinculado al servicio). Seleccione una opción Sí con un enlace para ver la documentación acerca del rol vinculado al servicio en cuestión.

Permisos de roles vinculados a servicios para Amazon EKS

Amazon MSK usa el rol vinculado al servicio denominado `AWSServiceRoleForAmazonEKS`. El rol permite a Amazon EKS administrar clústeres de la cuenta. Las políticas adjuntas permiten que el rol administre los siguientes recursos: interfaces de red, grupos de seguridad, registros y VPC.

> **Note**
>
> El rol vinculado al servicio `AWSServiceRoleForAmazonEKS` es distinto del rol requerido para la creación de clústeres. Para obtener más información, consulte the section called “Rol de IAM de clúster”.

El rol vinculado al servicio `AWSServiceRoleForAmazonEKS` confía en los siguientes servicios para asumir el rol:

- `eks.amazonaws.com`

La política de permisos del rol permite que Amazon EKS realice las siguientes acciones en los recursos especificados:

- AmazonEKSServiceRolePolicy

Debe configurar permisos para permitir a una entidad de IAM (como un usuario, grupo o rol) crear, editar o eliminar un rol vinculado a servicios. Para obtener más información, consulte Permisos de roles vinculados a servicios en la Guía del usuario de IAM.

Crear un rol vinculado a servicios para Amazon EKS

No necesita crear un rol vinculado a servicios de manera manual. Cuando crea un clúster en la Consola de administración de AWS, la AWS CLI, o la API de AWS, Amazon EKS crea el rol vinculado a servicios en su nombre.

Si elimina este rol vinculado a servicios y necesita crearlo de nuevo, puede utilizar el mismo proceso para volver a crear el rol en su cuenta. Al crear un clúster, Amazon EKS se encarga de crear de nuevo el rol vinculado a servicios en su nombre.

Editar un rol vinculado a servicios para Amazon EKS

Amazon EKS no permite editar el rol vinculado a servicios de `AWSServiceRoleForAmazonEKS`. Después de crear un rol vinculado al servicio, no podrá cambiar el nombre del rol, ya que varias entidades podrían hacer referencia al rol. Sin embargo, sí puede editar la descripción del rol con IAM. Para obtener más información, consulte Editing a service-linked role (Editar un rol vinculado a servicios) en la Guía del usuario de IAM.

Eliminar un rol vinculado a servicios para Amazon EKS

Si ya no necesita usar una característica o servicio que requieran un rol vinculado a un servicio, le recomendamos que elimine dicho rol. De esta forma, no tiene una entidad no utilizada que no se monitoree ni mantenga de forma activa. Sin embargo, debe limpiar el rol vinculado a servicios antes de eliminarlo manualmente.

Limpiar un rol vinculado a servicios

Antes de que pueda utilizar IAM para eliminar un rol vinculado a servicios, primero debe eliminar los recursos que utiliza el rol.

> **Note**
>
> Si el servicio de Amazon EKS utiliza el rol cuando intenta eliminar los recursos, la eliminación podría producir un error. En tal caso, espere unos minutos e intente de nuevo la operación.

1. Abra la consola de Amazon EKS.
2. En el panel de navegación izquierdo, elija Clusters (Clústeres).
3. Si el clúster tiene grupos de nodos o perfiles de Fargate, debe eliminarlos para poder eliminar el clúster. Para obtener más información, consulte the section called “Eliminar” y the section called “Eliminar perfiles”.
4. En la página Clusters (Clústeres), elija el clúster que desea eliminar y elija Delete (Eliminar).
5. Escriba el nombre del clúster en la ventana de confirmación de eliminación y, a continuación, elija Delete (Eliminar).
6. Repita este procedimiento para el resto de los clústeres de la cuenta. Espere a que finalicen todas las operaciones de eliminación.

Eliminación manual de un rol vinculado a servicios

Utilice la consola de IAM, la CLI de AWS o la API de AWS para eliminar el rol vinculado al servicio de `AWSServiceRoleForAmazonEKS`. Para obtener más información, consulte Eliminar un rol vinculado a un servicio en la Guía del usuario de IAM.

Regiones admitidas para los roles vinculados a servicios de Amazon EKS

Amazon EKS admite el uso de roles vinculados a servicios en todas las regiones en las que se encuentra disponible el servicio. Para obtener más información, consulte Cuotas y puntos de conexión de Amazon EKS.

Utilizar roles para grupos de nodos de Amazon EKS

Amazon EKS utiliza roles vinculados a servicios de AWS Identity and Access Management (IAM). Un rol vinculado a servicios es un tipo único de rol de IAM que se encuentra vinculado directamente a Amazon EKS. Los roles vinculados a servicios se encuentran predefinidos por Amazon EKS e incluyen todos los permisos que el servicio requiere para llamar a otros servicios de AWS en su nombre.

Un rol vinculado a servicios simplifica la configuración de Amazon EKS porque ya no tendrá que agregar de forma manual los permisos necesarios. Amazon EKS define los permisos de sus roles vinculados a servicios y, a menos que esté definido de otra manera, solo Amazon EKS puede asumir sus roles. Los permisos definidos incluyen las políticas de confianza y de permisos, y que la política de permisos no se pueda adjuntar a ninguna otra entidad de IAM.

Solo es posible eliminar un rol vinculado a un servicio después de eliminar sus recursos relacionados. De esta forma, se protegen los recursos de Amazon EKS, ya que se evita que se puedan eliminar accidentalmente permisos de acceso a los recursos.

Para obtener información sobre otros servicios que admiten roles vinculados al servicio, consulte Servicios de AWS que funcionan con IAM y busque los servicios que muestran Yes (Sí) en la columna Service-linked role (Rol vinculado al servicio). Seleccione una opción Sí con un enlace para ver la documentación acerca del rol vinculado al servicio en cuestión.

Permisos de roles vinculados a servicios para Amazon EKS

Amazon MSK usa el rol vinculado al servicio denominado `AWSServiceRoleForAmazonEKSNodegroup`. El rol permite a Amazon EKS administrar grupos de nodos de la cuenta. La política asociada de `AWSServiceRoleForAmazonEKSNodegroup` permite al rol administrar los siguientes recursos: grupos de escalado automático, grupos de seguridad, plantillas de lanzamiento y perfiles de instancia de IAM. Para obtener más información, consulte the section called “Política administrada de AWS: AWSServiceRoleForAmazonEKSNodegroup”.

El rol vinculado al servicio `AWSServiceRoleForAmazonEKSNodegroup` confía en los siguientes servicios para asumir el rol:

- `eks-nodegroup.amazonaws.com`

La política de permisos del rol permite que Amazon EKS realice las siguientes acciones en los recursos especificados:

- AWSServiceRoleForAmazonEKSNodegroup

Debe configurar permisos para permitir a una entidad de IAM (como un usuario, grupo o rol) crear, editar o eliminar un rol vinculado a servicios. Para obtener más información, consulte Permisos de roles vinculados a servicios en la Guía del usuario de IAM.

Crear un rol vinculado a servicios para Amazon EKS

No necesita crear un rol vinculado a servicios de manera manual. Cuando ejecuta CreateNodegroup en la Consola de administración de AWS, la AWS CLI, o la API de AWS, Amazon EKS crea el rol vinculado a servicios en su nombre.

> **Important**
>
> Este rol vinculado a servicios puede aparecer en su cuenta si se ha completado una acción en otro servicio que utilice las características compatibles con este rol. Si utilizaba el servicio de Amazon EKS antes del 1 de enero de 2017, fecha en que comenzó a admitir los roles vinculados a servicios, Amazon EKS creó el rol AWSServiceRoleForAmazonEKSNodegroup en su cuenta. Para obtener más información, consulte Un nuevo rol ha aparecido en mi cuenta de IAM.

Crear un rol vinculado a servicios en Amazon EKS (API de AWS)

No necesita crear un rol vinculado a servicios de manera manual. Cuando crea un grupo de nodos administrados en la Consola de administración de AWS, la AWS CLI o la API de AWS, Amazon EKS crea el rol vinculado a servicios en su nombre.

Si elimina este rol vinculado a servicios y necesita crearlo de nuevo, puede utilizar el mismo proceso para volver a crear el rol en su cuenta. Cuando crea otro grupo de nodos administrado, Amazon EKS vuelve a crear el rol vinculado a servicios.

Editar un rol vinculado a servicios para Amazon EKS

Amazon EKS no permite editar el rol vinculado a servicios de `AWSServiceRoleForAmazonEKSNodegroup`. Después de crear un rol vinculado al servicio, no podrá cambiar el nombre del rol, ya que varias entidades podrían hacer referencia al rol. Sin embargo, sí puede editar la descripción del rol con IAM. Para obtener más información, consulte Editing a service-linked role (Editar un rol vinculado a servicios) en la Guía del usuario de IAM.

Eliminar un rol vinculado a servicios para Amazon EKS

Si ya no necesita usar una característica o servicio que requieran un rol vinculado a un servicio, le recomendamos que elimine dicho rol. De esta forma, no tiene una entidad no utilizada que no se monitoree ni mantenga de forma activa. Sin embargo, debe limpiar el rol vinculado a servicios antes de eliminarlo manualmente.

Limpiar un rol vinculado a servicios

Antes de que pueda utilizar IAM para eliminar un rol vinculado a servicios, primero debe eliminar los recursos que utiliza el rol.

> **Note**
>
> Si el servicio de Amazon EKS utiliza el rol cuando intenta eliminar los recursos, la eliminación podría producir un error. En tal caso, espere unos minutos e intente de nuevo la operación.

1. Abra la consola de Amazon EKS.
2. En el panel de navegación izquierdo, elija Clusters (Clústeres).
3. Seleccione la pestaña Compute (Informática).
4. En la sección de Node Groups (Grupos de nodos), elija el grupo de nodos que desea eliminar.
5. Escriba el nombre del grupo de nodos en la ventana de confirmación de eliminación y, a continuación, elija Delete (Eliminar).
6. Repita este procedimiento para los demás grupos de nodos del clúster. Espere a que finalicen todas las operaciones de eliminación.

Eliminación manual de un rol vinculado a servicios

Utilice la consola de IAM, la CLI de AWS o la API de AWS para eliminar el rol vinculado al servicio de `AWSServiceRoleForAmazonEKSNodegroup`. Para obtener más información, consulte Eliminar un rol vinculado a un servicio en la Guía del usuario de IAM.

Regiones admitidas para los roles vinculados a servicios de Amazon EKS

Amazon EKS admite el uso de roles vinculados a servicios en todas las regiones en las que se encuentra disponible el servicio. Para obtener más información, consulte Cuotas y puntos de conexión de Amazon EKS.

Utilizar roles para perfiles de Fargate de Amazon EKS

Amazon EKS utiliza roles vinculados a servicios de AWS Identity and Access Management (IAM). Un rol vinculado a servicios es un tipo único de rol de IAM que se encuentra vinculado directamente a Amazon EKS. Los roles vinculados a servicios se encuentran predefinidos por Amazon EKS e

incluyen todos los permisos que el servicio requiere para llamar a otros servicios de AWS en su nombre.

Un rol vinculado a servicios simplifica la configuración de Amazon EKS porque ya no tendrá que agregar de forma manual los permisos necesarios. Amazon EKS define los permisos de sus roles vinculados a servicios y, a menos que esté definido de otra manera, solo Amazon EKS puede asumir sus roles. Los permisos definidos incluyen las políticas de confianza y de permisos, y que la política de permisos no se pueda adjuntar a ninguna otra entidad de IAM.

Solo es posible eliminar un rol vinculado a un servicio después de eliminar sus recursos relacionados. De esta forma, se protegen los recursos de Amazon EKS, ya que se evita que se puedan eliminar accidentalmente permisos de acceso a los recursos.

Para obtener información sobre otros servicios que admiten roles vinculados al servicio, consulte Servicios de AWS que funcionan con IAM y busque los servicios que muestran Yes (Sí) en la columna Service-linked role (Rol vinculado al servicio). Seleccione una opción Sí con un enlace para ver la documentación acerca del rol vinculado al servicio en cuestión.

Permisos de roles vinculados a servicios para Amazon EKS

Amazon MSK usa el rol vinculado al servicio denominado `AWSServiceRoleForAmazonEKSForFargate`. El rol permite a Fargate de Amazon EKS configurar la red de VPC que los pods de Fargate necesitan. Las políticas asociadas permiten que el rol cree y elimine interfaces de red elásticas y describa los recursos y las interfaces de red elásticas.

El rol vinculado al servicio `AWSServiceRoleForAmazonEKSForFargate` depende de los siguientes servicios para asumir el rol:

- `eks-fargate.amazonaws.com`

La política de permisos del rol permite que Amazon EKS realice las siguientes acciones en los recursos especificados:

- AmazonEKSForFargateServiceRolePolicy

Debe configurar permisos para permitir a una entidad de IAM (como un usuario, grupo o rol) crear, editar o eliminar un rol vinculado a servicios. Para obtener más información, consulte Permisos de roles vinculados a servicios en la Guía del usuario de IAM.

Crear un rol vinculado a servicios para Amazon EKS

No necesita crear un rol vinculado a servicios de manera manual. Cuando crea un perfil de Fargate en la Consola de administración de AWS, la AWS CLI, o la API de AWS, Amazon EKS crea el rol vinculado a servicios en su nombre.

> **Important**
>
> Este rol vinculado a servicios puede aparecer en su cuenta si se ha completado una acción en otro servicio que utilice las características compatibles con este rol. Si utilizaba el servicio de Amazon EKS antes del 13 de diciembre de 2019, fecha en que comenzó a admitir los roles vinculados a servicios, Amazon EKS creó el rol AWSServiceRoleForAmazonEKSForFargate en su cuenta. Para obtener más información, consulte Un nuevo rol ha aparecido en mi cuenta de IAM.

Crear un rol vinculado a servicios en Amazon EKS (API de AWS)

No necesita crear un rol vinculado a servicios de manera manual. Cuando crea un perfil de Fargate en la Consola de administración de AWS, la AWS CLI, o la API de AWS, Amazon EKS crea el rol vinculado a servicios en su nombre.

Si elimina este rol vinculado a servicios y necesita crearlo de nuevo, puede utilizar el mismo proceso para volver a crear el rol en su cuenta. Cuando crea otro grupo de nodos administrado, Amazon EKS vuelve a crear el rol vinculado a servicios.

Editar un rol vinculado a servicios para Amazon EKS

Amazon EKS no permite editar el rol vinculado a servicios de `AWSServiceRoleForAmazonEKSForFargate`. Después de crear un rol vinculado al servicio, no podrá cambiar el nombre del rol, ya que varias entidades podrían hacer referencia al rol. Sin embargo, sí puede editar la descripción del rol con IAM. Para obtener más información, consulte Editing a service-linked role (Editar un rol vinculado a servicios) en la Guía del usuario de IAM.

Eliminar un rol vinculado a servicios para Amazon EKS

Si ya no necesita usar una característica o servicio que requieran un rol vinculado a un servicio, le recomendamos que elimine dicho rol. De esta forma, no tiene una entidad no utilizada que no se monitoree ni mantenga de forma activa. Sin embargo, debe limpiar el rol vinculado a servicios antes de eliminarlo manualmente.

Limpiar un rol vinculado a servicios

Antes de que pueda utilizar IAM para eliminar un rol vinculado a servicios, primero debe eliminar los recursos que utiliza el rol.

> **Note**
>
> Si el servicio de Amazon EKS utiliza el rol cuando intenta eliminar los recursos, la eliminación podría producir un error. En tal caso, espere unos minutos e intente de nuevo la operación.

1. Abra la consola de Amazon EKS.
2. En el panel de navegación izquierdo, elija Clusters (Clústeres).
3. En la página Clusters (Clústeres), seleccione el clúster.
4. Seleccione la pestaña Compute (Informática).
5. Si hay algún perfil de Fargate en la sección de Fargate Profiles (Perfiles de Fargate), seleccione cada uno de forma individual y, a continuación, elija Delete (Eliminar).
6. Escriba el nombre del perfil en la ventana de confirmación de eliminación y, a continuación, elija Delete (Eliminar).
7. Repita este procedimiento para cualquier otro perfil de Fargate del clúster y cualquier otro clúster de su cuenta.

Eliminación manual de un rol vinculado a servicios

Utilice la consola de IAM, la AWS CLI o la API de AWS para eliminar el rol vinculado a servicios AWSServiceRoleForAmazonEKSForFargate. Para obtener más información, consulte Eliminar un rol vinculado a un servicio en la Guía del usuario de IAM.

Regiones admitidas para los roles vinculados a servicios de Amazon EKS

Amazon EKS admite el uso de roles vinculados a servicios en todas las regiones en las que se encuentra disponible el servicio. Para obtener más información, consulte Cuotas y puntos de conexión de Amazon EKS.

Uso de roles para conectar un clúster de Kubernetes a Amazon EKS

Amazon EKS utiliza roles vinculados a servicios de AWS Identity and Access Management (IAM). Un rol vinculado a servicios es un tipo único de rol de IAM que se encuentra vinculado directamente

a Amazon EKS. Los roles vinculados a servicios se encuentran predefinidos por Amazon EKS e incluyen todos los permisos que el servicio requiere para llamar a otros servicios de AWS en su nombre.

Un rol vinculado a servicios simplifica la configuración de Amazon EKS porque ya no tendrá que agregar de forma manual los permisos necesarios. Amazon EKS define los permisos de sus roles vinculados a servicios y, a menos que esté definido de otra manera, solo Amazon EKS puede asumir sus roles. Los permisos definidos incluyen las políticas de confianza y de permisos, y que la política de permisos no se pueda adjuntar a ninguna otra entidad de IAM.

Solo es posible eliminar un rol vinculado a un servicio después de eliminar sus recursos relacionados. De esta forma, se protegen los recursos de Amazon EKS, ya que se evita que se puedan eliminar accidentalmente permisos de acceso a los recursos.

Para obtener información sobre otros servicios que admiten roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM y busque los servicios que muestran Sí en la columna Rol vinculado a servicios. Seleccione una opción Sí con un enlace para ver la documentación acerca del rol vinculado al servicio en cuestión.

Permisos de roles vinculados a servicios para Amazon EKS

Amazon MSK usa el rol vinculado al servicio denominado `AWSServiceRoleForAmazonEKSConnector`. El rol permite a Amazon EKS conectar clústeres de Kubernetes. Las políticas adjuntas permiten que el rol administre los recursos necesarios para conectarse al clúster de Kubernetes registrado.

El rol vinculado al servicio `AWSServiceRoleForAmazonEKSConnector` depende de los siguientes servicios para asumir el rol:

- `eks-connector.amazonaws.com`

La política de permisos del rol permite que Amazon EKS realice las siguientes acciones en los recursos especificados:

- AmazonEKSConnectorServiceRolePolicy

Debe configurar permisos para permitir a una entidad de IAM (como un usuario, grupo o rol) crear, editar o eliminar un rol vinculado a servicios. Para obtener más información, consulte Permisos de roles vinculados a servicios en la Guía del usuario de IAM.

Este rol usa permisos de SSM (Systems Manager) para establecer conexiones seguras y administrar los clústeres de Kubernetes conectados.

Crear un rol vinculado a servicios para Amazon EKS

No necesita crear un rol vinculado a servicios de forma manual para conectar un clúster. Cuando conecta un clúster en la Consola de administración de AWS, la AWS CLI, `eksctl` o la API de AWS, Amazon EKS crea el rol vinculado a servicios en su nombre.

Si elimina este rol vinculado a servicios y necesita crearlo de nuevo, puede utilizar el mismo proceso para volver a crear el rol en su cuenta. Al conectar un clúster, Amazon EKS crea de nuevo el rol vinculado a servicios en su nombre.

Editar un rol vinculado a servicios para Amazon EKS

Amazon EKS no permite editar el rol vinculado a servicios de `AWSServiceRoleForAmazonEKSConnector`. Después de crear un rol vinculado al servicio, no podrá cambiar el nombre del rol, ya que varias entidades podrían hacer referencia al rol. Sin embargo, sí puede editar la descripción del rol con IAM. Para obtener más información, consulte Editing a service-linked role (Editar un rol vinculado a servicios) en la Guía del usuario de IAM.

Eliminar un rol vinculado a servicios para Amazon EKS

Si ya no necesita usar una característica o servicio que requieran un rol vinculado a un servicio, le recomendamos que elimine dicho rol. De esta forma, no tiene una entidad no utilizada que no se monitoree ni mantenga de forma activa. Sin embargo, debe limpiar el rol vinculado a servicios antes de eliminarlo manualmente.

Limpiar un rol vinculado a servicios

Antes de que pueda utilizar IAM para eliminar un rol vinculado a servicios, primero debe eliminar los recursos que utiliza el rol.

> **Note**
>
> Si el servicio de Amazon EKS utiliza el rol cuando intenta eliminar los recursos, la eliminación podría producir un error. En tal caso, espere unos minutos e intente de nuevo la operación.

1. Abra la consola de Amazon EKS.

2. En el panel de navegación izquierdo, elija Clusters (Clústeres).
3. En la página Clusters (Clústeres), seleccione el clúster.
4. Seleccione la pestaña Deregister (Anular registro) y, a continuación, la pestaña OK (Aceptar).

Eliminación manual de un rol vinculado a servicios

Utilice la consola de IAM, la AWS CLI o la API de AWS para eliminar el rol vinculado a servicios AWSServiceRoleForAmazonEKSConnector. Para obtener más información, consulte Eliminación de un rol vinculado a un servicio en la Guía del usuario de IAM.

Uso de roles para clústeres locales de Amazon EKS en Outpost

Amazon Elastic Kubernetes Service utiliza roles vinculados a servicios de AWS Identity and Access Management (IAM). Un rol vinculado a servicios es un tipo único de rol de IAM que se encuentra vinculado directamente a Amazon EKS. Los roles vinculados a servicios se encuentran predefinidos por Amazon EKS e incluyen todos los permisos que el servicio requiere para llamar a otros servicios de AWS en su nombre.

Un rol vinculado a servicios simplifica la configuración de Amazon EKS porque ya no tendrá que agregar de forma manual los permisos necesarios. Amazon EKS define los permisos de sus roles vinculados a servicios y, a menos que esté definido de otra manera, solo Amazon EKS puede asumir sus roles. Los permisos definidos incluyen las políticas de confianza y de permisos, y que la política de permisos no se pueda adjuntar a ninguna otra entidad de IAM.

Solo es posible eliminar un rol vinculado a un servicio después de eliminar sus recursos relacionados. De esta forma, se protegen los recursos de Amazon EKS, ya que se evita que se puedan eliminar accidentalmente permisos de acceso a los recursos.

Para obtener información sobre otros servicios que admiten roles vinculados al servicio, consulte Servicios de AWS que funcionan con IAM y busque los servicios que muestran Yes (Sí) en la columna Service-linked role (Rol vinculado al servicio). Seleccione una opción Sí con un enlace para ver la documentación acerca del rol vinculado al servicio en cuestión.

Permisos de roles vinculados a servicios para Amazon EKS

Amazon MSK usa el rol vinculado al servicio denominado `AWSServiceRoleForAmazonEKSLocalOutpost`. El rol de IAM permite a Amazon EKS administrar clústeres locales en la cuenta. Las políticas asociadas permiten que el rol administre los siguientes recursos: interfaces de red, grupos de seguridad, registros e instancias de Amazon EC2.

> **Note**
>
> El rol vinculado al servicio `AWSServiceRoleForAmazonEKSLocalOutpost` es distinto del rol requerido para la creación de clústeres. Para obtener más información, consulte the section called “Rol de IAM de clúster”.

El rol vinculado al servicio `AWSServiceRoleForAmazonEKSLocalOutpost` confía en los siguientes servicios para asumir el rol:

- `outposts.eks-local.amazonaws.com`

La política de permisos del rol permite que Amazon EKS realice las siguientes acciones en los recursos especificados:

- AmazonEKSServiceRolePolicy

Debe configurar permisos para permitir a una entidad de IAM (como un usuario, grupo o rol) crear, editar o eliminar un rol vinculado a servicios. Para obtener más información, consulte Permisos de roles vinculados a servicios en la Guía del usuario de IAM.

Crear un rol vinculado a servicios para Amazon EKS

No necesita crear un rol vinculado a servicios de manera manual. Cuando crea un clúster en la Consola de administración de AWS, la AWS CLI, o la API de AWS, Amazon EKS crea el rol vinculado a servicios en su nombre.

Si elimina este rol vinculado a servicios y necesita crearlo de nuevo, puede utilizar el mismo proceso para volver a crear el rol en su cuenta. Al crear un clúster, Amazon EKS se encarga de crear de nuevo el rol vinculado a servicios en su nombre.

Editar un rol vinculado a servicios para Amazon EKS

Amazon EKS no permite editar el rol vinculado a servicios de `AWSServiceRoleForAmazonEKSLocalOutpost`. Después de crear un rol vinculado a servicios, no puede cambiarle el nombre, ya que varias entidades pueden hacer referencia a él. Sin embargo, puede editar la descripción del rol mediante IAM. Para obtener más información, consulte Editing a service-linked role (Editar un rol vinculado a servicios) en la Guía del usuario de IAM.

Eliminar un rol vinculado a servicios para Amazon EKS

Si ya no necesita usar una característica o servicio que requieran un rol vinculado a un servicio, le recomendamos que elimine dicho rol. De esta forma, no tiene una entidad no utilizada que no se monitoree ni mantenga de forma activa. Sin embargo, debe limpiar el rol vinculado a servicios antes de eliminarlo manualmente.

Limpiar un rol vinculado a servicios

Antes de que pueda utilizar IAM para eliminar un rol vinculado a servicios, primero debe eliminar los recursos que utiliza el rol.

> **Note**
>
> Si el servicio de Amazon EKS utiliza el rol cuando intenta eliminar los recursos, la eliminación podría producir un error. En tal caso, espere unos minutos e intente de nuevo la operación.

1. Abra la consola de Amazon EKS.
2. En el panel de navegación izquierdo, elija Amazon EKS Clusters (Clústeres de Amazon EKS).
3. Si el clúster tiene grupos de nodos o perfiles de Fargate, debe eliminarlos para poder eliminar el clúster. Para obtener más información, consulte the section called “Eliminar” y the section called “Eliminar perfiles”.
4. En la página Clusters (Clústeres), elija el clúster que desea eliminar y elija Delete (Eliminar).
5. Escriba el nombre del clúster en la ventana de confirmación de eliminación y, a continuación, elija Delete (Eliminar).
6. Repita este procedimiento para el resto de los clústeres de la cuenta. Espere a que finalicen todas las operaciones de eliminación.

Eliminación manual de un rol vinculado a servicios

Utilice la consola de IAM, la CLI de AWS o la API de AWS para eliminar el rol vinculado al servicio de `AWSServiceRoleForAmazonEKSLocalOutpost`. Para obtener más información, consulte Eliminar un rol vinculado a un servicio en la Guía del usuario de IAM.

Regiones admitidas para los roles vinculados a servicios de Amazon EKS

Amazon EKS admite el uso de roles vinculados a servicios en todas las regiones en las que se encuentra disponible el servicio. Para obtener más información, consulte Cuotas y puntos de conexión de Amazon EKS.

Uso de roles para el panel de Amazon EKS

El panel de Amazon EKS utiliza este rol vinculado al servicio para agregar información sobre los recursos del clúster de varias regiones y cuentas. El panel se integra con AWS Organizations para leer de forma segura la información sobre varias cuentas de su organización.

Para obtener más información sobre el panel de Amazon EKS, consulte the section called “Panel de Amazon EKS”.

Introducción

Amazon EKS utiliza roles vinculados a servicios de AWS Identity and Access Management (IAM). Un rol vinculado a servicios es un tipo único de rol de IAM que se encuentra vinculado directamente a Amazon EKS. Los roles vinculados a servicios se encuentran predefinidos por Amazon EKS e incluyen todos los permisos que el servicio requiere para llamar a otros servicios de AWS en su nombre.

Un rol vinculado a servicios simplifica la configuración de Amazon EKS porque ya no tendrá que agregar de forma manual los permisos necesarios. Amazon EKS define los permisos de sus roles vinculados a servicios y, a menos que esté definido de otra manera, solo Amazon EKS puede asumir sus roles. Los permisos definidos incluyen las políticas de confianza y de permisos, y que la política de permisos no se pueda adjuntar a ninguna otra entidad de IAM.

Solo es posible eliminar un rol vinculado a un servicio después de eliminar sus recursos relacionados. De esta forma, se protegen los recursos de Amazon EKS, ya que se evita que se puedan eliminar accidentalmente permisos de acceso a los recursos.

Para obtener información sobre otros servicios que admiten roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM y busque los servicios que muestran Sí en la columna Rol vinculado a servicios. Seleccione una opción Sí con un enlace para ver la documentación acerca del rol vinculado al servicio en cuestión.

Permisos de roles vinculados a servicios para Amazon EKS

Amazon MSK usa el rol vinculado al servicio denominado `AWSServiceRoleForAmazonEKSDashboard`. El rol permite a Amazon EKS generar y mostrar el panel de EKS, incluida la información agregada sobre los recursos del clúster. La política asociada de `AmazonEKSDashboardServiceRolePolicy` permite al rol administrar los siguientes recursos: grupos de escalado automático, grupos de seguridad, plantillas de lanzamiento y perfiles de instancia de IAM. Para obtener más información, consulte the section called “Política administrada de AWS: AmazonEKSDashboardServiceRolePolicy”.

El rol vinculado al servicio `AWSServiceRoleForAmazonEKSDashboard` confía en los siguientes servicios para asumir el rol:

- `dashboard.eks.amazonaws.com`

Para consultar la versión más reciente del documento de política de JSON, consulte AmazonEKSDashboardServiceRolePolicy en la Guía de referencia de políticas administradas de AWS.

Debe configurar permisos para permitir a una entidad de IAM (como un usuario, grupo o rol) crear, editar o eliminar un rol vinculado a servicios. Para obtener más información, consulte Permisos de roles vinculados a servicios en la Guía del usuario de IAM.

Crear un rol vinculado a servicios para Amazon EKS

No necesita crear un rol vinculado a servicios de manera manual. Cuando activa el panel en la consola de AWS, Amazon EKS crea el rol vinculado al servicio.

Para obtener más información sobre cómo activar el panel de EKS, consulte the section called “Configuración de las organizaciones”.

> **Important**
>
> Este rol vinculado a servicios puede aparecer en su cuenta si se ha completado una acción en otro servicio que utilice las características compatibles con este rol.

Editar un rol vinculado a servicios para Amazon EKS

Amazon EKS no permite editar el rol vinculado a servicios de `AWSServiceRoleForAmazonEKSDashboard`. Después de crear un rol vinculado al servicio, no podrá cambiar el nombre del rol, ya que varias entidades podrían hacer referencia al rol. Sin embargo, sí puede editar la descripción del rol con IAM. Para obtener más información, consulte Editing a service-linked role (Editar un rol vinculado a servicios) en la Guía del usuario de IAM.

Eliminar un rol vinculado a servicios para Amazon EKS

Si ya no necesita usar una característica o servicio que requieran un rol vinculado a un servicio, le recomendamos que elimine dicho rol. De esta forma, no tiene una entidad no utilizada que no se monitoree ni mantenga de forma activa. Sin embargo, debe limpiar el rol vinculado a servicios antes de eliminarlo manualmente.

Limpiar un rol vinculado a servicios

Antes de que pueda utilizar IAM para eliminar un rol vinculado a servicios, primero debe eliminar los recursos que utiliza el rol.

> **Note**
>
> Si el servicio de Amazon EKS utiliza el rol cuando intenta eliminar los recursos, la eliminación podría producir un error. En tal caso, espere unos minutos e intente de nuevo la operación.

Para obtener más información sobre cómo desactivar el panel de EKS, consulte the section called “Configuración de las organizaciones”.

Eliminación manual de un rol vinculado a servicios

Utilice la consola de IAM, la CLI de AWS o la API de AWS para eliminar el rol vinculado al servicio de `AWSServiceRoleForAmazonEKSDashboard`. Para obtener más información, consulte Eliminar un rol vinculado a un servicio en la Guía del usuario de IAM.

Regiones admitidas para los roles vinculados a servicios de Amazon EKS

Amazon EKS admite el uso de roles vinculados a servicios en todas las regiones en las que se encuentra disponible el servicio. Para obtener más información, consulte Cuotas y puntos de conexión de Amazon EKS.

Rol de IAM de ejecución de pods de Amazon EKS

Se requiere el rol de ejecución de pods de Amazon EKS para ejecutar pods en la infraestructura de AWS Fargate.

Cuando su clúster crea pods en la infraestructura de AWS Fargate, los componentes que se ejecutan en la infraestructura de Fargate deben hacer llamadas a las API de AWS en su nombre. Es así para que puedan realizar acciones, como extraer imágenes de contenedores de Amazon ECR o enrutar registros a otros servicios de AWS. El rol de ejecución de pods de Amazon EKS proporciona los permisos de IAM para esta tarea.

Al crear un perfil de Fargate, debe especificar un rol de ejecución de pods para los componentes de Amazon EKS que se ejecutan en la infraestructura de Fargate con el perfil. Este rol se agrega al control de acceso basado en roles (RBAC) de Kubernetes del clúster para su autorización. Esto permite al `kubelet` que se está ejecutando en la infraestructura de Fargate registrarse en el clúster de Amazon EKS para que pueda aparecer en el clúster como un nodo.

> **Note**
>
> El perfil de Fargate debe tener un rol de IAM diferente a los grupos de nodos de Amazon EC2.

> **Important**
>
> Los contenedores que se ejecutan en el pod de Fargate no pueden asumir los permisos de IAM asociados a un rol de ejecución de pods. Para conceder permisos a los contenedores de su pod de Fargate a fin de acceder a otros servicios de AWS, debe utilizar Roles de IAM para cuentas de servicio.

Antes de crear un perfil de Fargate, debe crear un rol de IAM con la política AmazonEKSFargatePodExecutionRolePolicy.

Comprobación de la existencia de un rol de ejecución de pods configurado correctamente

Puede utilizar el siguiente procedimiento para verificar y ver si su cuenta ya dispone del rol de ejecución de pods de Amazon EKS correctamente configurado. Para evitar un problema

de seguridad adjunto confuso, es importante que la función restrinja el acceso basándose en `SourceArn`. Puede modificar el rol de ejecución según sea necesario para incluir compatibilidad con perfiles Fargate en otros clústeres.

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación izquierdo, elija Roles.
3. En la página Roles, busque la lista de roles para AmazonEKSFargatePodExecutionRole. Si el rol no existe, consulte the section called “Creación del rol de ejecución de pods de Amazon EKS” para crear el rol. Si el rol existe, elija el rol.
4. En la página AmazonEKSFargatePodExecutionRole, haga lo siguiente:
   a. Elija Permissions.
   b. Asegúrese de que la política AmazonEKSFargatePodExecutionRolePolicy administrada por Amazon esté asociada al rol.
   c. Seleccione Trust Relationships.
   d. Elija Edit trust policy (Editar la política de confianza).
5. En la página Editar la política de confianza, verifique que la relación de confianza contiene la siguiente política y que tenga una línea para los perfiles de Fargate en su clúster. Si es así, elija Cancel (Cancelar).

```
{
  "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"
    }
  ]
}
```

Si la política coincide pero no tiene una línea que especifique los perfiles de Fargate en su clúster, puede agregar la siguiente línea en la parte superior del objeto `ArnLike`. Reemplace *region-code* por la región de AWS donde se encuentra el clúster, *111122223333* por el ID de la cuenta y *my-cluster* por el nombre del clúster.

```
"aws:SourceArn": "arn:aws:eks:region-code:111122223333:fargateprofile/my-cluster/*",
```

Si la política no coincide, copie la política anterior completa en el formulario y elija Actualizar política. Reemplace *region-code* por la región de AWS en la que se encuentra el clúster. Si desea utilizar la misma función en todas las regiones de AWS en su cuenta, reemplace *region-code* con *. Reemplace *111122223333* con su ID de cuenta y *my-cluster* con el nombre de su clúster. Si quiere utilizar el mismo rol para todos los clústeres de su cuenta, reemplace *my-cluster* con *.

Creación del rol de ejecución de pods de Amazon EKS

Si aún no tiene el rol de ejecución de pods de Amazon EKS para su clúster, puede utilizar la Consola de administración de AWS o la AWS CLI para crearlo.

Consola de administración de AWS

a. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
b. En el panel de navegación izquierdo, elija Roles.
c. En la página Roles, elija Crear rol.
d. En la página Seleccionar entidad de confianza, haga lo siguiente:
    i. En la sección Tipo de entidad de confianza, elija Servicio de AWS.
    ii. En la lista desplegable Casos de uso para otros servicios de AWS, elija EKS.
    iii. Elija EKS - Pod de Fargate.
    iv. Elija Siguiente.
e. Elija Next (Siguiente) en la página Add permissions (Agregar permisos).
f. En la página Name, review, and create (Nombre, revisar y crear), haga lo siguiente:
    i. En Nombre del rol, ingrese un nombre único para su rol, por ejemplo, `AmazonEKSFargatePodExecutionRole`.

ii. En Agregar etiquetas (Opcional), de manera opcional, agregue metadatos al rol asociando etiquetas como pares de clave-valor. Para obtener más información sobre el uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM en la Guía de usuario de IAM.

iii. Seleccione Crear rol.

g. En la página Roles, busque la lista de roles para AmazonEKSFargatePodExecutionRole. Elija el rol.

h. En la página AmazonEKSFargatePodExecutionRole, haga lo siguiente:

i. Seleccione Trust Relationships.

ii. Elija Edit trust policy (Editar la política de confianza).

i. En la página Edit trust policy (Editar política de confianza), lleve a cabo las siguientes operaciones:

i. Copie y pegue los siguientes contenidos en el formulario Edit trust policy (Editar política de confianza). Reemplace *region-code* por la región de AWS en la que se encuentra el clúster. Si desea utilizar la misma función en todas las regiones de AWS en su cuenta, reemplace *region-code* con *. Reemplace *111122223333* con su ID de cuenta y *my-cluster* con el nombre de su clúster. Si quiere utilizar el mismo rol para todos los clústeres de su cuenta, reemplace *my-cluster* con *.

```
{
  "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. Elija Actualizar política.

AWS CLI

a. Copie y pegue los siguientes contenidos en un archivo denominado `pod-execution-role-trust-policy.json`. Reemplace *`region-code`* por la región de AWS en la que se encuentra el clúster. Si desea utilizar la misma función en todas las regiones de AWS en su cuenta, reemplace *`region-code`* con `*`. Reemplace *`111122223333`* con su ID de cuenta y *`my-cluster`* con el nombre de su clúster. Si quiere utilizar el mismo rol para todos los clústeres de su cuenta, reemplace *`my-cluster`* con `*`.

```
{
  "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. Cree un rol de IAM de ejecución de pods.

```
aws iam create-role \
  --role-name AmazonEKSFargatePodExecutionRole \
  --assume-role-policy-document file://"pod-execution-role-trust-policy.json"
```

c. Adjunte la política administrada de IAM por Amazon EKS requerida al rol.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSFargatePodExecutionRolePolicy \
  --role-name AmazonEKSFargatePodExecutionRole
```

Rol de IAM conector de Amazon EKS

Puede conectar clústeres de Kubernetes para verlos en la Consola de administración de AWS. Para conectarse a un clúster de Kubernetes, cree un rol de IAM.

Verificar si hay un rol de EKS Conector existente

Puede utilizar el siguiente procedimiento para verificar y ver si la cuenta ya dispone del rol conector de Amazon EKS.

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación izquierdo, elija Roles.
3. En la lista de roles, busque `AmazonEKSConnectorAgentRole`. Si no existe un rol que incluya `AmazonEKSConnectorAgentRole`, consulte the section called “Creación del rol de agente conector de Amazon EKS” para crearlo. Si existe un rol que incluye `AmazonEKSConnectorAgentRole`, seleccione el rol para ver las políticas asociadas.
4. Elija Permissions.
5. Asegúrese de que la política administrada AmazonEKSConnectorAgentPolicy se haya adjuntado al rol. Si la política se ha adjuntado, entonces el rol de Amazon EKS Connector se ha configurado correctamente.
6. Elija Relaciones de confianza y, a continuación, Editar política de confianza.
7. Verifique que la relación de confianza contiene la siguiente política. Si la relación de confianza coincide con la política a continuación, seleccione Cancelar. Si la relación de confianza no coincide, copie la política en la ventana Editar política de confianza y elija Actualizar política.

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

```
}
```

Creación del rol de agente conector de Amazon EKS

Puede utilizar la Consola de administración de AWS o AWS CloudFormation para crear un rol de agente conector.

AWS CLI

a. Cree un archivo con el nombre `eks-connector-agent-trust-policy.json`, que contenga el siguiente JSON que se va a utilizar para el rol de IAM.

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

b. Cree un archivo con el nombre `eks-connector-agent-policy.json` que contenga el siguiente JSON que se va a utilizar para el rol de IAM.

```
{
    "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. Cree el rol de agente de Amazon EKS Connector con la política de confianza y la política que creó en la lista de elementos anterior.

```
aws iam create-role \
     --role-name AmazonEKSConnectorAgentRole \
     --assume-role-policy-document file://eks-connector-agent-trust-policy.json
```

d. Adjunte la política a su rol de agente de Amazon EKS Connector.

```
aws iam put-role-policy \
     --role-name AmazonEKSConnectorAgentRole \
     --policy-name AmazonEKSConnectorAgentPolicy \
     --policy-document file://eks-connector-agent-policy.json
```

AWS CloudFormation

a. Guarde la siguiente plantilla de AWS CloudFormation en un archivo de texto en su sistema local.

> **Note**
>
> Esta plantilla también crea el rol vinculado al servicio que de otro modo se crearía cuando se llama a la API `registerCluster`. Para obtener más información, consulte the section called “Rol de conector de clúster”.

```
---
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. Abra la Consola de AWS CloudFormation.

c. Elija Crear pila con nuevos recursos (estándar).

d. Para Specify template (Especificar plantilla), seleccione Upload a template file (Actualizar un archivo de plantilla) y, a continuación, elija Choose file (Elegir archivo).
e. Elija el archivo que creó anteriormente y, a continuación, elija Next (Siguiente).
f. En Stack name (Nombre de pila), escriba un nombre para el rol, por ejemplo `eksConnectorAgentRole` y, a continuación, elija Next (Siguiente).
g. En la página Configurar opciones de pila, elija Siguiente.
h. En la página Review (Revisar), revise la información, confirme que la pila puede crear recursos de IAM y elija Create stack (Crear pila).

Políticas administradas de AWS para Amazon Elastic Kubernetes Service

Una política administrada de AWS es una política independiente que AWS crea y administra. Las políticas administradas de AWS se diseñan para ofrecer permisos para muchos casos de uso comunes, por lo que puede empezar a asignar permisos a los usuarios, grupos y roles.

Tenga en cuenta que es posible que las políticas administradas de AWS no concedan permisos de privilegio mínimo para los casos de uso concretos, ya que están disponibles para que las utilicen todos los clientes de AWS. Se recomienda reducir los permisos al definir políticas administradas por el cliente que sean específicas para sus casos de uso.

porNo puede cambiar los permisos definidos en las políticas administradas AWS. Si AWS actualiza los permisos definidos en un política administrada de AWS, la actualización afecta a todas las identidades principales (usuarios, grupos y roles) a las que está adjunta la política. Lo más probable es que AWS actualice una política administrada de AWS cuando se lance un nuevo servicio AWS o las operaciones de la API nuevas estén disponibles para los servicios existentes.

Para obtener más información, consulte Políticas administradas de AWS en la Guía del usuario de IAM.

Política administrada de AWS: AmazonEKS_CNI_Policy

No puede adjuntar la `AmazonEKS_CNI_Policy` a sus entidades de IAM. Antes de crear un grupo de nodos de Amazon EC2, esta política debe estar asociada al rol de IAM del nodo o a un rol de IAM utilizado de forma específica por el complemento CNI de Amazon VPC para Kubernetes. Esto es para que pueda realizar acciones en su nombre. Recomendamos que adjunte la política a un rol que solo utilice el complemento. Para obtener más información, consulte the section called “CNI de Amazon VPC” y the section called “Configuración para IRSA”.

Detalles sobre los permisos

Esta política incluye los siguientes permisos que permiten a Amazon EKS completar las siguientes tareas:

- **`ec2:*NetworkInterface`** y **`ec2:*PrivateIpAddresses`**: permite que el complemento CNI de Amazon VPC lleve a cabo acciones como el aprovisionamiento de interfaces de red elásticas y direcciones IP de los pods a fin de proporcionar redes para aplicaciones que se ejecutan en Amazon EKS.
- Acciones de lectura **`ec2`**: permite que el complemento CNI de Amazon VPC realice acciones como describir instancias y subredes para ver la cantidad de direcciones IP libres en las subredes de Amazon VPC. La CNI de la VPC puede usar las direcciones IP libres de cada subred para seleccionar las subredes con la mayor cantidad de direcciones IP libres y utilizarlas al crear una interfaz de red elástica.

Para ver la versión más reciente del documento de política JSON, consulte AmazonEKS_CNI_Policy en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEKSClusterPolicy

Puede adjuntar la `AmazonEKSClusterPolicy` a sus entidades de IAM. Antes de crear un clúster, debe tener un rol de IAM de clúster con esta política adjunta. Los clústeres de Kubernetes administrados por Amazon EKS realizan llamadas a otros servicios de AWS en su nombre. Lo hacen para administrar los recursos que utiliza con el servicio.

Esta política incluye los siguientes permisos que permiten a Amazon EKS completar las siguientes tareas:

- **`autoscaling`** : leer y actualizar la configuración de un grupo de escalado automático. Amazon EKS no utiliza estos permisos, pero permanecen en la política de compatibilidad con versiones anteriores.
- **`ec2`** : trabajar con volúmenes y recursos de red asociados a nodos de Amazon EC2. Esto es necesario para que el plano de control de Kubernetes pueda unir instancias a un clúster y aprovisionar y administrar de forma dinámica los volúmenes de Amazon EBS solicitados por los volúmenes persistentes de Kubernetes.
- **`ec2`** : eliminar las interfaces de red elásticas creadas por la CNI de la VPC. Esto es necesario para que EKS pueda limpiar las interfaces de red elásticas que se excluyen si la CNI de la VPC se cierra inesperadamente.

- **`elasticloadbalancing`** – trabaja con los Elastic Load Balancer y les agrega nodos como destinos. Esto es necesario para que el plano de control de Kubernetes pueda aprovisionar de forma dinámica los Elastic Load Balancer solicitados por los servicios de Kubernetes.
- **`iam`** – crea un rol vinculado a servicios. Esto es necesario para que el plano de control de Kubernetes pueda aprovisionar de forma dinámica los equilibradores de carga elásticos solicitados por los servicios de Kubernetes.
- **`kms`** : leer una clave de AWS KMS. Esto es necesario para que el plano de control de Kubernetes admita el cifrado de secretos de Kubernetes almacenados en `etcd`.

Para ver la versión más reciente del documento de política JSON, consulte AmazonEKSClusterPolicy en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEKSDashboardConsoleReadOnly

Puede adjuntar `AmazonEKSDashboardConsoleReadOnly` a sus entidades de IAM.

Esta política incluye los siguientes permisos que permiten a Amazon EKS completar las siguientes tareas:

- **`eks`**: acceso de solo lectura a los datos, recursos e información sobre las versiones de los clústeres del panel de control de EKS. Esto permite ver las métricas relacionadas con EKS y los detalles de configuración del clúster.
- **`organizations`**: acceso de solo lectura a la información de AWS Organizations, que incluye:
  - Ver detalles de la organización y acceso a servicios
  - Enumerar las raíces organizativas, las cuentas y las unidades organizativas
  - Ver la estructura de la organización

Para consultar la versión más reciente del documento de la política de JSON, consulte AmazonEKSDashboardServiceRolePolicy en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEKSFargatePodExecutionRolePolicy

Puede adjuntar la `AmazonEKSFargatePodExecutionRolePolicy` a sus entidades de IAM. Antes de crear un perfil de Fargate, debe crear un rol de ejecución de pod de Fargate y adjuntarle esta política. Para obtener más información, consulte the section called “Paso 2: creación de un rol de ejecución de pods de Fargate” y the section called “Definición de perfiles”.

Esta política concede al rol los permisos que proporcionan acceso a otros recursos de servicios de AWS necesarios para ejecutar pods de Amazon EKS en Fargate.

Detalles sobre los permisos

Esta política incluye los siguientes permisos que permiten a Amazon EKS completar las siguientes tareas:

- **`ecr`** – permite que los pods que se ejecutan en Fargate extraigan imágenes del contenedor almacenadas en Amazon ECR.

Para ver la versión más reciente del documento de política JSON, consulte AmazonEKSFargatePodExecutionRolePolicy en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEKSConnectorServiceRolePolicy

No puede asociar `AmazonEKSConnectorServiceRolePolicy` a sus entidades IAM. Esta política se encuentra adjunta a un rol vinculado a servicios que permite a Amazon EKS realizar acciones en su nombre. Para obtener más información, consulte the section called “Rol de conector de clúster”.

El rol permite a Amazon EKS conectar clústeres de Kubernetes. Las políticas adjuntas permiten que el rol administre los recursos necesarios para conectarse al clúster de Kubernetes registrado.

Detalles de los permisos

Esta política incluye los siguientes permisos que permiten a Amazon EKS completar las siguientes tareas.

- **`SSM Management`**: cree, describa y elimine activaciones de SSM y anule el registro de las instancias administradas. Esto permite las operaciones básicas de Systems Manager.
- **`Session Management`**: inicie sesiones de SSM específicamente para clústeres de EKS y ejecute comandos no interactivos mediante el documento AmazonEKS.
- **`IAM Role Passing`**: transfiera roles de IAM específicamente al servicio de SSM, controlado por una condición que restringe los roles transferidos a `ssm.amazonaws.com`.
- **`EventBridge Rules`**: cree reglas y destinos de EventBridge, pero solo cuando los administre `eks-connector.amazonaws.com`. Las reglas están limitadas específicamente para que AWS SSM sea el origen del evento.

Para consultar la versión más reciente del documento de política de JSON, consulte AmazonEKSConnectorServiceRolePolicy en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEKSForFargateServiceRolePolicy

No puede asociar `AmazonEKSForFargateServiceRolePolicy` a sus entidades IAM. Esta política se encuentra adjunta a un rol vinculado a servicios que permite a Amazon EKS realizar acciones en su nombre. Para obtener más información, consulte `AWSServiceRoleforAmazonEKSForFargate`.

Esta política concede los permisos necesarios a Amazon EKS para ejecutar tareas de Fargate. Solo se utiliza la política si cuenta con nodos de Fargate.

Detalles de los permisos

Esta política incluye los siguientes permisos que permiten a Amazon EKS completar las siguientes tareas.

- **`ec2`** – crea y elimina interfaces de red elásticas y describe los recursos y las interfaces de red elásticas. Esto es necesario a fin de que el servicio Fargate de Amazon EKS pueda configurar las redes VPC necesarias para los pods de Fargate.

Para ver la versión más reciente del documento de política JSON, consulte AmazonEKSForFargateServiceRolePolicy en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEKSComputePolicy

Puede adjuntar `AmazonEKSComputePolicy` a sus entidades de IAM. Puede adjuntar esta política al rol de IAM de su clúster para ampliar los recursos que Amazon EKS puede administrar en su cuenta.

Esta política concede los permisos necesarios para que Amazon EKS cree y administre instancias de EC2 para el clúster de EKS, y los permisos de IAM necesarios para configurar EC2. Además, esta política otorga permisos para que Amazon EKS cree el rol vinculado al servicio EC2 Spot en su nombre.

Detalles sobre los permisos

Esta política incluye los siguientes permisos que permiten a Amazon EKS completar las siguientes tareas:

- **`ec2`** Permisos de:
  - `ec2:CreateFleet` y `ec2:RunInstances`: permite crear instancias de EC2 y utilizar recursos específicos de EC2 (imágenes, grupos de seguridad, subredes) para nodos de clúster de EKS.
  - `ec2:CreateLaunchTemplate`: permite crear plantillas de lanzamiento de EC2 para los nodos de clústeres de EKS.
  - La política también incluye condiciones para limitar el uso de estos permisos EC2 únicamente para los recursos etiquetados con el nombre del clúster de EKS y otras etiquetas relevantes.
  - `ec2:CreateTags`: permite agregar etiquetas a los recursos de EC2 creados por las acciones `CreateFleet`, `RunInstances` y `CreateLaunchTemplate`.
- **`iam`** Permisos de:
  - `iam:AddRoleToInstanceProfile`: permite agregar un rol de IAM al perfil de instancia de computación de EKS.
  - `iam:PassRole`: permite transmitir los roles de IAM necesarios al servicio de EC2.

Para ver la versión más reciente del documento de política de JSON, consulte AmazonEKSComputePolicy en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEKSNetworkingPolicy

Puede adjuntar `AmazonEKSNetworkingPolicy` a sus entidades de IAM. Puede adjuntar esta política al rol de IAM de su clúster para ampliar los recursos que Amazon EKS puede administrar en su cuenta.

Esta política está diseñada para conceder los permisos necesarios para que Amazon EKS cree y administre las interfaces de red para el clúster de Amazon EKS, lo que permite que el plano de control y los nodos de trabajo se comuniquen y funcionen correctamente.

Detalles sobre los permisos

Esta política concede los siguientes permisos para permitir que Amazon EKS administre las interfaces de red del clúster:

- **`ec2`** Permisos de interfaces de red de:
  - `ec2:CreateNetworkInterface`: permite crear interfaces de red de EC2.
  - La política incluye condiciones para restringir el uso de este permiso a las interfaces de red etiquetadas con el nombre del clúster de EKS y el nombre del nodo de CNI de Kubernetes.

- `ec2:CreateTags`: permite añadir etiquetas a las interfaces de red creadas por la acción `CreateNetworkInterface`.

- **`ec2`** Permisos de administración de la interfaz de red de:
  - `ec2:AttachNetworkInterface`, `ec2:DetachNetworkInterface`: permiten asociar y desconectar interfaces de red a instancias de EC2.
  - `ec2:UnassignPrivateIpAddresses`, `ec2:UnassignIpv6Addresses`, `ec2:AssignPrivateIpAddresses`, `ec2:AssignIpv6Addresses`: permiten administrar las asignaciones de dirección IP de las interfaces de red.
  - Estos permisos están restringidos a las interfaces de red etiquetadas con el nombre del clúster de EKS.

Para ver la versión más reciente del documento de política JSON,
consulte AmazonEKSNetworkingPolicy en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEKSBlockStoragePolicy

Puede adjuntar `AmazonEKSBlockStoragePolicy` a sus entidades de IAM. Puede adjuntar esta política al rol de IAM de su clúster para ampliar los recursos que Amazon EKS puede administrar en su cuenta.

Esta política concede los permisos necesarios para que Amazon EKS cree, administre y mantenga volúmenes de EC2 e instantáneas para el clúster de EKS, lo que permite que el plano de control y los nodos de trabajo aprovisionen y utilicen almacenamiento persistente según lo requieran las cargas de trabajo de Kubernetes.

Detalles sobre los permisos

Esta política de IAM concede los siguientes permisos para permitir que Amazon EKS administre volúmenes e instantáneas de EC2:

- **`ec2`** Permisos de administración de volúmenes:
  - `ec2:AttachVolume`, `ec2:DetachVolume`, `ec2:ModifyVolume`, `ec2:EnableFastSnapshotRestores`: permite asociar, desasociar, modificar y habilitar restauraciones rápidas de instantáneas para volúmenes de EC2.
  - Estos permisos están restringidos a los volúmenes etiquetados con el nombre del clúster de EKS.

  - `ec2:CreateTags`: permite agregar etiquetas a los volúmenes de EC2 y a las instantáneas creadas por las acciones `CreateSnapshot` y `CreateVolume`.
- **`ec2`** Permisos de creación de volúmenes:
  - `ec2:CreateVolume`: permite crear nuevos volúmenes de EC2.
  - La política incluye condiciones para restringir el uso de este permiso a volúmenes etiquetados con el nombre del clúster de EKS y otras etiquetas relevantes.
  - `ec2:CreateSnapshot`: permite crear nuevas instantáneas de volúmenes de EC2.
  - La política incluye condiciones para restringir el uso de este permiso a las instantáneas etiquetadas con el nombre del clúster de EKS y otras etiquetas relevantes.

Para consultar la versión más reciente del documento de política de JSON, consulte AmazonEKSBlockStoragePolicy en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEKSLoadBalancingPolicy

Puede adjuntar `AmazonEKSLoadBalancingPolicy` a sus entidades de IAM. Puede adjuntar esta política al rol de IAM de su clúster para ampliar los recursos que Amazon EKS puede administrar en su cuenta.

Esta política de IAM concede los permisos necesarios para que Amazon EKS trabaje con varios servicios de AWS para administrar los equilibradores de carga elásticos (ELB) y los recursos relacionados.

Detalles sobre los permisos

Los permisos clave concedidos por esta política son:

- **`elasticloadbalancing`** : permite crear, modificar y administrar equilibradores de carga elásticos y grupos de destino. Esto incluye permisos para crear, actualizar y eliminar equilibradores de carga, grupos de destino, agentes de escucha y reglas.
- **`ec2`** : permite crear y administrar grupos de seguridad necesarios para que el plano de control de Kubernetes pueda unir instancias a un clúster y administrar los volúmenes de Amazon EBS. Además, permite describir y enumerar los recursos de EC2, como instancias, VPC, subredes, grupos de seguridad y otros recursos de red.
- **`iam`** : permite crear un rol vinculado al servicio para Elastic Load Balancing, que es necesario para que el plano de control de Kubernetes aprovisione los equilibradores de carga elásticos de forma dinámica.

- **`kms`**: permite leer una clave de AWS KMS, que es necesaria para que el plano de control de Kubernetes admita el cifrado de los secretos de Kubernetes almacenados en etcd.
- **`wafv2`** y **`shield`** : permite asociar y desasociar ACL web y crear/eliminar protecciones de AWS Shield para los equilibradores de carga elásticos.
- **`cognito-idp`** , **`acm`** , y **`elasticloadbalancing`** : concede permisos para describir clientes de grupos de usuarios, enumerar y describir certificados y describir grupos de destino, que son necesarios para que el plano de control de Kubernetes administre los equilibradores de carga elásticos.

La política también incluye varias comprobaciones de condiciones para garantizar que los permisos se circunscriben al clúster de EKS específico que se administra, mediante la etiqueta `eks:eks-cluster-name`.

Para consultar la versión más reciente del documento de política JSON, consulte AmazonEKSLoadBalancingPolicy en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEKSMCPReadOnlyAccess

Puede adjuntar `AmazonEKSMCPReadOnlyAccess` a sus entidades de IAM. Esta política proporciona acceso de solo lectura a los recursos de Amazon EKS y los servicios de AWS relacionados, lo que permite al servidor Protocolo de contexto para modelos (MCP) de Amazon EKS llevar a cabo operaciones de observabilidad y solución de problemas sin hacer ninguna modificación en la infraestructura.

Detalles de los permisos

Esta política incluye los siguientes permisos que permiten a las entidades principales completar las siguientes tareas:

- **`eks`** permite a las entidades principales describir y enumerar los clústeres, los grupos de nodos, los complementos, las entradas de acceso y la información de EKS, así como acceder a la API de Kubernetes para operaciones de solo lectura.
- **`iam`** permite a las entidades principales recuperar información sobre las políticas y los roles de IAM y sus archivos adjuntos para comprender los permisos asociados a los recursos de EKS.
- **`ec2`** permite a las entidades principales describir las VPC, las subredes y las tablas de enrutamiento para comprender la configuración de red de los clústeres de EKS.
- **`sts`** permite a las entidades principales recuperar la información de identidad del autor de la llamada con fines de autenticación y autorización.

- **`logs`** permite a las entidades principales iniciar consultas y recuperar resultados de las consultas de Registros de CloudWatch para la solución de problemas y la supervisión.
- **`cloudwatch`** permite a las entidades principales recuperar datos de métricas para supervisar el rendimiento de los clústeres y las cargas de trabajo.
- **`eks-mcp`** permite a las entidades principales invocar operaciones de MCP y llamar a herramientas de solo lectura dentro del servidor MCP de Amazon EKS.

Para consultar la versión más reciente del documento de la política de JSON, consulte AmazonEKSMCPReadOnlyAccess en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEKSServicePolicy

Puede adjuntar `AmazonEKSServicePolicy` a sus entidades de IAM. Los clústeres creados antes del 16 de abril de 2020 requerían que creara un rol de IAM y le adjuntara esta política. Los clústeres creados a partir del 16 de abril de 2020 no requieren que cree un rol ni que asigne esta política. Cuando crea un clúster mediante una entidad principal de IAM que tiene el permiso `iam:CreateServiceLinkedRole`, el rol vinculado a servicios AWSServiceRoleforAmazonEKS se crea de forma automática. El rol vinculado a servicios trae adjunta la política administrada AmazonEKSServiceRolePolicy.

Esta política permite a Amazon EKS crear y administrar los recursos necesarios para operar clústeres de Amazon EKS.

Detalles de los permisos

Esta política incluye los siguientes permisos que permiten a Amazon EKS completar las siguientes tareas.

- **`eks`** – actualiza la versión de Kubernetes de su clúster después de iniciar una actualización. Amazon EKS no utiliza este permiso, pero permanece en la política de compatibilidad con versiones anteriores.
- **`ec2`** – trabaja con interfaces de red elásticas y otros recursos y etiquetas de red. Amazon EKS lo requiere para configurar redes que faciliten la comunicación entre nodos y el plano de control de Kubernetes. Obtenga información sobre los grupos de seguridad. Actualice las etiquetas de los grupos de seguridad.
- **`route53`** – asocia una VPC con una zona alojada. Amazon EKS lo requiere a fin de habilitar las redes privadas de punto de conexión para su servidor de API de clúster de Kubernetes.

- **`logs`** – registra eventos. Esto es necesario para que Amazon EKS pueda enviar registros de planos de control de Kubernetes a CloudWatch.
- **`iam`** – crea un rol vinculado a servicios. Esto es necesario para que Amazon EKS can cree el rol vinculado al servicio the section called “Permisos de roles vinculados a servicios para Amazon EKS” en su nombre.

Para ver la versión más reciente del documento de política JSON, consulte AmazonEKSServicePolicy en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEKSServiceRolePolicy

No puede asociar `AmazonEKSServiceRolePolicy` a sus entidades IAM. Esta política se encuentra adjunta a un rol vinculado a servicios que permite a Amazon EKS realizar acciones en su nombre. Para obtener más información, consulte the section called “Permisos de roles vinculados a servicios para Amazon EKS”. Al crear un clúster mediante una entidad principal de IAM que tiene el permiso `iam:CreateServiceLinkedRole`, el rol vinculado a servicios AWSServiceRoleforAmazonEKS se crea de forma automática en su nombre y se le asocia esta política.

Esta política permite al rol vinculado a servicios llamar a servicios de AWS en su nombre.

Detalles de los permisos

Esta política incluye los siguientes permisos que permiten a Amazon EKS completar las siguientes tareas.

- **`ec2`** – crea y describe las interfaces de red elásticas y las instancias de Amazon EC2, el grupo de seguridad de clúster y la VPC necesarios para la creación de clústeres. Para obtener más información, consulte the section called “Requisitos del grupo de seguridad”. Obtenga información sobre los grupos de seguridad. Actualice las etiquetas de los grupos de seguridad. Lea información sobre las reservas de capacidad bajo demanda. Lea la configuración de la VPC, que incluye las tablas de enrutamiento y las ACL de red, para detectar problemas de configuración como parte de la información sobre el clúster.
- Modo automático de **`ec2`**: termine las instancias de EC2 creadas por el modo automático de EKS. Para obtener más información, consulte Modo automático de EKS.
- **`iam`** – enumera todas las políticas administradas que se asocian a un rol de IAM. Esto es necesario para que Amazon EKS pueda enumerar y validar todos los permisos y políticas administrados necesarios para crear un clúster.

- Asocie un VPC con una zona alojada: Amazon EKS lo requiere a fin de habilitar las redes privadas de punto de conexión para su servidor de API de clúster de Kubernetes.
- Evento de registro: esto es necesario para que Amazon EKS pueda enviar registros de planos de control de Kubernetes a CloudWatch.
- Métrica Put: se necesita para que Amazon EKS pueda enviar registros del plano de control de Kubernetes a CloudWatch.
- **`eks`** - administre las entradas y políticas de acceso al clúster, que permiten ejercer un control detallado sobre quién puede acceder a los recursos del EKS y las acciones pueden realizar. Esto incluye la asociación de políticas de acceso estándar para operaciones de computación, redes, equilibrio de carga y almacenamiento.
- **`elasticloadbalancing`** - cree, administre y elimine los equilibradores de carga y sus componentes (agentes de escucha, grupos de destino, certificados) asociados a los clústeres de EKS. Consulte los atributos y el estado del equilibrador de carga.
- **`events`**: cree y administre reglas de EventBridge para supervisar eventos de EC2 y AWS Health relacionados con clústeres de EKS, lo que permite dar una respuesta automatizada a los cambios en la infraestructura y a las alertas de estado.
- **`iam`** - Administre los perfiles de instancia de EC2 con el prefijo “eks”, incluidas la creación, la eliminación y la asociación de roles, lo cual es necesario para la administración de nodos de EKS.
- **`pricing shield`**: acceda a la información de precios de AWS y al estado de protección de Shield, lo que facilita la administración de costos y el acceso a características avanzadas de seguridad para los recursos de EKS.
- Limpieza de recursos: elimine de forma segura los recursos etiquetados con EKS, incluidos volúmenes, instantáneas, plantillas de lanzamiento e interfaces de red durante las operaciones de limpieza del clúster.

Para ver la versión más reciente del documento de política JSON, consulte AmazonEKSServiceRolePolicy en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEKSVPCResourceController

Puede asociar la política `AmazonEKSVPCResourceController` a las identidades de IAM. Si utiliza grupos de seguridad de pods, debe asociar esta política a su rol de IAM del clúster de Amazon EKS para realizar acciones en su nombre.

Esta política concede permisos al rol de clúster para administrar las interfaces de red elásticas y las direcciones IP de los nodos.

Detalles sobre los permisos

Esta política incluye los siguientes permisos que permiten a Amazon EKS completar las siguientes tareas:

- **`ec2`** – administra interfaces de red elásticas y direcciones IP para admitir grupos de seguridad de pods y nodos de Windows.

Para ver la versión más reciente del documento de política JSON, consulte AmazonEKSVPCResourceController en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEKSWorkerNodePolicy

No puede adjuntar la `AmazonEKSWorkerNodePolicy` a sus entidades de IAM. Debe asociar esta política a un rol de IAM de nodo que especifique al crear nodos de Amazon EC2 que permiten a Amazon EKS realizar acciones en su nombre. Si crea un grupo de nodos con `eksctl`, crea el rol de IAM de nodo y adjunta esta política al rol de forma automática.

Esta política concede permisos a los nodos de Amazon EC2 de Amazon EKS para conectarse a los clústeres de Amazon EKS.

Detalles sobre los permisos

Esta política incluye los siguientes permisos que permiten a Amazon EKS completar las siguientes tareas:

- **`ec2`** – lee el volumen de la instancia y la información de red. Esto es necesario para que los nodos de Kubernetes puedan describir información sobre los recursos de Amazon EC2 necesarios a fin de que el nodo se una al clúster de Amazon EKS.
- **`eks`** – opcionalmente, describe el clúster como parte del arranque de nodos.
- **`eks-auth:AssumeRoleForPodIdentity`** – permite la recuperación de credenciales para las cargas de trabajo de EKS en el nodo. Esto es necesario para que la Pod Identity de EKS funcione correctamente.

Para ver la versión más reciente del documento de política JSON, consulte AmazonEKSWorkerNodePolicy en la Guía de referencia de políticas administradas de AWS.

Política administrada por AWS: AmazonEKSWorkerNodeMinimalPolicy

Puede vincular la política AmazonEKSWorkerNodeMinimalPolicy a sus entidades de IAM. Puede asociar esta política a un rol de IAM de nodo que especifique al crear nodos de Amazon EC2 que permiten a Amazon EKS realizar acciones en su nombre.

Esta política concede permisos a los nodos de Amazon EC2 de Amazon EKS para conectarse a los clústeres de Amazon EKS. Esta política tiene menos permisos en comparación con la política AmazonEKSWorkerNodePolicy.

Detalles sobre los permisos

Esta política incluye los siguientes permisos que permiten a Amazon EKS completar las siguientes tareas:

- `eks-auth:AssumeRoleForPodIdentity`: permite la recuperación de credenciales para las cargas de trabajo de EKS en el nodo. Esto es necesario para que la Pod Identity de EKS funcione correctamente.

Para ver la versión más reciente del documento de política JSON, consulte AmazonEKSWorkerNodePolicy en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AWSServiceRoleForAmazonEKSNodegroup

No puede asociar `AWSServiceRoleForAmazonEKSNodegroup` a sus entidades IAM. Esta política se encuentra adjunta a un rol vinculado a servicios que permite a Amazon EKS realizar acciones en su nombre. Para obtener más información, consulte the section called “Permisos de roles vinculados a servicios para Amazon EKS”.

Esta política otorga permisos de rol de `AWSServiceRoleForAmazonEKSNodegroup` que permiten crear y administrar grupos de nodos de Amazon EC2 en su cuenta.

Detalles sobre los permisos

Esta política incluye los siguientes permisos que permiten a Amazon EKS completar las siguientes tareas:

- **`ec2`** – trabaja con grupos de seguridad, etiquetas, reservas de capacidad y plantillas de lanzamiento. Esto es necesario para que los grupos de nodos administrados por Amazon

EKS habiliten la configuración de acceso remoto y describan las reservas de capacidad que se pueden usar en grupos de nodos administrados por Amazon EKS. Además, los grupos de nodos administrados por Amazon EKS crean una plantilla de lanzamiento en su nombre. Esto es para configurar el grupo de Amazon EC2 Auto Scaling que respalda cada grupo de nodos administrados.

- **`iam`** – crea un rol vinculado a servicios y transfiere un rol. Los grupos de nodos administrados por Amazon EKS lo requieren para administrar los perfiles de instancias del rol que se transfiere al crear un grupo de nodos administrados. Las instancias de Amazon EC2 lanzadas como parte de un grupo de nodos administrados utilizan este perfil de instancias. Amazon EKS necesita crear roles vinculadas a servicios para otros servicios, como los grupos de Amazon EC2 Auto Scaling. Estos permisos se usan en la creación de un grupo de nodos administrados.
- **`autoscaling`** – trabaja con grupos de escalado automático de seguridad. Los grupos de nodos administrados por Amazon EKS lo requieren para administrar el grupo de Amazon EC2 Auto Scaling que respalda cada grupo de nodos administrados. También se utiliza para admitir funciones como expulsar pods cuando los nodos se terminan o reciclan durante las actualizaciones de grupos de nodos.

Para ver la versión más reciente del documento de política JSON, consulte AWSServiceRoleForAmazonEKSNodegroup en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEKSDashboardServiceRolePolicy

No puede asociar `AmazonEKSDashboardServiceRolePolicy` a sus entidades IAM. Esta política se encuentra adjunta a un rol vinculado a servicios que permite a Amazon EKS realizar acciones en su nombre. Para obtener más información, consulte the section called “Permisos de roles vinculados a servicios para Amazon EKS”.

Esta política otorga permisos de rol de `AWSServiceRoleForAmazonEKSDashboard` que permiten crear y administrar grupos de nodos de Amazon EC2 en su cuenta.

Detalles sobre los permisos

Esta política incluye los siguientes permisos que permiten el acceso para completar las siguientes tareas:

- **`organizations`**: vea información sobre la estructura y las cuentas de AWS Organizations. Esto incluye permisos para enumerar las cuentas de su organización, ver las unidades organizativas

y las raíces, enumerar los administradores delegados, ver los servicios que tienen acceso a su organización y recuperar información detallada sobre su organización y sus cuentas.

Para consultar la versión más reciente del documento de política de JSON, consulte AmazonEKSDashboardServiceRolePolicy en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEBSCSIDriverPolicy

La política `AmazonEBSCSIDriverPolicy` permite que el controlador Interfaz de almacenamiento de contenedores (CSI) de Amazon EBS cree, modifique, copie, asocie, desasocie y elimine volúmenes en su nombre. Esto incluye modificar las etiquetas de los volúmenes existentes y habilitar la restauración rápida de instantáneas (FSR) en los volúmenes de EBS. También concede al controlador CSI de EBS permisos para crear, restaurar y eliminar instantáneas, así como para enumerar instancias, volúmenes e instantáneas.

Para ver la versión más reciente del documento de política JSON, consulte AmazonEBSCSIDriverServiceRolePolicy en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEFSCSIDriverPolicy

La política de `AmazonEFSCSIDriverPolicy` permite a la interfaz de almacenamiento de contenedores (CSI) de Amazon EFS crear y eliminar puntos de acceso en su nombre. También otorga permisos al controlador CSI de Amazon EFS para enumerar sus puntos de acceso, sistemas de archivos, destinos de montaje y zonas de disponibilidad de Amazon EC2.

Para ver la versión más reciente del documento de política JSON, consulte AmazonEFSCSIDriverServiceRolePolicy en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEKSLocalOutpostClusterPolicy

Puede asociar esta política a entidades de IAM. Antes de crear un clúster local, debe adjuntar esta política al rol del clúster. Los clústeres de Kubernetes administrados por Amazon EKS realizan llamadas a otros servicios de AWS en su nombre. Lo hacen para administrar los recursos que utiliza con el servicio.

La `AmazonEKSLocalOutpostClusterPolicy` incluye los siguientes permisos:

- Acciones de lectura de **ec2**: permite a las instancias del plano de control describir las propiedades de la zona de disponibilidad, la tabla de enrutamiento, la instancia y la interfaz de red. Permisos necesarios para que las instancias de Amazon EC2 se unan correctamente al clúster como instancias del plano de control.
- **ssm** – permite la conexión de Amazon EC2 Systems Manager a la instancia del plano de control, que Amazon EKS usa para comunicar y administrar el clúster local de su cuenta.
- **logs** – permite que las instancias envíen registros a Amazon CloudWatch.
- **secretsmanager** : permite a las instancias obtener y eliminar los datos de arranque de las instancias del plano de control de forma segura desde AWS Secrets Manager.
- **ecr** – permite que los pods y los contenedores que se ejecutan en las instancias del plano de control extraigan imágenes de contenedor almacenadas en Amazon Elastic Container Registry.

Para ver la versión más reciente del documento de política JSON, consulte AmazonEKSLocalOutpostClusterPolicy en la Guía de referencia de políticas administradas de AWS.

Política administrada de AWS: AmazonEKSLocalOutpostServiceRolePolicy

No puede adjuntar esta política a sus entidades de IAM. Al crear un clúster mediante una entidad principal de IAM que tiene el permiso `iam:CreateServiceLinkedRole`, Amazon EKS crea el rol vinculado a servicios AWSServiceRoleforAmazonEKSLocalOutpost de forma automática en su nombre y le asocia esta política. Esta política permite al rol vinculado a servicios llamar a servicios de AWS para clústeres locales en su nombre.

La `AmazonEKSLocalOutpostServiceRolePolicy` incluye los siguientes permisos:

- **ec2** – permite a Amazon EKS trabajar con la seguridad, la red y otros recursos para lanzar y administrar correctamente las instancias del plano de control en su cuenta.
- **ssm**, **ssmmessages** – permite la conexión de Amazon EC2 Systems Manager a las instancias del plano de control, que Amazon EKS usa para comunicar y administrar el clúster local de su cuenta.
- **iam** – permite a Amazon EKS administrar el perfil de instancia asociado a las instancias del plano de control.
- **secretsmanager** : permite a Amazon EKS colocar los datos de arranque de las instancias del plano de control en AWS Secrets Manager para que pueda consultarse de forma segura durante el arranque de la instancia.
- **outposts** – permite a Amazon EKS obtener información de Outpost de su cuenta para lanzar correctamente un clúster local en un Outpost.

Para ver la versión más reciente del documento de política JSON, consulte AmazonEKSLocalOutpostServiceRolePolicy en la Guía de referencia de políticas administradas de AWS.

Actualizaciones de Amazon EKS en las políticas administradas de AWS

Es posible consultar los detalles sobre las actualizaciones de las políticas administradas de AWS para Amazon EKS debido a que este servicio comenzó a realizar el seguimiento de estos cambios.

Para recibir notificaciones de todos los cambios en el archivo de origen de esta página de documentación específica, puede suscribirse a la siguiente URL con un lector de RSS:

```
https://github.com/awsdocs/amazon-eks-user-guide/commits/mainline/latest/ug/security/iam-reference/security-iam-awsmanpol.adoc.atom
```

Cambio	Descripción	Fecha
Presentó la política the section called “Política administrada de AWS: AmazonEKSMCPReadOnlyAccess”.	Amazon EKS introdujo una nueva política administrada de `AmazonEKSMCPReadOnlyAccess` para activar herramientas de solo lectura en el servidor MCP de Amazon EKS para la observabilidad y la solución de problemas.	21 de noviembre de 2025
Se agregaron permisos a AmazonEBSCSIDriverPolicy.	Se agregó el permiso `ec2:CopyVolumes` para permitir que el controlador CSI de EBS copie los volúmenes de EBS directamente.	17 de noviembre de 2025
Se agregó el permiso a the section called “Política administrada de AWS: AmazonEKSServiceRolePolicy”.	Se agregaron los permisos `ec2:DescribeRouteTables` y `ec2:DescribeNetworkAcls` a `AmazonEKSServiceRo`	22 de octubre de 2025

Cambio	Descripción	Fecha
	`lePolicy` . De este modo, Amazon EKS puede detectar problemas de configuración con las tablas de enrutamiento de VPC y las ACL de red para nodos híbridos como parte de la información de los clústeres .	
Se agregó el permiso a AWSServiceRoleForAmazonEKSConnector.	Se agregó el permiso `ssmmessages:OpenDataChannel` a `AmazonEKSConnectorServiceRolePolicy` .	15 de octubre de 2025
Se agregó el permiso a the section called “Política administrada de AWS: AmazonEKSServiceRolePolicy”	Este rol puede adjuntar la nueva política de acceso `AmazonEKSEventPolicy` . Permisos restringidos para `ec2:DeleteLaunchTemplate` y `ec2:TerminateInstances` .	26 de agosto de 2025
Permiso agregado a the section called “Política administrada de AWS: AmazonEKSLocalOutpostServiceRolePolicy”	Se agregó el permiso `ssmmessages:OpenDataChannel` a `AmazonEKSLocalOutpostServiceRolePolicy` .	26 de junio de 2025

Cambio	Descripción	Fecha
Se agregó el permiso a the section called “Política administrada de AWS: AmazonEKSComputePolicy”.	Se actualizaron los permisos de recursos para las acciones `ec2:RunInstances` y `ec2:CreateFleet` a fin de incluir las reservas de capacidad `arn:aws:ec2:*:*:capacity-reservation/*` . Esto permite que el modo automático de Amazon EKS ejecute instancias mediante las reservas de capacidad bajo demanda de EC2 en su cuenta. Se agregó `iam:CreateServiceLinkedRole` para permitir que el modo automático de Amazon EKS cree el rol `AWSServiceRoleForEC2Spot` vinculado al servicio EC2 Spot en su nombre.	20 de junio de 2025
Se agregó un permiso a AmazonEKSServiceRolePolicy.	Se agregó el permiso `ec2:DescribeCapacityReservations` para permitir que el modo automático de Amazon EKS ejecute instancias mediante las reservas de capacidad bajo demanda de EC2 en su cuenta.	20 de junio de 2025

Cambio	Descripción	Fecha
Presentó la política the section called “Política administrada de AWS: AmazonEKSDashboardConsoleReadOnly”.	Se introdujo una nueva política `AmazonEKSDashboardConsoleReadOnly` .	19 de junio de 2025
Presentó la política the section called “Política administrada de AWS: AmazonEKSDashboardServiceRolePolicy”.	Se introdujo una nueva política `AmazonEKSDashboardServiceRolePolicy` .	21 de mayo de 2025
Se agregaron permisos a AmazonEKSClusterPolicy.	Se agregó el permiso `ec2:DeleteNetworkInterfaces` para permitir que Amazon EKS elimine las interfaces de red elásticas que se excluyen si la CNI de la VPC se cierra inesperadamente.	16 de abril de 2025
Se agregó un permiso a AmazonEKSServiceRolePolicy.	Como parte de la versión 1.33 de EKS, se agregaron los permisos `ec2:RevokeSecurityGroupEgress` y `ec2:AuthorizeSecurityGroupEgress` para que los clientes de IA/ML de EKS puedan añadir reglas de salida de grupos de seguridad al clúster SG de EKS predeterminado que sean compatibles con EFA.	14 de abril de 2025

Cambio	Descripción	Fecha
Se agregaron permisos a AmazonEKSServiceRolePolicy.	Se agregó el permiso para terminar instancias de EC2 creadas por el modo automático de EKS.	28 de febrero de 2025

Cambio	Descripción	Fecha
Se agregaron permisos a AmazonEBSCSIDriverPolicy.	Se agregó una nueva declaración que autoriza al controlador CSI de EBS a restaurar todas las instantán eas. La política vigente permitía esto anteriormente, pero se requiere una nueva declaración explícita debido a un cambio en la gestión de IAM para `CreateVolume` . Se concedió al controlador CSI de EBS la capacidad de modificar las etiquetas de los volúmenes existentes. El controlador de CSI de EBS puede modificar las etiquetas de los volúmenes existente s a través de parámetros en VolumeAttributesClasses de Kubernetes. Se concedió al controlador CSI de EBS la capacidad de habilitar la restauración rápida de instantáneas (FSR) en los volúmenes de EBS. El controlador de CSI de EBS puede habilitar la FSR en volúmenes nuevos mediante parámetros en clases de almacenamiento de Kubernete s.	13 de enero de 2025

Cambio	Descripción	Fecha
Se agregaron permisos a the section called “Política administrada de AWS: AmazonEKSLoadBalancingPolicy”.	Se actualizó `AmazonEKSLoadBalancingPolicy` para permitir la enumeración y descripción de los recursos de redes y direcciones IP.	26 de diciembre de 2024
Se agregaron permisos a the section called “Política administrada de AWS: AWSServiceRoleForAmazonEKSNodegroup”.	`AWSServiceRoleForAmazonEKSNodegroup` se actualizó de modo que sea compatible con regiones de China.	22 de noviembre de 2024
Se agregaron permisos a the section called “Política administrada de AWS: AmazonEKSLocalOutpostClusterPolicy”	Se agregó un permiso `ec2:DescribeAvailabilityZones` a AWS de modo que el administrador de controladores de la nube de `AmazonEKSLocalOutpostClusterPolicy` en el plano de control del clúster pueda identificar la zona de disponibilidad en la que se encuentra cada nodo.	21 de noviembre de 2024
Se agregaron permisos a the section called “Política administrada de AWS: AWSServiceRoleForAmazonEKSNodegroup”.	Se actualizó la política `AWSServiceRoleForAmazonEKSNodegroup` para permitir `ec2:RebootInstances` para las instancias creadas por grupos de nodos administrados por Amazon EKS. Se restringieron los permisos `ec2:CreateTags` para los recursos de Amazon EC2.	20 de noviembre de 2024

Cambio	Descripción	Fecha
Se agregaron permisos a the section called “Política administrada de AWS: AmazonEKSServiceRolePolicy”.	EKS actualizó la política `AmazonEKSServiceRolePolicy` administrada por AWS. Se han agregado permisos para las políticas de acceso a EKS, la administración del equilibrador de carga y la limpieza automatizada de los recursos del clúster.	16 de noviembre de 2024
Presentó la política the section called “Política administrada de AWS: AmazonEKSComputePolicy”.	EKS actualizó la política `AmazonEKSComputePolicy` administrada por AWS. Se actualizaron los permisos de recursos para la acción `iam:AddRoleToInstanceProfile` .	7 de noviembre de 2024
Presentó la política the section called “Política administrada de AWS: AmazonEKSComputePolicy”.	AWS presentó la `AmazonEKSComputePolicy` .	1 de noviembre de 2024
Se agregaron permisos a `AmazonEKSClusterPolicy`	Se agregó el permiso `ec2:DescribeInstanceTopology` para permitir que Amazon EKS asocie información de topología al nodo como marcas.	1 de noviembre de 2024
Presentó la política the section called “Política administrada de AWS: AmazonEKSBlockStoragePolicy”.	AWS presentó la `AmazonEKSBlockStoragePolicy` .	30 de octubre de 2024

Cambio	Descripción	Fecha
Presentó la política the section called “Política administrada de AWS: AmazonEKSLoadBalancingPolicy”.	AWS presentó la `AmazonEKSLoadBalancingPolicy` .	30 de octubre de 2024
Se agregaron permisos a AmazonEKSServiceRolePolicy.	Se agregaron permisos `cloudwatch:PutMetricData` para permitir que Amazon EKS publique métricas en Amazon CloudWatch.	29 de octubre de 2024
Presentó la política the section called “Política administrada de AWS: AmazonEKSNetworkingPolicy”.	AWS presentó la `AmazonEKSNetworkingPolicy` .	28 de octubre de 2024
Permisos añadidos a las políticas `AmazonEKSServicePolicy` y `AmazonEKSServiceRolePolicy`	Se añadió `ec2:GetSecurityGroupsForVpc` y los permisos de etiquetas asociados para permitir que Amazon EKS lea la información del grupo de seguridad y actualice las etiquetas relacionadas.	10 de octubre de 2024
Introdujo AmazonEKSWorkerNodeMinimalPolicy.	AWS presentó la `AmazonEKSWorkerNodeMinimalPolicy` .	3 de octubre de 2024

Cambio	Descripción	Fecha
Se agregaron permisos a AWSServiceRoleForAmazonEKSNodegroup.	Se añadieron los permisos `autoscaling:ResumeProcesses` y `autoscaling:SuspendProcesses` para permitir que Amazon EKS suspenda y reanude el `AZRebalance` en los grupos de escalado automático administrados por Amazon EKS.	21 de agosto de 2024
Se agregaron permisos a AWSServiceRoleForAmazonEKSNodegroup.	Se agregó el permiso `ec2:DescribeCapacityReservations` para permitir que Amazon EKS describa la reserva de capacidad en la cuenta de usuario. Se agregó el permiso `autoscaling:PutScheduledUpdateGroupAction` para permitir configurar el escalado programado en los grupos de nodos `CAPACITY_BLOCK` .	27 de junio de 2024

Cambio	Descripción	Fecha
AmazonEKS_CNI_Policy: actualización de una política existente	Amazon EKS ha agregado nuevos permisos `ec2:DescribeSubnets` para que el complemento CNI de Amazon VPC para Kubernetes pueda ver la cantidad de direcciones IP libres en las subredes de Amazon VPC. La CNI de la VPC puede usar las direcciones IP libres de cada subred para seleccionar las subredes con la mayor cantidad de direcciones IP libres y utilizarlas al crear una interfaz de red elástica.	4 de marzo de 2024
AmazonEKSWorkerNodePolicy: Actualización de una política existente	Amazon EKS agregó nuevos permisos para permitir las Pod Identities de EKS. El agente de Pod Identity de Amazon EKS usa el rol de nodo.	26 de noviembre de 2023
Se presentó AmazonEFSCSIDriverPolicy.	AWS presentó la `AmazonEFSCSIDriverPolicy` .	26 de julio de 2023
Se agregaron permisos a AmazonEKSClusterPolicy.	Se agregó un permiso de `ec2:DescribeAvailabilityZones` para permitir que Amazon EKS obtenga los detalles de AZ durante la detección automática de subredes al crear equilibradores de carga.	7 de febrero de 2023

Cambio	Descripción	Fecha
Condiciones de la política actualizadas en AmazonEBSCSIdRiverPolicy.	Eliminación de condiciones de política no válidas con caracteres comodín en el campo clave de `StringLike`. También se agregó una nueva condición `ec2:ResourceTag/kubernetes.io/created-for/pvc/name: "*"` a `ec2:DeleteVolume`, que permite al controlador CSI de EBS eliminar los volúmenes creados por el complemento integrado en el árbol.	17 de noviembre de 2022
Se agregaron permisos a AmazonEKSLocalOutpostServiceRolePolicy.	Se agregó `ec2:DescribeVPCAttribute`, `ec2:GetConsoleOutput` y `ec2:DescribeSecret` para permitir una mejor validación de los requisitos previos y un control del ciclo de vida administrado. También se agregó `ec2:DescribePlacementGroups` y `"arn:aws:ec2:*:*:placement-group/*"` a `ec2:RunInstances` para admitir el control de ubicación de las instancias de Amazon EC2 del plano de control en Outposts.	24 de octubre de 2022

Cambio	Descripción	Fecha
Actualice los permisos del registro de Amazon Elastic Container Registry en AmazonEKSLocalOutpostClusterPolicy.	Se movió la acción `ecr:GetDownloadUrlForLayer` de todas las secciones de recursos a una sección específica. Se agregó el recurso `arn:aws:ecr:*:*:repository/eks/` . Se eliminó el recurso `arn:aws:ecr:` . Este recurso está cubierto por el recurso agregado `arn:aws:ecr:*:*:repository/eks/*` .	20 de octubre de 2022
Se agregaron permisos a AmazonEKSLocalOutpostClusterPolicy.	Se agregó el repositorio de Amazon Elastic Container Registry `arn:aws:ecr:*:*:repository/kubelet-config-updater` para que las instancias del plano de control del clúster puedan actualizar algunos argumentos de `kubelet`.	31 de agosto de 2022
Se presentó AmazonEKSLocalOutpostClusterPolicy.	AWS presentó la `AmazonEKSLocalOutpostClusterPolicy` .	24 de agosto de 2022
Se presentó AmazonEKSLocalOutpostServiceRolePolicy.	AWS presentó la `AmazonEKSLocalOutpostServiceRolePolicy` .	23 de agosto de 2022
Se presentó AmazonEBSCSIDriverPolicy.	AWS presentó la `AmazonEBSCSIDriverPolicy` .	4 de abril de 2022

Cambio	Descripción	Fecha
Se agregaron permisos a AmazonEKSWorkerNodePolicy.	Se agregó `ec2:DescribeInstanceTypes` para habilitar las AMI optimizadas de Amazon EKS que pueden detectar en forma automática las propiedades de nivel de instancia.	21 de marzo de 2022
Se agregaron permisos a AWSServiceRoleForAmazonEKSNodegroup.	Se agregó el permiso `autoscaling:EnableMetricsCollection` para permitir que Amazon EKS habilite la recopilación de métricas.	13 de diciembre de 2021
Se agregaron permisos a AmazonEKSClusterPolicy.	Se han agregados permisos de `ec2:DescribeAccountAttributes`, `ec2:DescribeAddresses` y `ec2:DescribeInternetGateways` a fin de permitir que Amazon EKS cree un rol vinculado a servicios para un equilibrador de carga de red.	17 de junio de 2021
Amazon EKS comenzó a realizar el seguimiento de los cambios.	Amazon EKS comenzó a realizar el seguimiento de los cambios de las políticas administradas de AWS.	17 de junio de 2021

Solución de problemas de IAM

En este tema se tratan algunos errores habituales que pueden aparecer al utilizar Amazon EKS con IAM y cómo solucionarlos.

AccessDeniedException

Si recibe una `AccessDeniedException` al llamar a una operación de API de AWS, las credenciales de la entidad principal de IAM que utiliza no tienen los permisos necesarios para hacer esa llamada.

```
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
```

En el mensaje de ejemplo anterior, el usuario no tiene permisos para llamar a la operación `DescribeCluster` de la API de Amazon EKS. Para proporcionar permisos de administrador de Amazon EKS a una entidad principal de IAM, consulte the section called “Políticas basadas en identidades”.

Para obtener información general sobre IAM, consulte Control del acceso con políticas en la Guía del usuario de IAM.

No puede ver Nodos en la pestaña Informática o cualquier cosa de la pestaña Recursos y recibe un error en la Consola de administración de AWS

Es posible que aparezca un mensaje de error en la consola que dice `Your current user or role does not have access to Kubernetes objects on this EKS cluster`. Asegúrese de que el usuario principal de IAM con el que está utilizando la Consola de administración de AWS tenga los permisos necesarios. Para obtener más información, consulte the section called “Permisos necesarios”.

El **`ConfigMap`** de aws-auth no concede acceso al clúster

El autenticador de IAM de AWS no permite una ruta de acceso en el ARN de rol utilizado en el `ConfigMap`. Por lo tanto, antes de especificar `rolearn`, elimine la ruta de acceso. Por ejemplo, cambie `arn:aws:iam::111122223333:role/team/developers/eks-admin` a `arn:aws:iam::111122223333:role/eks-admin` .

No tengo autorización para realizar la operación iam:PassRole

Si recibe un error que indica que no tiene autorización para llevar a cabo la acción `iam:PassRole`, sus políticas deben actualizarse para permitirle pasar un rol a Amazon MQ.

Algunos servicios de AWS le permiten transferir un rol existente a dicho servicio en lugar de crear un nuevo rol de servicio o uno vinculado al servicio. Para ello, debe tener permisos para transferir el rol al servicio.

En el siguiente ejemplo, el error se produce cuando un usuario de IAM denominado `marymajor` intenta utilizar la consola para realizar una acción en Amazon EKS. Sin embargo, la acción requiere que el servicio cuente con permisos que otorguen un rol de servicio. Mary no tiene permisos para transferir el rol al servicio.

```
User: {arn-aws}iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

En este caso, las políticas de Mary se deben actualizar para permitirle realizar la acción `iam:PassRole`.

Si necesita ayuda, póngase en contacto con su gestionador de AWS. El gestionador es la persona que le proporcionó las credenciales de inicio de sesión.

Quiero permitir que personas ajenas a mi cuenta de AWS accedan a mis recursos de Amazon EKS.

Puede crear un rol que los usuarios de otras cuentas o las personas externas a la organización puedan utilizar para acceder a sus recursos. Puede especificar una persona de confianza para que asuma el rol. En el caso de los servicios que admitan las políticas basadas en recursos o las listas de control de acceso (ACL), puede utilizar dichas políticas para conceder a las personas acceso a sus recursos.

Para obtener más información, consulte lo siguiente:

- Para saber si Amazon EKS admite estas características, consulte the section called “Amazon EKS e IAM”.
- Para obtener información acerca de cómo proporcionar acceso a los recursos de las cuentas de AWS de su propiedad, consulte Proporcionar acceso a un usuario de IAM a otra cuenta de AWS de la que es propietario en la Guía del usuario de IAM.
- Para obtener información acerca de cómo proporcionar acceso a los recursos a cuentas de AWS de terceros, consulte Proporcionar acceso a cuentas de AWS que son propiedad de terceros en la Guía del usuario de IAM.

- Para obtener información sobre cómo proporcionar acceso mediante una federación de identidades, consulta Proporcionar acceso a usuarios autenticados externamente (identidad federada) en la Guía del usuario de IAM.
- Para conocer sobre la diferencia entre las políticas basadas en roles y en recursos para el acceso entre cuentas, consulte Acceso a recursos entre cuentas en IAM en la Guía del usuario de IAM.

Los contenedores de pods muestran el siguiente error: **`An error occurred (SignatureDoesNotMatch) when calling the GetCallerIdentity operation: Credential should be scoped to a valid region`**

Los contenedores reciben este error si la aplicación lleva a cabo solicitudes explícitamente al punto de conexión de AWS STS global (`https://sts.amazonaws`) y su cuenta de servicio de Kubernetes está configurada para utilizar un punto de conexión regional. Puede resolver el problema con una de las siguientes opciones:

- Actualice el código de la aplicación para eliminar las llamadas explícitas al punto de conexión global de AWS STS.
- Actualice el código de la aplicación para realizar llamadas explícitas a puntos de conexión regionales, como `https://sts.us-west-2.amazonaws.com` . Su aplicación debe tener redundancia integrada para seleccionar una región de AWS diferente en caso de producirse un error del servicio en la región de AWS. Para obtener más información, consulte Administración de AWS STS en una región de AWS en la guía del usuario de IAM.
- Configure sus cuentas de servicio para utilizar el punto de conexión global. Los clústeres utilizan el punto de conexión regional de forma predeterminada. Para obtener más información, consulte the section called “Puntos de conexión de STS”.

Rol de IAM del clúster de Amazon EKS

Se requiere un rol de IAM de clúster de Amazon EKS para cada clúster. Los clústeres de Kubernetes administrados por Amazon EKS utilizan este rol para administrar los nodos, y el proveedor de nube heredado lo usa para crear equilibradores de carga con Elastic Load Balancing para los servicios.

Para poder crear clústeres de Amazon EKS, debe crear un rol de IAM con una de las siguientes políticas de IAM:

- AmazonEKSClusterPolicy

- Una política de IAM personalizada. Los permisos mínimos que aparecen a continuación permiten que el clúster de Kubernetes administre los nodos, pero no que el proveedor de nube heredado cree equilibradores de carga con Elastic Load Balancing. La política de IAM personalizada debe disponer al menos de los siguientes permisos:

```
{
  "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**
>
> Antes del 3 de octubre de 2023, se requería AmazonEKSClusterPolicy en el rol de IAM de cada clúster.

Antes del 16 de abril de 2020, se requería AmazonEKSServicePolicy y AmazonEKSClusterPolicy y el nombre sugerido era `eksServiceRole`. Con el rol vinculado a servicios de `AWSServiceRoleForAmazonEKS`, la política AmazonEKSServicePolicy ya no es necesaria para clústeres creados a partir del 16 de abril de 2020.

Comprobar si existe un rol de clúster existente

Puede utilizar el siguiente procedimiento para verificar y conocer si la cuenta ya dispone del rol de clúster de Amazon EKS.

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación izquierdo, elija Roles.
3. En la lista de roles, busque `eksClusterRole`. Si no existe un rol que incluya `eksClusterRole`, consulte the section called “Crear el rol de clúster de Amazon EKS” para crearlo. Si existe un rol que incluye `eksClusterRole`, seleccione el rol para ver las políticas asociadas.
4. Elija Permissions.
5. Asegúrese de que la política administrada AmazonEKSClusterPolicyse ha asociado al rol. Si la política se ha adjuntado, entonces el rol de clúster de Amazon EKS se ha configurado correctamente.
6. Elija Trust relationships (Relaciones de confianza) y, a continuación, Edit trust policy (Editar política de confianza).
7. Verifique que la relación de confianza contiene la siguiente política. Si la relación de confianza coincide con la política a continuación, seleccione Cancelar. Si la relación de confianza no coincide, copie la política en la ventana Editar política de confianza y elija Actualizar política.

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

```
}
```

Crear el rol de clúster de Amazon EKS

Para crear el rol de clúster, puede utilizar la Consola de administración de AWS o la AWS CLI.

Consola de administración de AWS

a. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
b. Elija Roles y, a continuación, Create role (Crear rol).
c. En Tipo de entidad de confianza, seleccione Servicio de AWS.
d. En la lista desplegable Casos de uso para otros servicios de AWS, elija EKS.
e. Elija EKS: clúster para su caso de uso y, luego, elija Siguiente.
f. En la pestaña Agregar permisos, seleccione Siguiente.
g. En Nombre del rol, ingrese un nombre único para su rol, por ejemplo, `eksClusterRole`.
h. En Description (Descripción), ingrese texto descriptivo, como `Amazon EKS - Cluster role`.
i. Seleccione Crear rol.

AWS CLI

a. Copie el siguiente contenido en un archivo denominado *`cluster-trust-policy.json`*.

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

b. Creación del rol. Puede reemplazar *`eksClusterRole`* con el nombre que usted elija.

```
aws iam create-role \
  --role-name eksClusterRole \
```

```
  --assume-role-policy-document file://"cluster-trust-policy.json"
```

c. Adjunte la política de IAM necesaria al rol de IAM

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

Rol de IAM de nodo de Amazon EKS

El daemon de `kubelet` del nodo de Amazon EKS realiza llamadas a las API de AWS en su nombre. Los nodos reciben permisos de dichas llamadas de API a través de políticas asociadas y de un perfil de instancias de IAM. Antes de poder lanzar nodos y registrarlos en un clúster, debe crear un rol de IAM para dichos nodos, para utilizarlo cuando se lancen. Este requisito se aplica a nodos lanzados con la AMI optimizada para Amazon EKS proporcionada por Amazon o con cualquier otra AMI de nodo que pretenda utilizar. Además, este requisito se aplica tanto a los grupos de nodos administrados como a los administrados por el usuario.

> **Note**
>
> No se puede utilizar el mismo rol que se utiliza para crear clústeres.

Antes de crear un nodo, debe crear un rol de IAM con los siguientes permisos:

- Permisos para que el `kubelet` pueda describir los recursos de Amazon EC2 en la VPC, tal y como los proporciona la política AmazonEKSWorkerNodePolicy. Esta política también proporciona los permisos para el agente de Pod Identity de Amazon EKS.
- Permisos para que el `kubelet` pueda utilizar imágenes de contenedores de Amazon Elastic Container Registry (Amazon ECR), según lo dispuesto en la política AmazonEC2ContainerRegistryPullOnly. Los permisos para utilizar imágenes de contenedor de Amazon Elastic Container Registry (Amazon ECR) son necesarios porque los complementos integrados para redes ejecutan pods que utilizan imágenes de contenedor de Amazon ECR.
- (Opcional) Permisos para que el agente de Pod Identity de Amazon EKS utilice la acción `eks-auth:AssumeRoleForPodIdentity` para recuperar las credenciales de los pods. Si no usa AmazonEKSWorkerNodePolicy, debe proporcionar este permiso además de los permisos de EC2 para utilizar Pod Identity de EKS.

- (Opcional) Si no utiliza Pod Identity de EKS e IRSA para otorgar permisos a los pods de CNI de VPC, debe proporcionar permisos para la CNI de la VPC en el rol de instancia. Puede utilizar la política administrada `AmazonEKS_CNI_Policy` (si creó el clúster con la familia `IPv4`) o una política IPv6 que usted cree (si creó el clúster con la familia `IPv6`). Sin embargo, en lugar de adjuntar la política a este rol, le recomendamos adjuntar la política a un rol independiente utilizado específicamente para el complemento Amazon VPC CNI. Para obtener más información acerca de cómo crear un rol independiente para el complemento Amazon VPC CNI, consulte the section called “Configuración para IRSA”.

> **Note**
>
> Los grupos de nodos de Amazon EC2 deben tener un rol de IAM diferente al perfil de Fargate. Para obtener más información, consulte the section called “Rol de IAM de ejecución de pods”.

Verificar un rol de nodo existente

Puede utilizar el siguiente procedimiento para verificar y conocer si la cuenta ya dispone del rol de nodo de Amazon EKS.

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación izquierdo, elija Roles.
3. En la lista de roles, busque `eksNodeRole`, `AmazonEKSNodeRole` o `NodeInstanceRole`. Si no existe un rol con alguno de esos nombres, consulte the section called “Crear el rol de IAM de nodo de Amazon EKS” para crear el rol. Si existe un rol que contiene `eksNodeRole`, `AmazonEKSNodeRole` o `NodeInstanceRole`, seleccione el rol para ver las políticas adjuntas.
4. Elija Permisos.
5. Asegúrese de que las políticas administradas AmazonEKSWorkerNodePolicy y AmazonEC2ContainerRegistryPullOnly estén asociadas al rol, o que haya una política personalizada asociada con los permisos mínimos.

> **Note**
>
> Si la política AmazonEKS_CNI_Policy se encuentra adjunta al rol, se recomienda eliminarla y adjuntarla a un rol de IAM asignado a la cuenta de servicio de Kubernetes

de `aws-node` en su lugar. Para obtener más información, consulte the section called "Configuración para IRSA".

6. Elija Relaciones de confianza y, a continuación, Editar política de confianza.
7. Verifique que la relación de confianza contiene la siguiente política. Si la relación de confianza coincide con la política a continuación, seleccione Cancelar. Si la relación de confianza no coincide, copie la política en la ventana Editar política de confianza y elija Actualizar política.

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

Crear el rol de IAM de nodo de Amazon EKS

Puede crear el rol de IAM del nodo con la Consola de administración de AWS o la CLI de AWS.

Consola de administración de AWS

a. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
b. En el panel de navegación izquierdo, elija Roles.
c. En la página Roles, elija Crear rol.
d. En la página Seleccionar entidad de confianza, haga lo siguiente:
    i. En la sección Tipo de entidad de confianza, elija Servicio de AWS.
    ii. En Caso de uso, elija EC2.
    iii. Elija Siguiente.

e. En la página Agregar permisos, asocie una política personalizada o haga lo siguiente:

i. En el cuadro Filtrar políticas, escriba `AmazonEKSWorkerNodePolicy`.

ii. A continuación, marque la casilla situada a la izquierda de AmazonEKSWorkerNodePolicy en los resultados de la búsqueda.

iii. Elija Borrar filtros.

iv. En el cuadro Filtrar políticas, escriba `AmazonEC2ContainerRegistryPullOnly`.

v. Marque la casilla situada a la izquierda de AmazonEC2ContainerRegistryPullOnly en los resultados de la búsqueda.

La política administrada AmazonEKS_CNI_Policy o una política de IPv6 que cree también se tiene que adjuntar a este rol o a un rol diferente asignado a la cuenta de servicio de Kubernetes de `aws-node`. Se recomienda asignar la política al rol asociado a la cuenta de servicio de Kubernetes en lugar de asignarla a este rol. Para obtener más información, consulte the section called “Configuración para IRSA”.

vi. Elija Siguiente.

f. En la página Nombrar, revisar y crear, haga lo siguiente:

i. En Nombre del rol, ingrese un nombre único para su rol, por ejemplo, `AmazonEKSNodeRole`.

ii. En Descripción, sustituya el texto actual por un texto descriptivo, como `Amazon EKS - Node role`.

iii. En Agregar etiquetas (Opcional), de manera opcional, agregue metadatos al rol asociando etiquetas como pares de clave-valor. Para obtener más información sobre el uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM en la Guía de usuario de IAM.

iv. Seleccione Crear rol.

AWS CLI

a. Ejecute el siguiente comando para crear un archivo `node-role-trust-relationship.json`.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRole"
```

```
            ],
            "Principal": {
                "Service": [
                    "ec2.amazonaws.com"
                ]
            }
        }
    ]
}
```

b. Creación del rol de IAM.

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

c. Adjunte al rol de IAM las dos políticas administradas por IAM necesarias.

```
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. Adjunte una de las siguientes políticas de IAM al rol de IAM en función de la familia de IP con la que haya creado el clúster. La política debe adjuntarse a este rol o a un rol asociado a la cuenta de servicio de `aws-node` de Kubernetes que se utiliza para el complemento CNI de Amazon VPC. Se recomienda asignar la política al rol asociado a la cuenta de servicio de Kubernetes. Para asignar la política al rol asociado a la cuenta de servicio de Kubernetes, consulte the section called “Configuración para IRSA”.

- IPv4

  ```
  aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
    --role-name AmazonEKSNodeRole
  ```

- IPv6

  A. Copie el siguiente texto y guárdelo en un archivo llamado `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. Creación de la política de IAM.

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-
document file://vpc-cni-ipv6-policy.json
```

C. Adjunte la política de IAM al rol de IAM. Reemplace *111122223333* por el ID de su cuenta.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy \
  --role-name AmazonEKSNodeRole
```

Rol de IAM del clúster del modo automático de Amazon EKS

Se requiere un rol de IAM de clúster de Amazon EKS para cada clúster. Los clústeres de Kubernetes administrados por Amazon EKS utilizan este rol para automatizar las tareas rutinarias de almacenamiento, redes y escalado automático de computación.

Antes de crear clústeres de Amazon EKS, debe crear un rol de IAM con las políticas necesarias para el modo automático de EKS. Puede asociar las políticas administradas de AWS IAM sugeridas o crear políticas personalizadas con permisos equivalentes.

- AmazonEKSComputePolicy
- AmazonEKSBlockStoragePolicy
- AmazonEKSLoadBalancingPolicy
- AmazonEKSNetworkingPolicy
- AmazonEKSClusterPolicy

Comprobar si existe un rol de clúster existente

Puede utilizar el siguiente procedimiento para verificar y conocer si la cuenta ya dispone del rol de clúster de Amazon EKS.

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación izquierdo, elija Roles.
3. En la lista de roles, busque `AmazonEKSAutoClusterRole`. Si no existe un rol que incluya `AmazonEKSAutoClusterRole`, consulte las instrucciones de la siguiente sección para crear el rol. Si existe un rol que incluye `AmazonEKSAutoClusterRole`, seleccione el rol para ver las políticas asociadas.
4. Elija Permissions.
5. Asegúrese de que la política administrada AmazonEKSClusterPolicyse ha asociado al rol. Si la política se ha adjuntado, entonces el rol de clúster de Amazon EKS se ha configurado correctamente.
6. Elija Trust relationships (Relaciones de confianza) y, a continuación, Edit trust policy (Editar política de confianza).
7. Verifique que la relación de confianza contiene la siguiente política. Si la relación de confianza coincide con la política a continuación, seleccione Cancelar. Si la relación de confianza no coincide, copie la política en la ventana Editar política de confianza y elija Actualizar política.

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

> **Note**
>
> AWS no requiere el nombre `AmazonEKSAutoClusterRole` de este rol.

Crear el rol de clúster de Amazon EKS

Para crear el rol de clúster, puede utilizar la Consola de administración de AWS o la AWS CLI.

Consola de administración de AWS

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Elija Roles y, a continuación, Create role (Crear rol).
3. En Tipo de entidad de confianza, seleccione Servicio de AWS.
4. En la lista desplegable Casos de uso para otros servicios de AWS, elija EKS.
5. Elija EKS: clúster para su caso de uso y, luego, elija Siguiente.
6. En la pestaña Agregar permisos, seleccione las políticas y, a continuación, elija Siguiente.
   - AmazonEKSComputePolicy
   - AmazonEKSBlockStoragePolicy
   - AmazonEKSLoadBalancingPolicy
   - AmazonEKSNetworkingPolicy

- AmazonEKSClusterPolicy

7. En Nombre del rol, ingrese un nombre único para su rol, por ejemplo, `AmazonEKSAutoClusterRole`.
8. En Description (Descripción), ingrese texto descriptivo, como `Amazon EKS - Cluster role`.
9. Seleccione Crear rol.

AWS CLI

1. Copie el siguiente contenido en un archivo denominado *`cluster-trust-policy.json`*.

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

2. Creación del rol. Puede reemplazar *`AmazonEKSAutoClusterRole`* por el nombre que elija.

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

3. Asocie las políticas de IAM necesarias al rol:

AmazonEKSClusterPolicy

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

AmazonEKSComputePolicy:

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

AmazonEKSBlockStoragePolicy:

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

AmazonEKSLoadBalancingPolicy:

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

AmazonEKSNetworkingPolicy:

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

Rol de IAM de nodo del modo automático de Amazon EKS

> **Note**
>
> No se puede utilizar el mismo rol que se utiliza para crear clústeres.

Antes de crear nodos, debe crear un rol de IAM con las siguientes políticas, o permisos equivalentes:

- AmazonEKSWorkerNodeMinimalPolicy
- AmazonEC2ContainerRegistryPullOnly

Verificar un rol de nodo existente

Puede utilizar el siguiente procedimiento para verificar y conocer si la cuenta ya dispone del rol de nodo de Amazon EKS.

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación izquierdo, elija Roles.
3. En la lista de roles, busque `AmazonEKSAutoNodeRole`. Si no existe un rol con uno de esos nombres, consulte las instrucciones que figuran en la siguiente sección para crear el rol. Si existe un rol que contiene `AmazonEKSAutoNodeRole`, seleccione el rol para ver las políticas asociadas.
4. Elija Permisos.
5. Asegúrese de que se asocian las políticas requeridas mencionadas anteriormente, o políticas personalizadas equivalentes.
6. Elija Relaciones de confianza y, a continuación, Editar política de confianza.
7. Verifique que la relación de confianza contiene la siguiente política. Si la relación de confianza coincide con la política a continuación, seleccione Cancelar. Si la relación de confianza no coincide, copie la política en la ventana Editar política de confianza y elija Actualizar política.

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

Crear el rol de IAM de nodo de Amazon EKS

Puede crear el rol de IAM del nodo con la Consola de administración de AWS o la CLI de AWS.

Consola de administración de AWS

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.

2. En el panel de navegación izquierdo, elija Roles.
3. En la página Roles, elija Crear rol.
4. En la página Seleccionar entidad de confianza, haga lo siguiente:
   a. En la sección Tipo de entidad de confianza, elija Servicio de AWS.
   b. En Caso de uso, elija EC2.
   c. Elija Siguiente.
5. En la página Agregar permisos, elija las siguientes políticas:
   - AmazonEKSWorkerNodeMinimalPolicy
   - AmazonEC2ContainerRegistryPullOnly
6. En la página Nombrar, revisar y crear, haga lo siguiente:
   a. En Nombre del rol, ingrese un nombre único para su rol, por ejemplo, `AmazonEKSAutoNodeRole`.
   b. En Descripción, sustituya el texto actual por un texto descriptivo, como `Amazon EKS - Node role`.
   c. En Agregar etiquetas (Opcional), de manera opcional, agregue metadatos al rol asociando etiquetas como pares de clave-valor. Para obtener más información sobre el uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM en la Guía de usuario de IAM.
   d. Seleccione Crear rol.

AWS CLI

Creación del rol de IAM de nodo

Utilice el archivo node-trust-policy.json del paso anterior para definir qué entidades pueden asumir el rol. Ejecute el siguiente comando para crear el rol de IAM del nodo:

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

Apunte el ARN del rol

Después de crear el rol, recupere y guarde el ARN del rol de IAM del nodo. Necesitará este ARN en los pasos siguientes. Utilice el siguiente comando para obtener el ARN:

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

Asociar las políticas necesarias

Asocie las siguientes políticas administradas de AWS al rol de IAM del nodo para proporcionar los permisos necesarios:

Para asociar AmazonEKSWorkerNodeMinimalPolicy:

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

Para asociar AmazonEC2ContainerRegistryPullOnly:

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

Rol de IAM de capacidad de Amazon EKS

Las capacidades de EKS necesitan que se configure un rol de IAM de capacidad (o rol de capacidad). Las capacidades utilizan este rol para llevar a cabo acciones en los servicios de AWS y acceder a los recursos de Kubernetes del clúster mediante entradas de acceso que se crean automáticamente.

Antes de poder especificar un rol de capacidad durante la creación de la capacidad, debe crear el rol de IAM con la política de confianza y los permisos adecuados para el tipo de capacidad. Una vez creado este rol de IAM, se puede reutilizar para cualquier cantidad de recursos de capacidad.

Requisitos del rol de capacidad

El rol de capacidad debe cumplir los siguientes requisitos:

- El rol debe estar en la misma cuenta de AWS que el clúster y el recurso de capacidad.
- El rol debe tener una política de confianza que permita al servicio de las capacidades de EKS asumir el rol.
- El rol debe tener los permisos adecuados para el tipo de capacidad y los requisitos del caso de uso (consulte the section called “Permisos por tipo de capacidad”).

Política de confianza para roles de capacidad

Todos los roles de capacidad deben incluir la siguiente política de confianza:

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

Esta política de confianza permite a EKS hacer lo siguiente:

- Asumir el rol para llevar a cabo operaciones de API de AWS
- Etiquetar sesiones con fines de auditoría y seguimiento

Permisos por tipo de capacidad

Los permisos de IAM necesarios dependen de la capacidad que utilice y del modelo de implementación.

> **Note**
>
> Para las implementaciones de producción que utilizan selectores de roles de IAM con ACK, o cuando se utilizan kro o Argo CD sin integración de servicios de AWS, es posible que el rol de capacidad no necesite ningún permiso de IAM más allá de la política de confianza.

kro (Kube Resource Orchestrator)

No se necesita ningún permiso de IAM. Puede crear un rol de capacidad sin políticas adjuntas. kro solo necesita los permisos de RBAC de Kubernetes para crear y administrar los recursos de Kubernetes.

AWS Controladores para Kubernetes de (ACK)

ACK admite dos modelos de permisos:

- Configuración sencilla (desarrollo y pruebas): agregue los permisos de los servicios de AWS directamente al rol de capacidad. Funciona bien para comenzar, para implementaciones de una sola cuenta o cuando todos los usuarios necesitan los mismos permisos.
- Práctica recomendada de producción: utilice los selectores de roles de IAM para implementar el acceso de privilegio mínimo. Con este enfoque, el rol de capacidad solo necesita el permiso `sts:AssumeRole` para asumir roles específicos del servicio. No agregará ningún permiso de servicio de AWS (como S3 o RDS) al propio rol de capacidad, ya que esos permisos se conceden a los roles de IAM individuales que se asignan a espacios de nombres específicos.

  Los selectores de roles de IAM permiten lo siguiente:

  - Aislamiento de permisos del espacio de nombres
  - Administración de recursos entre cuentas
  - Roles de IAM específicos de equipos
  - Modelo de seguridad de privilegio mínimo

  Ejemplo de política de rol de capacidad para el enfoque del selector de roles de IAM:

```
{
  "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"
      ]
    }
  ]
}
```

Para obtener información detallada sobre la configuración de los permisos de ACK, lo que incluye los selectores de roles de IAM, consulte the section called “Configuración de permisos”.

Argo CD

No se necesita ningún permiso de IAM de forma predeterminada. Es posible que se necesiten permisos opcionales para:

- AWS Secrets Manager: si utiliza Secrets Manager para almacenar las credenciales del repositorio de Git
- AWS CodeConnections: si utiliza CodeConnections para la autenticación del repositorio de Git

Ejemplo de política de Secrets Manager y 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/*"
    }
  ]
}
```

Para obtener información detallada sobre los requisitos de permisos de Argo CD, consulte the section called “Consideraciones sobre Argo CD”.

Comprobación de un rol de capacidad existente

Puede utilizar el procedimiento siguiente para comprobar si la cuenta ya dispone del rol de IAM de capacidad adecuado para su caso de uso.

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación izquierdo, elija Roles.
3. Busque el nombre del rol de capacidad en la lista de roles (por ejemplo, `ACKCapabilityRole` o `ArgoCDCapabilityRole`).
4. Si existe un rol, selecciónelo para ver las políticas adjuntas y la relación de confianza.
5. Elija Relaciones de confianza y, a continuación, Editar política de confianza.
6. Compruebe que la relación de confianza coincida con la política de confianza de capacidad. Si no coincide, actualice la política de confianza.
7. Elija Permisos y compruebe que el rol tenga los permisos adecuados para su tipo de capacidad y caso de uso.

Creación de un rol de IAM de capacidad

Puede utilizar la Consola de administración de AWS o la AWS CLI para crear un rol de capacidad.

Consola de administración de AWS

a. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
b. Elija Roles y, a continuación, Create role (Crear rol).
c. En Tipo de entidad de confianza, seleccione Política de confianza personalizada.
d. Copie y pegue la política de confianza de capacidad en el editor de políticas de confianza.
e. Elija Siguiente.
f. En la pestaña Agregar permisos, seleccione o cree las políticas adecuadas para su tipo de capacidad (consulte the section called “Permisos por tipo de capacidad”). En el caso de kro, también puede omitir este paso.
g. Elija Siguiente.
h. En Nombre del rol, ingrese un nombre único para su rol (por ejemplo, `ACKCapabilityRole`, `ArgoCDCapabilityRole` o `kroCapabilityRole`).
i. En Description (Descripción), ingrese texto descriptivo, como `Amazon EKS - ACK capability role`.

j. Seleccione Crear rol.

AWS CLI

a. Copie la política de confianza de capacidad en un archivo denominado `capability-trust-policy.json`.

b. Creación del rol. Sustituya `ACKCapabilityRole` por el nombre del rol que desee.

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

c. Asocie las políticas de IAM necesarias al rol. En el caso de ACK, adjunte políticas para los servicios de AWS que desee administrar. En el caso de Argo CD, adjunte políticas para Secrets Manager o CodeConnections si es necesario. En el caso de kro, también puede omitir este paso.

Ejemplo de ACK con permisos de S3:

```
aws iam put-role-policy \
  --role-name ACKCapabilityRole \
  --policy-name S3Management \
  --policy-document file://s3-policy.json
```

Solución de problemas de roles de capacidad

Se produce un error en la creación de la capacidad porque el rol de IAM no es válido

Verifique lo siguiente:

- El rol existe en la misma cuenta que el clúster.
- La política de confianza coincide con la política de confianza de capacidad.
- Tiene el permiso `iam:PassRole` para el rol.

La capacidad muestra errores de permisos

Verifique lo siguiente:

- El rol tiene los permisos de IAM necesarios para el tipo de capacidad.
- La entrada de acceso existe en el clúster para el rol.
- Si es necesario, se deben configurar permisos de Kubernetes adicionales (consulte the section called “Permisos de Kubernetes adicionales”).

Los recursos de ACK fallan con errores de permiso denegado

Verifique lo siguiente:

- El rol tiene los permisos de IAM necesarios para su caso de uso.
- En el caso de los controladores de ACK que hacen referencia a secretos, asegúrese de haber asociado la política de entrada de acceso `AmazonEKSSecretReaderPolicy` definida a los espacios de nombres adecuados.

Para obtener más orientación sobre la solución de problemas, consulte the section called “Consideraciones para las capacidades de EKS”.

Supervisión del rendimiento de un clúster y visualización de registros

Puede observar sus datos en Amazon EKS utilizando muchas herramientas de monitoreo o registro disponibles. Los datos de registro de Amazon EKS se pueden transmitir a los servicios de AWS o herramientas de socios para análisis de datos. Hay muchos servicios disponibles en el Consola de administración de AWS que proporcionan datos para solucionar problemas de Amazon EKS. También puede utilizar una solución AWS de código abierto compatible para monitorear la infraestructura de Amazon EKS.

Después de seleccionar Clústeres en el panel de navegación izquierdo de la Consola de Amazon EKS, podrá ver el estado y los detalles del clúster al elegir el nombre del clúster y seleccionar la pestaña Observabilidad. Para ver detalles acerca de los recursos de Kubernetes existentes implementados en el clúster, consulte the section called “Acceso a recursos del clúster”.

El monitoreo es una parte importante a la hora de mantener la fiabilidad, la disponibilidad y el rendimiento de Amazon EKS y de las soluciones de AWS. Le recomendamos que recopile datos de monitorización de todas las partes de su solución de AWS. De esta forma, puede depurar con mayor facilidad un error que se produce en distintas partes del error que se produce en distintos puntos. Antes de comenzar a supervisar Amazon EKS, asegúrese de que su plan de monitorización responde a las siguientes preguntas.

- ¿Cuáles son los objetivos? ¿Necesita notificaciones en tiempo real si los clústeres escalan drásticamente?
- ¿Qué recursos hay que observar?
- ¿Con qué frecuencia necesita observar estos recursos? ¿Quiere su empresa responder rápidamente a los riesgos?
- ¿Qué herramientas pretende utilizar? Si ya ejecuta AWS Fargate como parte de su lanzamiento, puede utilizar el enrutador de registros.
- ¿A quién tiene la intención de asignar las tareas de supervisión?
- ¿A quién quiere que se envíen notificaciones cuando algo sale mal?

Supervisión y registro en Amazon EKS

Amazon EKS proporciona herramientas integradas para la supervisión y el registro. Para las versiones compatibles, el panel de observabilidad ofrece visibilidad del rendimiento del clúster. Ayuda a detectar, solucionar y remediar rápidamente los problemas. Además de las características de supervisión, incluye listas basadas en los registros de auditoría del plano de control. El plano de control de Kubernetes expone una serie de métricas que también se pueden obtener fuera de la consola.

El registro del plano de control registra todas las llamadas de API a los clústeres, la información de auditoría que captura qué usuarios realizaron qué acciones en los clústeres, así como la información basada en roles. Para obtener más información, consulte Registro y monitorización en Amazon EKS en la Guía prescriptiva de AWS.

El registro del plano de control de Amazon EKS proporciona registros de auditoría y diagnóstico directamente desde el plano de control de Amazon EKS a CloudWatch Logs en su cuenta. Estos registros hacen que le resulte más fácil asegura y ejecutar los clústeres. Puede seleccionar los tipos de registro exactos que necesita. Los registros se envían como secuencias de registro a un grupo para cada clúster de Amazon EKS en CloudWatch. Para obtener más información, consulte the section called “Registros del plano de control”.

> **Note**
>
> Al verificar los registros de autenticador de Amazon EKS en Amazon CloudWatch, se muestran las entradas que contienen texto similar al siguiente texto de ejemplo.
>
> ```
> level=info msg="mapping IAM role" groups="[]"
>  role="arn:aws:iam::111122223333:role/XXXXXXXXXXXXXXXXXX-NodeManagerRole-
> XXXXXXXX" username="eks:node-manager"
> ```
>
> Se esperan entradas que contengan este texto. El `username` es una función del servicio interna de Amazon EKS que realiza operaciones específicas para grupos de nodos administrados y Fargate.
> Para un registro de bajo nivel y personalizable, Registros de Kubernetes está disponible.

Amazon EKS se integra con AWS CloudTrail, un servicio que proporciona un registro de las acciones realizadas por un usuario, un rol o un servicio de AWS en Amazon EKS. CloudTrail captura todas las

llamadas a la API para Amazon EKS como eventos. Las llamadas capturadas incluyen las llamadas desde la consola de Amazon EKS y las llamadas desde el código a las operaciones de la API de Amazon EKS. Para obtener más información, consulte the section called “ AWS CloudTrail”.

El servidor de la API de Kubernetes expone una serie de métricas que son útiles a efectos del monitoreo y el análisis. Para obtener más información, consulte the section called “Métricas de Prometheus”.

Para configurar Fluent Bit de los Registros de Amazon CloudWatch personalizados, consulte Configuración de Fluent Bit en la Guía del usuario de Amazon CloudWatch.

Herramientas de supervisión y registro en Amazon EKS

Amazon Web Services proporciona varias herramientas que puede utilizar para monitorear Amazon EKS. Puede configurar algunas herramientas para configurar la monitorización automática, pero algunas requieren llamadas manuales. Le recomendamos que automatice las tareas de monitorización de la misma manera que lo permitan su entorno y el conjunto de herramientas existente.

En la siguiente tabla se describen varias opciones de herramientas de supervisión.

Áreas	Herramienta	Descripción	Configuración
Plano de control	Panel de observabilidad	Para las versiones compatibles, el panel de observabilidad ofrece visibilidad del rendimiento del clúster. Ayuda a detectar, solucionar y remediar rápidamente los problemas.	Procedimiento de configuración
Aplicaciones y plano de control	Prometheus	Prometheus se puede usar para supervisar las métricas y las alertas de las	Procedimiento de configuración

Áreas	Herramienta	Descripción	Configuración
		aplicaciones y el plano de control.	
Aplicaciones	Información de contenedores de CloudWatch	CloudWatch Container Insights recopila, agrega y resume métricas y registros de las aplicaciones y microservicios en contenedores.	Procedimiento de configuración
Aplicaciones	AWS Distro for OpenTelemetry (ADOT)	ADOT puede recopilar y enviar métricas correlacionadas, datos de seguimien to y metadatos a servicios de supervisi ón de AWS o socios. Se puede configurar a través de Información de contenedores de CloudWatch.	Procedimiento de configuración
Aplicaciones	Amazon DevOps Guru	Amazon DevOps Guru detecta la disponibilidad y el rendimiento operacion al a nivel de nodo.	Procedimiento de configuración

Áreas	Herramienta	Descripción	Configuración
Aplicaciones	AWS X-Ray	AWS X-Ray recibe datos de seguimiento sobre la aplicación. Estos datos de seguimiento incluyen solicitudes entrantes y salientes y metadatos sobre las solicitudes. Para Amazon EKS, la implementación requiere el complemento OpenTelemetry.	Procedimiento de configuración
Aplicaciones	Amazon CloudWatch	CloudWatch proporciona algunas métricas básicas de Amazon EKS de forma gratuita en las versiones compatibles. Puede ampliar esta funcionalidad con el operador de observabilidad de CloudWatch para gestionar la recopilación de métricas, registros y datos de seguimiento.	Procedimiento de configuración

En la siguiente tabla se describen varias opciones de herramientas de registro.

Áreas	Herramienta	Descripción	Configuración
Plano de control	Panel de observabilidad	En las versiones compatibles, el panel de observabilidad muestra listas basadas en los registros de auditoría del plano de control. También incluye enlaces a los registros del plano de control en Amazon CloudWatch.	Procedimiento de configuración
Aplicaciones	Información de contenedores de Amazon CloudWatch	Información de contenedores de Amazon CloudWatch recopila, agrega y resume métricas y registros de las aplicaciones y microservicios en contenedores.	Procedimiento de configuración
Plano de control	Registros de Amazon CloudWatch	Puede enviar registros de auditoría y diagnóstico directamente desde el plano de control de Amazon EKS a los Registros de CloudWatch en la cuenta.	Procedimiento de configuración

Áreas	Herramienta	Descripción	Configuración
Plano de control	AWS CloudTrail	Registra las llamadas a la API de un usuario, rol o servicio.	Procedimiento de configuración
Múltiples áreas para instancias de AWS Fargate	AWS Enrutador de registro de Fargate	Para las instancias de AWS Fargate, el enrutador de registros transmite los registros a los servicios de AWS o herramien tas asociadas. Utiliza AWS para Fluent Bit. Los registros se pueden transmitir a otros servicios de AWS o herramientas de socios.	Procedimiento de configuración

Supervisión del clúster con el panel de observabilidad

La consola de Amazon EKS incluye un panel de observabilidad que ofrece visibilidad del rendimiento del clúster. La información le proporciona ayuda para detectar, solucionar y remediar rápidamente los problemas. Para abrir la sección correspondiente del panel de observabilidad, puede elegir un elemento en el resumen de estado y rendimiento. Este resumen se encuentra en varios lugares, entre ellos la pestaña Observabilidad.

El panel de observabilidad se divide en varias pestañas.

Resumen

El resumen de estado y rendimiento indica la cantidad de elementos en varias categorías. Cada número funciona como un hipervínculo a una ubicación en el panel de observabilidad con una lista para esa categoría.

Estado del clúster

El estado del clúster proporciona notificaciones importantes que conviene conocer, algunas de las cuales pueden requerir que actúe cuanto antes. En esta lista se pueden ver las descripciones y los recursos afectados. El estado del clúster incluye dos tablas: Problemas de estado e Información de configuración. Para actualizar el estado de Problemas de estado, seleccione el botón Actualizar (↻). La información de configuración se actualiza automáticamente una vez cada 24 horas y no se puede actualizar manualmente.

Para obtener más información sobre Problemas de estado, consulte the section called “Preguntas frecuentes sobre el estado de los clústeres y los códigos de error con rutas de resolución”.
Para obtener más información sobre Información de contenedores, consulte the section called “Información sobre clústeres”.

Supervisión del plano de control

La pestaña Supervisión del plano de control se divide en tres secciones, cada una de las cuales ayuda a supervisar y solucionar los problemas del plano de control del clúster.

Métricas

En el caso de los clústeres de Kubernetes de la versión `1.28` y posteriores, la sección Métricas muestra gráficos de varias métricas recopiladas para varios componentes del plano de control.

Para establecer el periodo utilizado por el eje X de cada gráfico, puede seleccionar las opciones que aparecen en la parte superior de la sección. Puede actualizar los datos con el botón de actualizar (↻). Para cada gráfico independiente, el botón de elipses verticales (⋮) abre un menú con opciones de CloudWatch.

Estas métricas y otras más están disponibles automáticamente como métricas básicas de supervisión en CloudWatch bajo el espacio de nombres `AWS/EKS`. Para obtener más información, consulte Supervisión básica y detallada en la Guía del usuario de Amazon CloudWatch. Para obtener métricas, visualización e información más detalladas, consulte Información sobre contenedores en la Guía del usuario de Amazon CloudWatch. O si prefiere la supervisión basada en Kubernetes, consulte the section called “Métricas de Prometheus”.

En la tabla que aparece a continuación se describen las métricas disponibles.

Métrica	Descripción
Solicitudes de APIServer	Las solicitudes por minuto realizadas al servidor de la API.
Total de solicitudes APIServer 4XX	El recuento de solicitudes al servidor de la API por minuto que tuvieron códigos de respuesta HTTP 4XX (errores del lado del cliente).
Total de solicitudes APIServer 5XX	El recuento de solicitudes al servidor de la API por minuto que tuvieron códigos de respuesta HTTP 5XX (errores del lado del servidor).
Total de solicitudes APIServer 429	El recuento de solicitudes al servidor de la API por minuto que tuvieron códigos de respuesta HTTP 429 (demasiadas solicitudes).
Tamaño del almacenamiento	El tamaño de la base de datos de almacenamiento (`etcd`).
Intentos del programador	La cantidad de intentos de programar pods por resultados “no programable”, “error” y “programado”.
Pods pendientes	El número de pods pendientes por tipo de cola: “activa”, “retroceso”, “no programable” y “restringida”.
Latencia de las solicitudes al servidor de la API	La latencia de las solicitudes al servidor de la API.
Solicitudes actuales sobre la marcha al servidor de la API	Las solicitudes actuales sobre la marcha para el servidor de la API.
Solicitudes de webhooks	Las solicitudes de webhook por minuto.
Rechazos de solicitudes de webhook	Recuento de solicitudes de webhook rechazadas.

Métrica	Descripción
P99 de latencia de solicitud de webhook	El percentil 99 de latencia de las solicitudes de webhooks externos de terceros.

Información sobre los registros de CloudWatch

La sección Información de registros de CloudWatch ofrece varias listas basadas en los registros de auditoría del plano de control. Los registros del plano de control de Amazon EKS deben estar activados para utilizar esta característica. Puede activarlos desde la sección Ver registros del plano de control en CloudWatch.

Cuando haya transcurrido el tiempo suficiente para recopilar datos, podrá Ejecutar todas las consultas o elegir Ejecutar consulta para una sola lista cada vez. Se incurrirá en un costo adicional de CloudWatch cada vez que ejecute consultas. Elija el periodo para el que desea ver los resultados que aparecen en la parte superior de la sección. Si desea ejercer un control más avanzado sobre cualquier consulta, puede elegir Ver en CloudWatch. De este modo, podrá actualizar una consulta en CloudWatch según sus necesidades.

Para obtener más información, consulte Análisis de datos de registro con Información de registros de CloudWatch en la Guía del usuario de los Registros de Amazon CloudWatch.

Visualización de los registros del plano de control en CloudWatch

Seleccione Administrar registro para actualizar los tipos de registro disponibles. Los registros tardan varios minutos en aparecer en los Registros de CloudWatch después de habilitar el registro. Cuando haya transcurrido el tiempo suficiente, elija cualquiera de los enlaces Ver que aparecen en esta sección para ir al registro correspondiente.

Para obtener más información, consulte the section called “Registros del plano de control”.

Información sobre clústeres

En la tabla Información sobre actualizaciones se indican los problemas y se recomiendan medidas correctivas, lo que acelera el proceso de validación para actualizar a nuevas versiones de Kubernetes. Amazon EKS analiza automáticamente los clústeres en función de una lista de posibles problemas que afectan a la actualización de la versión de Kubernetes. En la tabla Información sobre actualizaciones se enumeran las comprobaciones de aportes de información realizadas por Amazon EKS en relación con este clúster, junto con sus estados asociados.

Amazon EKS mantiene la lista de comprobaciones de información que se deben realizar y la actualiza periódicamente en función de las evaluaciones de los cambios en el proyecto de Kubernetes, así como de los cambios en el servicio de Amazon EKS vinculados a las nuevas versiones. La consola de Amazon EKS actualiza automáticamente el estado de cada aporte de información, que se puede ver en la columna de la última actualización.

Para obtener más información, consulte the section called “Información sobre clústeres”.

Problemas de estado de los nodos

El agente de supervisión de nodos de Amazon EKS lee automáticamente los registros de los nodos para detectar problemas de estado. Independientemente de la configuración de reparación automática, se notifican todos los problemas de estado de los nodos, lo que le permite investigarlos según sea necesario. Si un tipo de problema se presenta sin una descripción, puede consultar la descripción en su ventana emergente.

Al actualizar la página, los problemas resueltos desaparecerán de la lista. Si la reparación automática está habilitada, es posible que aparezcan temporalmente algunos problemas de estado que se resolverán sin que se tome ninguna medida. Los problemas que no son compatibles con la reparación automática pueden requerir que actúe manualmente, según el tipo.

Para que los problemas de estado de los nodos se notifiquen, el clúster debe utilizar el modo automático de Amazon EKS o contar con el complemento de agente de supervisión de nodos. Para obtener más información, consulte the section called “Estado de los nodos”.

Capacidades de EKS

En la sección Capacidades, se muestra el estado de los recursos de la capacidad de EKS del clúster. Las notificaciones de estado de ambas capacidades y sus recursos de Kubernetes administrados del clúster se pueden supervisar aquí. Al actualizar la página, los problemas resueltos desaparecerán de la lista.

Para obtener más información, consulte the section called “Uso de capacidades”.

Supervisión del tráfico de cargas de trabajo de Kubernetes con la observabilidad de la red de contenedores

Amazon EKS ofrece características de observabilidad de la red mejoradas que proporcionan información más detallada sobre el entorno de redes de contenedores. Estas capacidades lo ayudan

a comprender, supervisar y solucionar mejor los problemas de su entorno de red de Kubernetes en AWS. Con una observabilidad de la red de contenedores mejorada, puede aprovechar las métricas detalladas relacionadas con la red para detectar mejor y proactivamente las anomalías en el tráfico de clústeres, los flujos entre zonas de disponibilidad y los servicios de AWS. Con estas métricas, puede medir el rendimiento del sistema y visualizar las métricas subyacentes con su pila de observabilidad preferida.

Además, Amazon EKS ahora ofrece visualizaciones de supervisión de la red en la consola de AWS que aceleran y mejoran la solución de problemas precisa para un análisis más rápido de la causa raíz. También puede aprovechar estas capacidades visuales para identificar los recursos con un mayor consumo y los flujos de red que causan las retransmisiones y los tiempos de espera de las retransmisiones, lo que elimina los puntos ciegos durante los incidentes.

Amazon CloudWatch Network Flow Monitor permite el uso de estas capacidades.

Casos de uso

Medición del rendimiento de la red para detectar anomalías

Varios equipos estandarizan una pila de observabilidad que les permite medir el rendimiento del sistema, visualizar las métricas del sistema y alarmarse en caso de que se supere un umbral específico. La observabilidad de la red de contenedores en EKS se alinea con esto, ya que expone las métricas clave del sistema que puede extraer para ampliar la observabilidad del rendimiento de la red del sistema en los pods y nodos de trabajo.

Uso de las visualizaciones de la consola para una solución de problemas más precisa

En caso de que se active una alarma en su sistema de supervisión, puede que desee centrarse en el clúster y la carga de trabajo en los que se originó el problema. Para ello, puede usar las visualizaciones de la consola de EKS, que reducen el ámbito de la investigación al clúster y aceleran la divulgación de los flujos de red responsables de la mayoría de las retransmisiones, los tiempos de espera de las retransmisiones y el volumen de datos transferidos.

Seguimiento de los recursos con mayor consumo en su entorno de Amazon EKS

Muchos equipos ejecutan EKS como base para sus plataformas, lo que lo convierte en el punto central de la actividad de red de un entorno de aplicaciones. Con las capacidades de supervisión de red de esta característica, puede hacer un seguimiento de las cargas de trabajo responsables de la mayor parte del tráfico (medido por el volumen de datos) dentro del clúster, entre zonas de

disponibilidad, así como del tráfico a destinos externos dentro de AWS (DynamoDB y S3) y fuera de la nube de AWS (Internet o en las instalaciones). Además, puede supervisar el rendimiento de cada uno de estos flujos en función de las retransmisiones, los tiempos de espera de las retransmisiones y los datos transferidos.

Características

1. Métricas de rendimiento: esta característica le permite extraer las métricas del sistema relacionadas con la red para los pods y los nodos de trabajo directamente desde el agente de Network Flow Monitor (NFM) que se ejecuta en su clúster de EKS.
2. Mapa de servicios: esta característica visualiza de forma dinámica la intercomunicación entre las cargas de trabajo del clúster, lo que le permite revelar rápidamente las métricas clave (retransmisiones [RT], tiempos de espera de retransmisiones [RTO] y datos transferidos [DT] asociadas a los flujos de red entre los módulos que se comunican.
3. Tabla de flujos: con esta tabla, puede supervisar los recursos con mayor consumo en las cargas de trabajo de Kubernetes de su clúster desde tres ángulos diferentes: vista de servicio de AWS, vista de clúster y vista externa. Para cada vista, puede ver las retransmisiones, los tiempos de espera de las retransmisiones y los datos transferidos entre el pod de origen y su destino.
   - Vista de servicio de AWS: muestra los recursos con mayor consumo de los servicios de AWS (DynamoDB y S3)
   - Vista de clúster: muestra los recursos con mayor consumo dentro del clúster (del este ← al → oeste)
   - Vista externa: muestra los recursos con mayor consumo de los destinos externos al clúster fuera de AWS

Introducción

Para comenzar, active la observabilidad de la red de contenedores en la consola de EKS para un clúster nuevo o existente. De este modo, se automatizará la creación de dependencias de Network Flow Monitor (NFM) (recursos de ámbito y supervisión). Además, tendrá que instalar el complemento Agente de Network Flow Monitor. Como alternativa, puede instalar estas dependencias mediante la `AWS CLI`, las API de EKS (para el complemento), las API de NFM o infraestructura como código (como Terraform). Una vez establecidas estas dependencias, puede configurar la herramienta de supervisión que prefiera para extraer las métricas de rendimiento de la red de los pods y nodos de trabajo desde el agente de NFM. Para visualizar la actividad de la red y el rendimiento de las cargas de trabajo, puede ir a la consola de EKS, en la pestaña “Red” del panel de observabilidad del clúster.

Al utilizar Network Flow Monitor en EKS, puede mantener su flujo de trabajo de observabilidad y la pila de tecnología actuales y, al mismo tiempo, aprovechar un conjunto de características adicionales que le permiten comprender y optimizar aún más la capa de red de su entorno de EKS. Puede obtener más información sobre los precios de Network Flow Monitor aquí.

Requisitos previos y notas importantes

1. Como se mencionó anteriormente, si habilita la observabilidad de la red de contenedores desde la consola de EKS, las dependencias de recursos de NFM subyacentes (ámbito y supervisión) se crearán automáticamente en su nombre y se le guiará durante el proceso de instalación del complemento de EKS para NFM.
2. Si desea activar esta característica mediante infraestructura como código (IaC) como Terraform, tendrá que definir las siguientes dependencias en su IaC: ámbito de NFM, supervisión de NFM, complemento de EKS para NFM. Además, tendrá que otorgar los permisos pertinentes al complemento de EKS mediante Pod Identity o roles de IAM para cuentas de servicio (IRSA).
3. Debe ejecutar como mínimo la versión 1.1.0 para el complemento de EKS del agente de NFM.
4. Debe utilizar la versión 6.21.0 o una posterior del Proveedor de AWS de Terraform para admitir los recursos de Network Flow Monitor.

Permisos de IAM necesarios

Complemento de EKS para el agente de NFM

Puede usar la política administrada de AWS `CloudWatchNetworkFlowMonitorAgentPublishPolicy` con Pod Identity. Esta política contiene permisos para que el agente de NFM envíe informes de telemetría (métricas) al punto de conexión de Network Flow Monitor.

```
{
  "Version" : "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "networkflowmonitor:Publish"
      ],
      "Resource" : "*"
    }
  ]
```

```
}
```

Observabilidad de la red de contenedores en la consola de EKS

Se requieren los siguientes permisos para activar la característica y visualizar el mapa de servicios y la tabla de flujos en la consola.

```
{
  "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": "*"
    }
  ]
}
```

Uso de la AWS CLI, la API de EKS y la API de NFM

```
#!/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})"
```

Uso de la infraestructura como código (IaC)

Terraform

Si utiliza Terraform para administrar su infraestructura en la nube de AWS, puede incluir las siguientes configuraciones de recursos para activar la observabilidad de la red de contenedores en su clúster.

Ámbito de NFM

```
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"
  }
```

```
}
```

Supervisión de NFM

```
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"
  }
}
```

Complemento de EKS para NFM

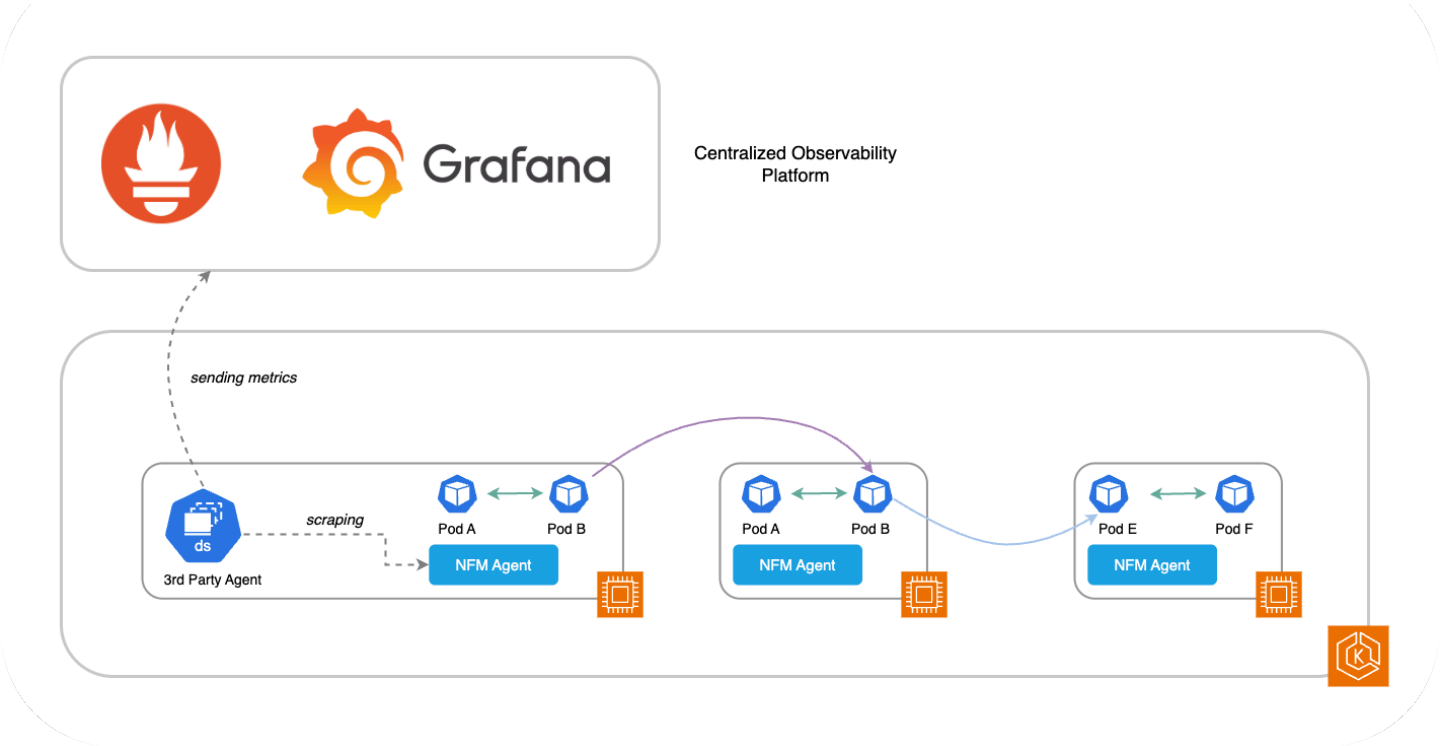
```
resource "aws_eks_addon" "example" {
  cluster_name                = aws_eks_cluster.example.name
  addon_name                  = "aws-network-flow-monitoring-agent"
}
```

¿Cómo funciona?

Métricas de desempeño

Métricas del sistema

Si ejecuta herramientas de terceros para supervisar su entorno de EKS (como Prometheus y Grafana), puede extraer las métricas del sistema compatibles directamente desde el agente de Network Flow Monitor. Estas métricas se pueden enviar a la pila de supervisión para ampliar la medición del rendimiento de la red del sistema en los pods y nodos de trabajo. Las métricas disponibles se muestran en la tabla, en Métricas del sistema compatibles.



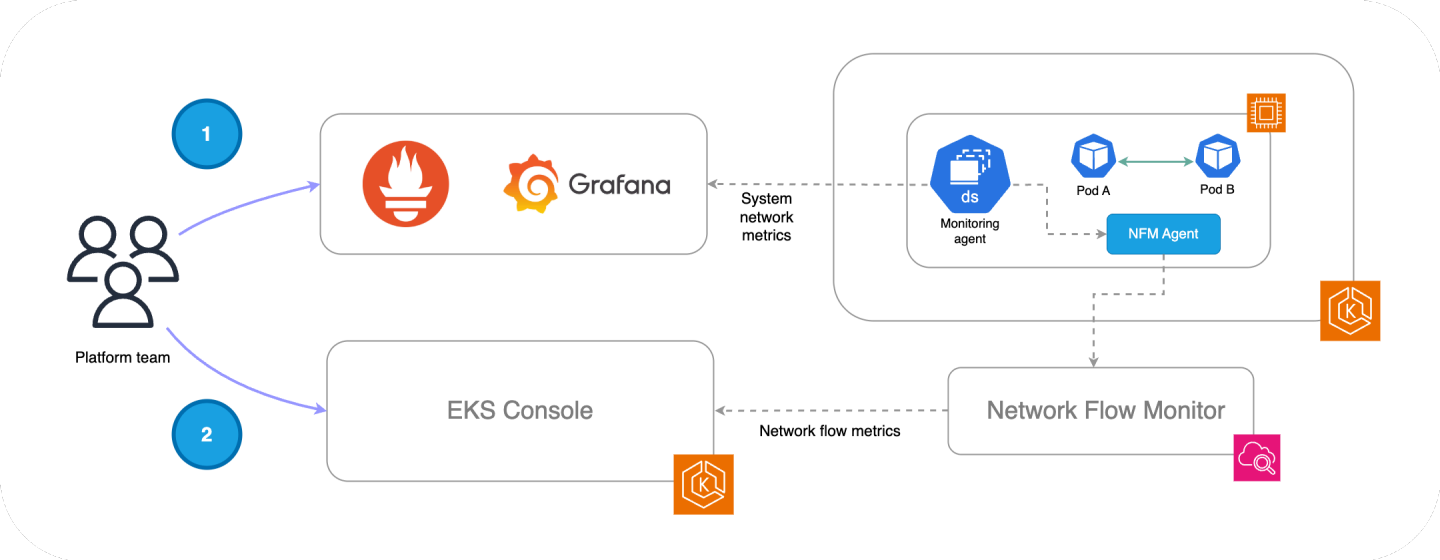
Para activar estas métricas, anule las siguientes variables de entorno mediante las variables de configuración durante el proceso de instalación (consulte: https://aws.amazon.com/blogs/containers/amazon-eks-add-ons-advanced-configuration/):

```
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]
```

Métricas de flujos

Además, Network Flow Monitor captura los datos de los flujos de red junto con las métricas de flujos: retransmisiones, tiempos de espera de las retransmisiones y datos transferidos. Network Flow Monitor procesa estos datos y los puede ver en la consola de EKS para detectar el tráfico en el entorno del clúster y su rendimiento en función de estas métricas de flujos.

En el siguiente diagrama se muestra un flujo de trabajo en el que se pueden aprovechar ambos tipos de métricas (del sistema y de flujos) para obtener más inteligencia operativa.



1. El equipo de la plataforma puede recopilar y visualizar las métricas del sistema en su pila de supervisión. Una vez implementadas las alertas, puede detectar anomalías en la red o problemas que afecten a pods o nodos de trabajo mediante las métricas del sistema del agente de NFM.
2. Como siguiente paso, los equipos de la plataforma pueden aprovechar las visualizaciones nativas de la consola de EKS para reducir aún más el ámbito de la investigación y acelerar la solución de problemas en función de las representaciones de los flujos y las métricas asociadas.

Nota importante: La extracción de las métricas del sistema del agente de NFM y el proceso por el que el agente de NFM envía las métricas de flujos al backend de NFM son procesos independientes.

Métricas del sistema compatibles

Nota importante: las métricas del sistema se exportan en formato OpenMetrics.

Nombre de métrica	Tipo	Dimensiones	Descripción
ingress_flow	Calibre	instance_id, iface, pod, espacio de nombres, nodo	Recuento de flujos de TCP de entrada (TcpPassiveOpens)

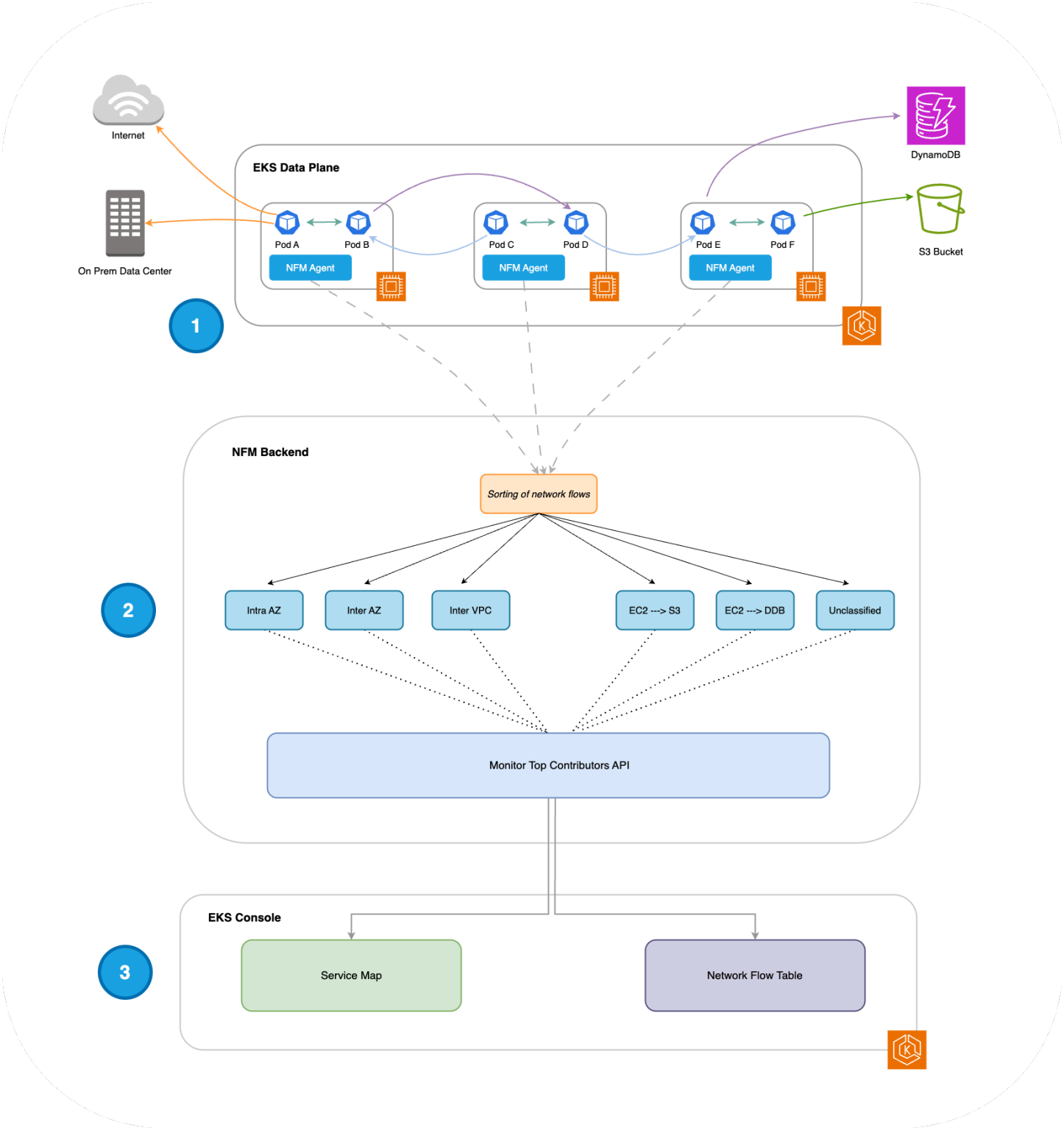
Nombre de métrica	Tipo	Dimensiones	Descripción
egress_flow	Calibre	instance_id, iface, pod, espacio de nombres, nodo	Recuento de flujos de TCP de salida (TcpActiveOpens)
ingress_packets	Calibre	instance_id, iface, pod, espacio de nombres, nodo	Recuento de paquetes de entrada (delta)
egress_packets	Calibre	instance_id, iface, pod, espacio de nombres, nodo	Recuento de paquetes de salida (delta)
ingress_bytes	Calibre	instance_id, iface, pod, espacio de nombres, nodo	Recuento de bytes de entrada (delta)
egress_bytes	Calibre	instance_id, iface, pod, espacio de nombres, nodo	Recuento de bytes de salida (delta)
bw_in_allowance_exceeded	Calibre	instance_id, eni, nodo	Paquetes puestos en cola o descartados debido al límite de ancho de banda de entrada
bw_out_allowance_exceeded	Calibre	instance_id, eni, nodo	Paquetes puestos en cola o descartados debido al límite de ancho de banda de salida
pps_allowance_exceeded	Calibre	instance_id, eni, nodo	Paquetes puestos en cola o descartados debido al límite de PPS bidireccional

Nombre de métrica	Tipo	Dimensiones	Descripción
conntrack_allowance_exceeded	Calibre	instance_id, eni, nodo	Paquetes descartados debido al límite de seguimiento de la conexión
linklocal_allowance_exceeded	Calibre	instance_id, eni, nodo	Paquetes descartados debido al límite de PPS del servicio de proxy local

Métricas de flujos compatibles

Nombre de métrica	Tipo	Descripción
Retransmisiones de TCP	Contador	Número de veces que un remitente reenvía un paquete que se perdió o se dañó durante la transmisión.
Tiempos de espera de las retransmisiones de TCP	Contador	Número de veces que un remitente inició un periodo de espera para determinar si un paquete se perdió en tránsito.
Datos (bytes) transferidos	Contador	Volumen de datos transferidos entre un origen y un destino para un flujo determinado.

Mapa de servicios y tabla de flujos

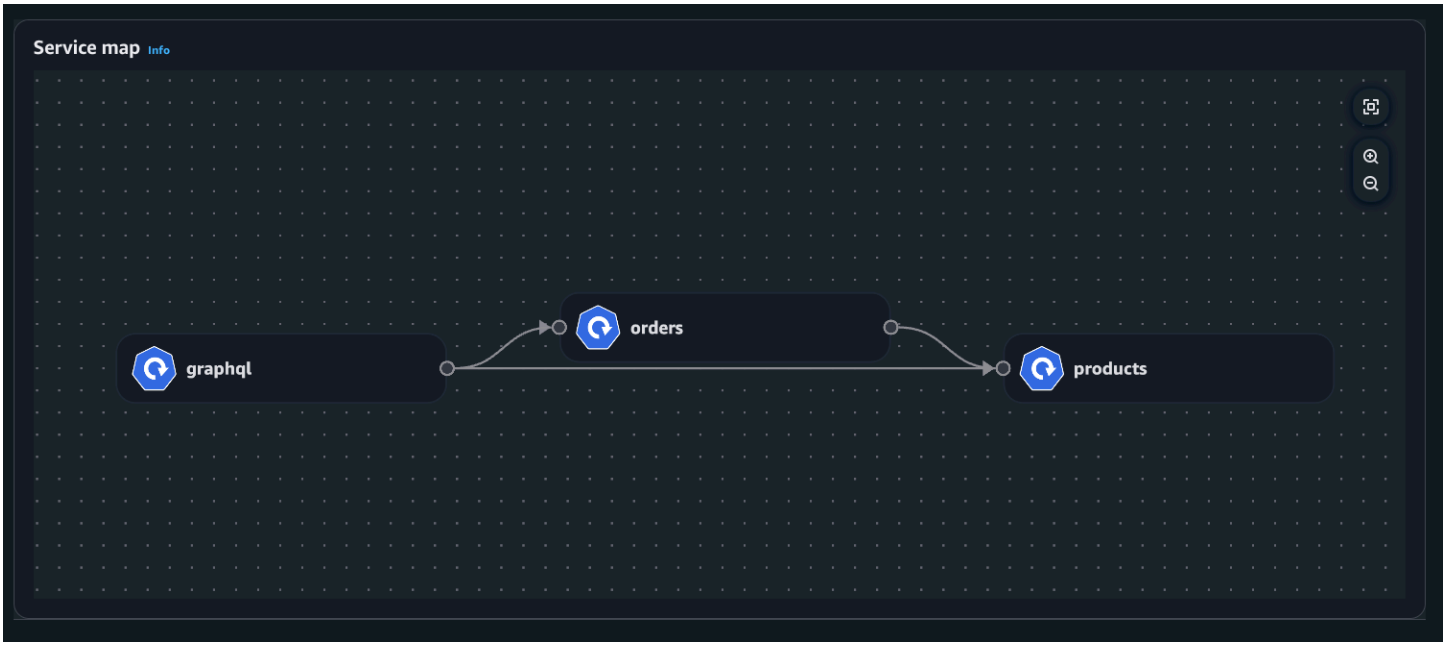


1. Una vez instalado, el agente de Network Flow Monitor se ejecuta como un DaemonSet en cada nodo de trabajo y recopila los 500 flujos de red principales (en función del volumen de datos transferidos) cada 30 segundos.
2. Estos flujos de red se clasifican en las siguientes categorías: Intra AZ, Inter AZ, EC2 → S3, EC2 → DynamoDB (DDB) y Sin clasificar. Cada flujo tiene 3 métricas asociadas: retransmisiones, tiempos de espera de las retransmisiones y datos transferidos (en bytes).
   - Intra AZ: flujos de red entre pods en la misma zona de disponibilidad
   - Inter AZ: flujos de red entre pods en diferentes zonas de disponibilidad
   - EC2 → S3: flujos de red de pods a S3
   - EC2 → DDB: flujos de red de pods a DDB
   - Sin clasificar: flujos de red de pods a Internet o en las instalaciones
3. Los flujos de red de la API de contribuidores principales de Network Flow Monitor se utilizan para mejorar las siguientes experiencias en la consola de EKS:
   - Mapa de servicios: visualización de los flujos de red dentro del clúster (Intra AZ e Inter AZ).
   - Tabla de flujos: presentación en una tabla de los flujos de red dentro del clúster (Intra AZ e Inter AZ), de pods a servicios de AWS (EC2 → S3 y EC2 → DDB) y de pods a destinos externos (sin clasificar).

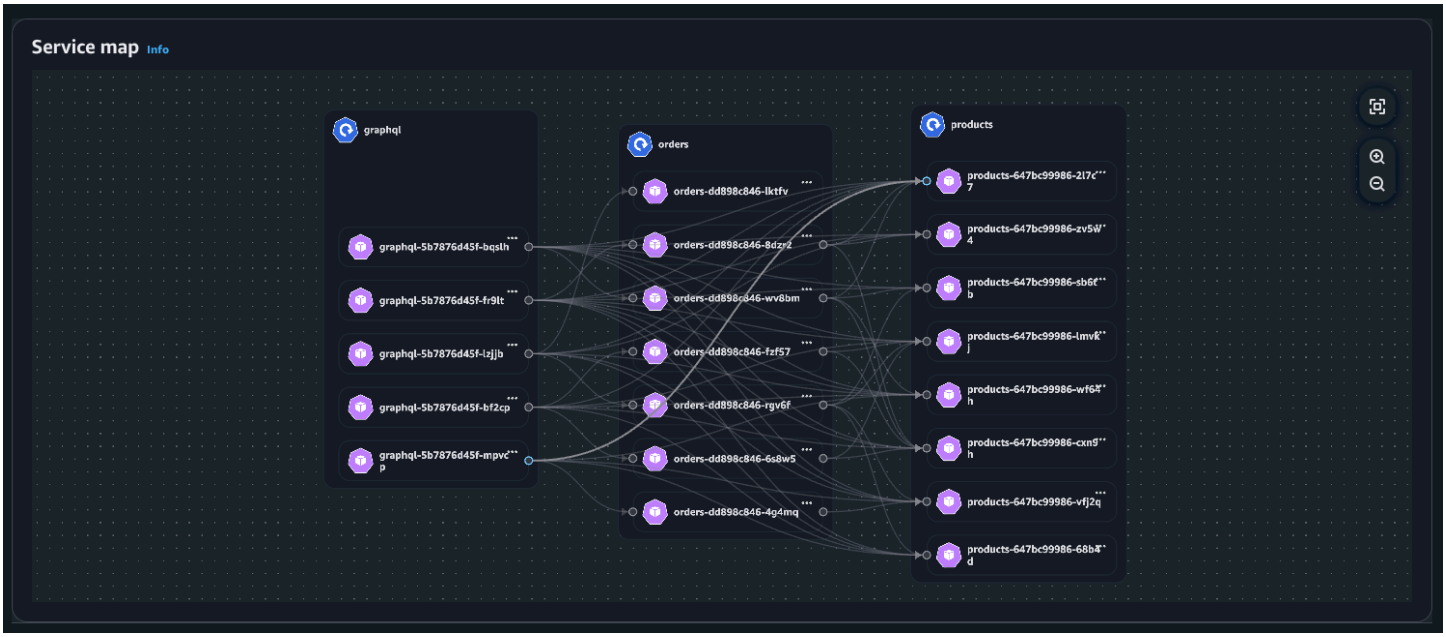
Los flujos de red extraídos de la API de contribuidores principales tienen un ámbito de intervalo de 1 hora y pueden incluir hasta 500 flujos de cada categoría. En el caso del mapa de servicios, esto significa que se pueden obtener y presentar hasta 1000 flujos de las categorías de flujos Intra AZ e Inter AZ en un intervalo de tiempo de 1 hora. En el caso de la tabla de flujos, esto significa que se pueden obtener y presentar hasta 3000 flujos de red de las 6 categorías de flujos de red en un intervalo de tiempo de 2 horas.

Ejemplo: mapa de servicios

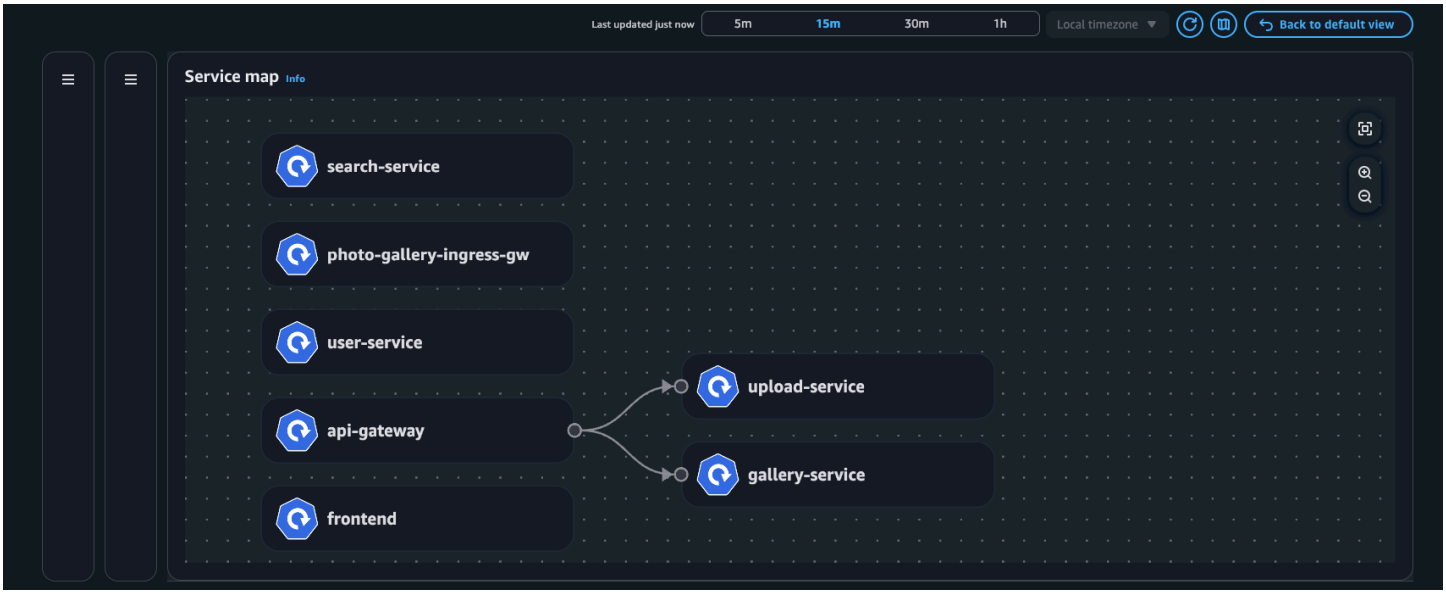
Vista de implementación



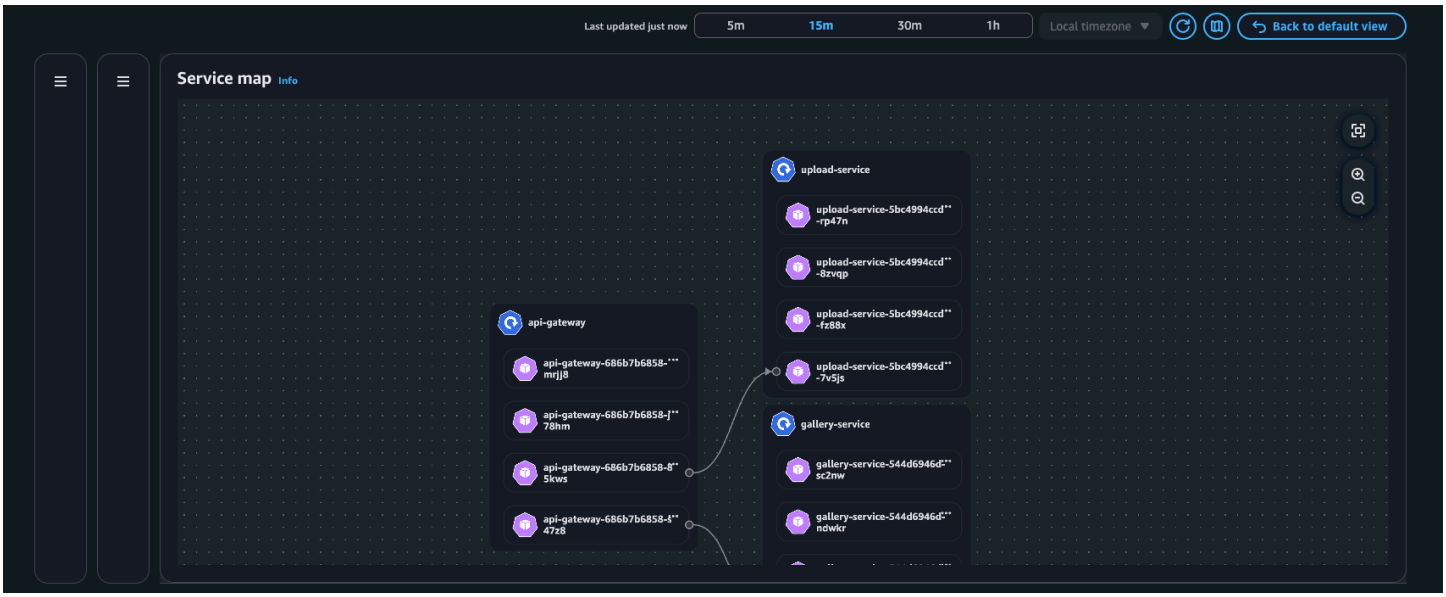
Vista de pod



Vista de implementación

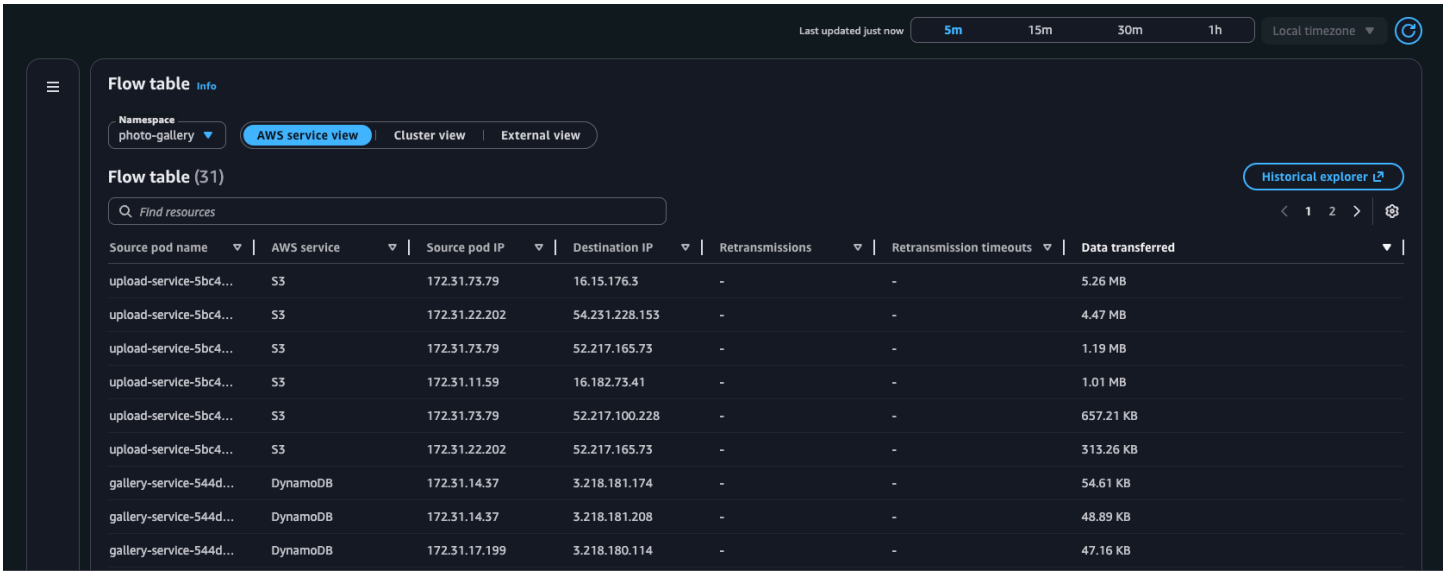


Vista de pod

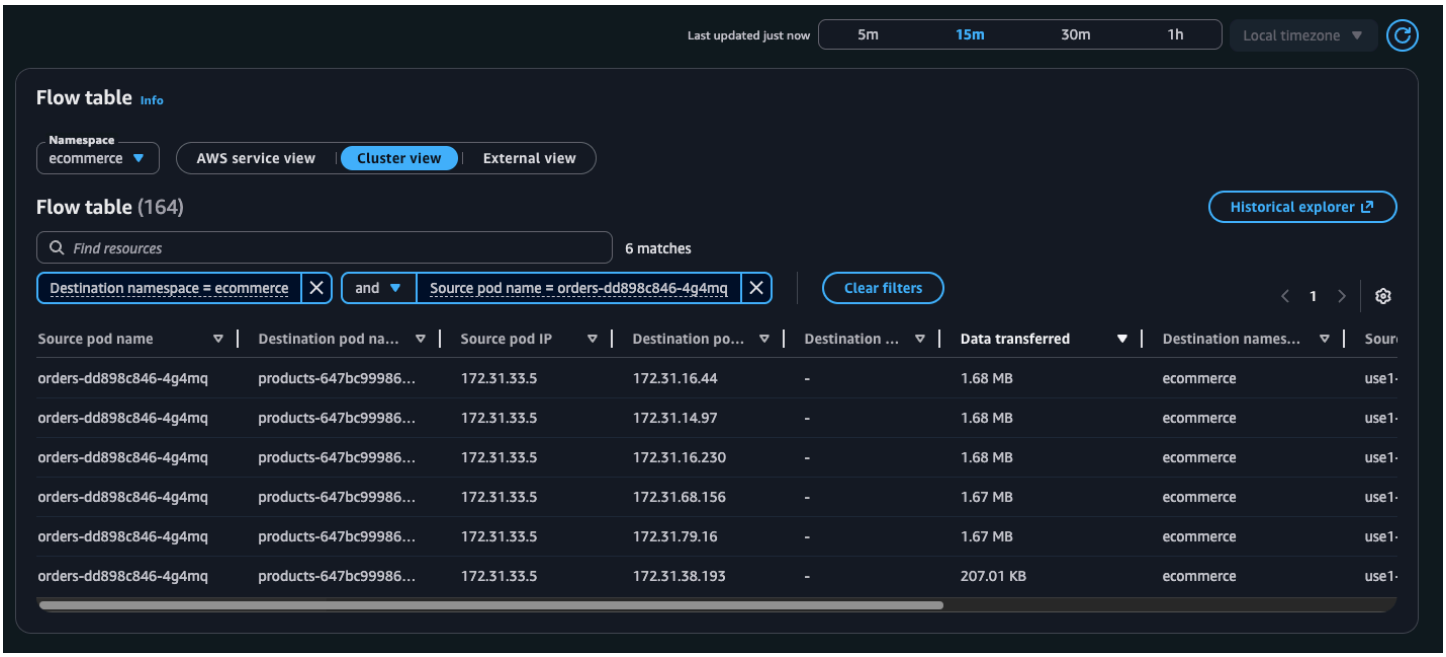


Ejemplo: tabla de flujos

Vista de servicio de AWS



Vista de clúster



Consideraciones y limitaciones

- La observabilidad de la red de contenedores en EKS solo está disponible en las regiones en las que se admite Network Flow Monitor.
- Las métricas del sistema compatibles están en formato OpenMetrics y se pueden extraer directamente del agente de Network Flow Monitor (NFM).

- Para activar la observabilidad de la red de contenedores en EKS mediante infraestructura como código (IaC) como Terraform, es necesario definir y crear estas dependencias en las configuraciones: el ámbito de NFM, la supervisión de NFM y el agente de NFM.
- Network Flow Monitor admite aproximadamente hasta 5 millones de flujos por minuto. Se trata de aproximadamente 5000 instancias de EC2 (nodos de trabajo de EKS) con el agente Network Flow Monitor instalado. La instalación de agentes en más de 5000 instancias puede afectar al rendimiento de la supervisión hasta que se disponga de capacidad adicional.
- Debe ejecutar como mínimo la versión 1.1.0 para el complemento de EKS del agente de NFM.
- Debe utilizar la versión 6.21.0 o una posterior del Proveedor de AWS de Terraform para admitir los recursos de Network Flow Monitor.
- Para enriquecer los flujos de red con metadatos de los pods, los pods deben ejecutarse en su propio espacio de nombres de red aislado, no en el espacio de nombres de la red host.

Supervisión de las métricas del clúster con Prometheus

Prometheus es una base de datos de serie temporal y de monitoreo que recopila puntos de conexión. Ofrece la posibilidad de consultar, agregar y almacenar los datos recopilados. También se puede utilizar para la generación de alertas y su agregación. En este tema se explica cómo configurar Prometheus como una opción administrada o de código abierto. La monitorización de las métricas del plano de control de Amazon EKS es un caso de uso común.

Amazon Managed Service for Prometheus es un servicio de supervisión y alertas compatible con Prometheus que facilita el monitoreo de las aplicaciones y la infraestructura en contenedores a escala. Es un servicio totalmente administrado que escala automáticamente la ingesta, el almacenamiento, las consultas y las alertas de sus métricas. También se integra con servicios de seguridad de AWS para permitir un acceso rápido y seguro a sus datos. Puede utilizar el lenguaje de consulta ProMQL de código abierto para consultar sus métricas y crear alertas sobre ellas. También puede utilizar el administrador de alertas en Amazon Managed Service para Prometheus para configurar reglas de alerta para alertas críticas. A continuación, puede enviar estas alertas críticas como notificaciones a un tema de Amazon SNS.

Existen varias opciones diferentes para el uso de Prometheus con Amazon EKS:

- Puede activar las métricas de Prometheus al crear un clúster de Amazon EKS por primera vez, o puede crear su propio raspador de Prometheus para clústeres existentes. En este tema se tratan estas dos opciones.

- Puede implementar Prometheus con Helm. Para obtener más información, consulte the section called “Implementación mediante Helm”.
- Puede ver las métricas sin procesar del plano de control en formato Prometheus. Para obtener más información, consulte the section called “Plano de control”.

Paso 1: activación de métricas de Prometheus

> **Important**
>
> Los recursos de Amazon Managed Service para Prometheus están fuera del ciclo de vida del clúster y deben mantenerse por fuera del clúster. Al eliminar el clúster, asegúrese de eliminar, también, cualquier raspador para reducir los costos aplicables. Para más información, consulte Búsqueda y eliminación de rapsadores en la Guía de usuario de Amazon Managed Service para Prometheus.

Prometheus descubre y recopila las métricas de su clúster mediante un modelo basado en la extracción denominado raspado. Los raspadores están configurados para recopilar datos de la infraestructura del clúster y de las aplicaciones en contenedores. Al activar la opción de enviar métricas de Prometheus, Amazon Managed Service para Prometheus proporciona un rastreador sin agentes totalmente administrado.

Si aún no ha creado el clúster, puede activar la opción de enviar métricas de Prometheus al crear el clúster. En la consola de Amazon EKS, esta opción se encuentra en el paso Configurar la observabilidad, que consiste en crear un clúster nuevo. Para obtener más información, consulte the section called “Creación de un clúster ”.

Si ya tiene un clúster, puede crear su propio raspador de Prometheus. Para ello, en la consola de Amazon EKS, vaya a la pestaña Observabilidad del clúster y elija el botón Añadir raspador. Si prefiere hacerlo con la API de AWS o la AWS CLI, consulte Create a scraper en la Guía del usuario de Amazon Managed Service para Prometheus.

Las siguientes opciones están disponibles al crear el raspador con la consola de Amazon EKS.

Alias de raspador

(Opcional) Escriba un alias único para el raspador.

Destino

Elija un espacio de trabajo de Amazon Managed Service para Prometheus. Un espacio de trabajo es un espacio lógico dedicado al almacenamiento y la consulta de las métricas de Prometheus. Con este espacio de trabajo, podrá ver las métricas de Prometheus de las cuentas que tienen acceso a él. La opción Crear un nuevo espacio de trabajo indica a Amazon EKS que cree un espacio de trabajo en su nombre con el alias de espacio de trabajo que proporcione. Con la opción Seleccionar un espacio de trabajo existente, puede seleccionar un espacio de trabajo existente de una lista desplegable. Para obtener más información sobre los espacios de trabajo, consulte Gestión de espacios de trabajo en la Guía del usuario de Amazon Managed Service for Prometheus.

Acceso a los servicios

En esta sección se resumen los permisos que se conceden al enviar métricas de Prometheus:

- Permiso para que Amazon Managed Service for Prometheus describa el clúster de Amazon EKS
- Permiso para la escritura remota en el espacio de trabajo de Amazon Managed para Prometheus

Si el `AmazonManagedScraperRole` ya existe, el raspador lo usa. Elija el enlace de `AmazonManagedScraperRole` para ver los detalles del permiso. Si el `AmazonManagedScraperRole` aún no existe, seleccione el enlace Ver detalles del permiso para ver los permisos específicos que está concediendo mediante el envío de las métricas de Prometheus.

Subredes

Modifique las subredes que heredará el raspador según lo que necesite. Si necesita agregar una opción de subred atenuada, vuelva al paso de creación del clúster de Especificar red.

Configuración del raspador

Modifique la configuración del raspador en formato YAML según sea necesario. Para ello, utilice el formulario o cargue un archivo YAML de reemplazo. Para obtener más información, consulte Configuración del raspador en la Guía del usuario de Amazon Managed Service for Prometheus.

Amazon Managed Service para Prometheus hace referencia al rastreador sin agente que se crea junto con el clúster como recopilador administrado de AWS. Para obtener más información sobre los

recopiladores administrados por AWS, consulte Ingesta de métricas con recopiladores administrados por AWS en la Guía del usuario de Amazon Managed Service para Prometheus.

> **Important**
>
> - Si crea un raspador de Prometheus mediante la AWS CLI o la API de AWS, debe ajustar su configuración para otorgarle permisos dentro del clúster. Para obtener más información, consulte Configuración del clúster de Amazon EKS en la Guía del usuario de Amazon Managed Service for Prometheus.
> - Si tiene un raspador de Prometheus creado antes del 11 de noviembre de 2024 que utilice el `ConfigMap` `aws-auth` en lugar de entradas de acceso, deberá actualizarlo para obtener acceso a métricas adicionales del plano de control del clúster de Amazon EKS. Para obtener la configuración actualizada, consulte Configuración manual de Amazon EKS para el acceso del sistema de extracción en la Guía del usuario de Amazon Managed Service para Prometheus.

Paso 2: uso de métricas de Prometheus

Para obtener más información sobre cómo utilizar las métricas de Prometheus después de activarlas en su clúster, consulte la Guía del usuario de Amazon Managed Service para Prometheus.

Paso 3: administración de raspadores de Prometheus

Para administrar raspadores, elija la pestaña Observabilidad en la consola de Amazon EKS. En una tabla se muestra una lista de los raspadores del clúster, que incluye información como el ID, el alias, el estado y la fecha de creación del raspador. Puede agregar más raspadores, editarlos, eliminarlos o ver más información sobre los raspadores actuales.

Para ver más detalles sobre un raspador, elija el enlace de ID del raspador. Por ejemplo, puede ver el ARN, el entorno, el ID del espacio de trabajo, el rol de IAM, la configuración y la información de la red. Puede usar el ID de raspador como entrada en Amazon Managed Service para operaciones de la API de Prometheus, como `DescribeScraper`, `UpdateScraper` y `DeleteScraper`. Para obtener más información sobre el uso de la API de Prometheus, consulte la referencia de la API de Amazon Managed Service para Prometheus.

Implementación de Prometheus con Helm

Como alternativa al uso de Amazon Managed Service para Prometheus, puede implementar Prometheus en su clúster con Helm. Si ya tiene instalado Helm, puede comprobar su versión con el comando `helm version`. Helm es un administrador de paquetes para los clústeres de Kubernetes. Para obtener más información sobre Helm y sobre cómo instalarlo, consulte the section called "Implementación de aplicaciones con Helm".

Después de configurar Helm para su clúster de Amazon EKS, puede utilizarlo para implementar Prometheus con los pasos que se describen a continuación.

1. Cree un espacio de nombres para Prometheus.

   ```
   kubectl create namespace prometheus
   ```

2. Agregue el repositorio de gráficos de `prometheus-community`.

   ```
   helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
   ```

3. Implemente Prometheus.

   ```
   helm upgrade -i prometheus prometheus-community/prometheus \
       --namespace prometheus \
       --set alertmanager.persistence.storageClass="gp2" \
       --set server.persistentVolume.storageClass="gp2"
   ```

   > **Note**
   >
   > Si recibe el error `Error: failed to download "stable/prometheus" (hint: running helm repo update may help)` al ejecutar este comando, ejecute `helm repo update prometheus-community` y, a continuación, vuelva a ejecutar el comando del Paso 2.

   Si recibe el error `Error: rendered manifests contain a resource that already exists`, ejecute `helm uninstall` *`your-release-name`* `-n` *`namespace`* y, a continuación, vuelva a ejecutar el comando del Paso 3.

4. Compruebe que todos los pods en el espacio de nombres de `prometheus` se encuentran en estado `READY`.

```
kubectl get pods -n prometheus
```

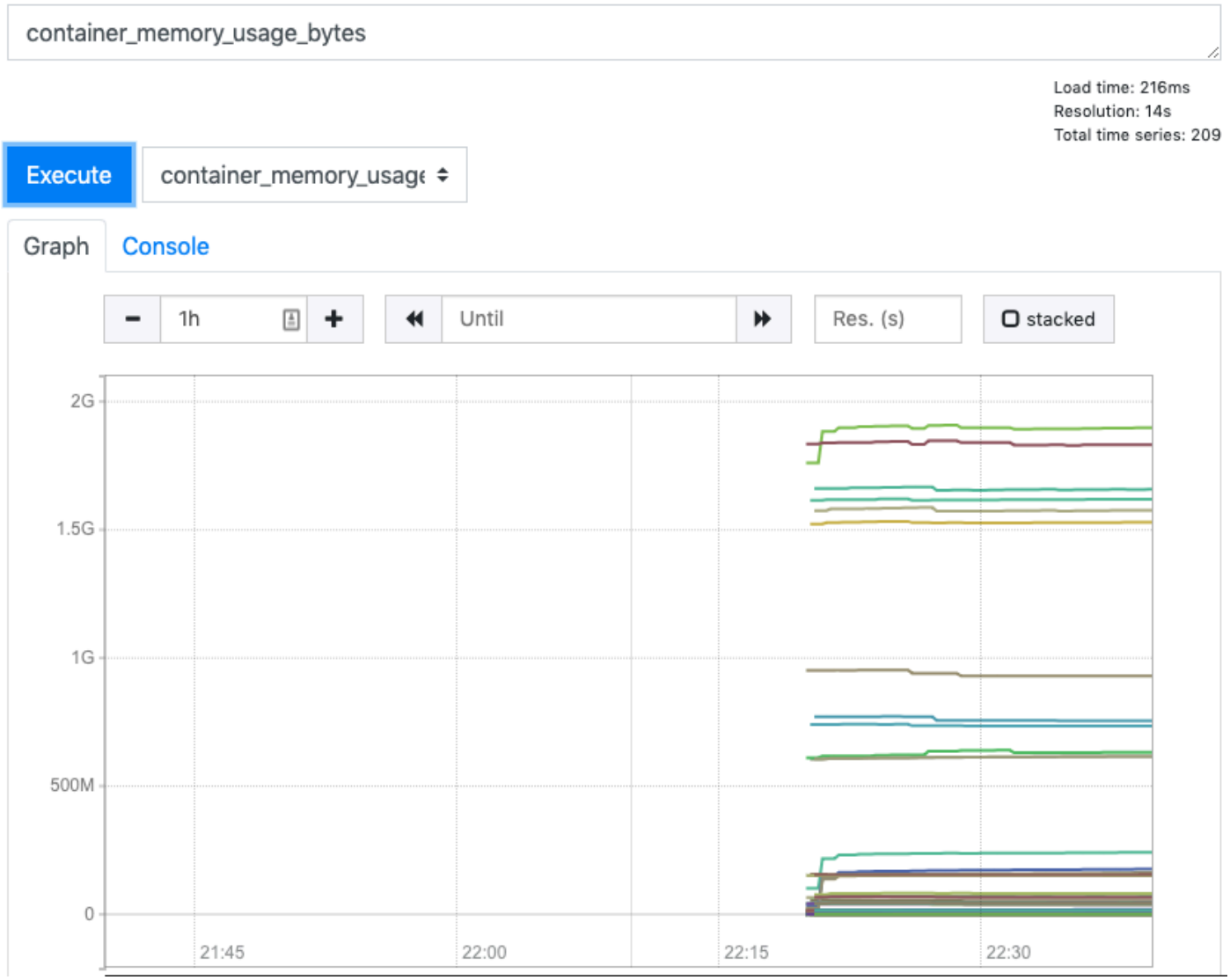
Un ejemplo de salida sería el siguiente.

```
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
```

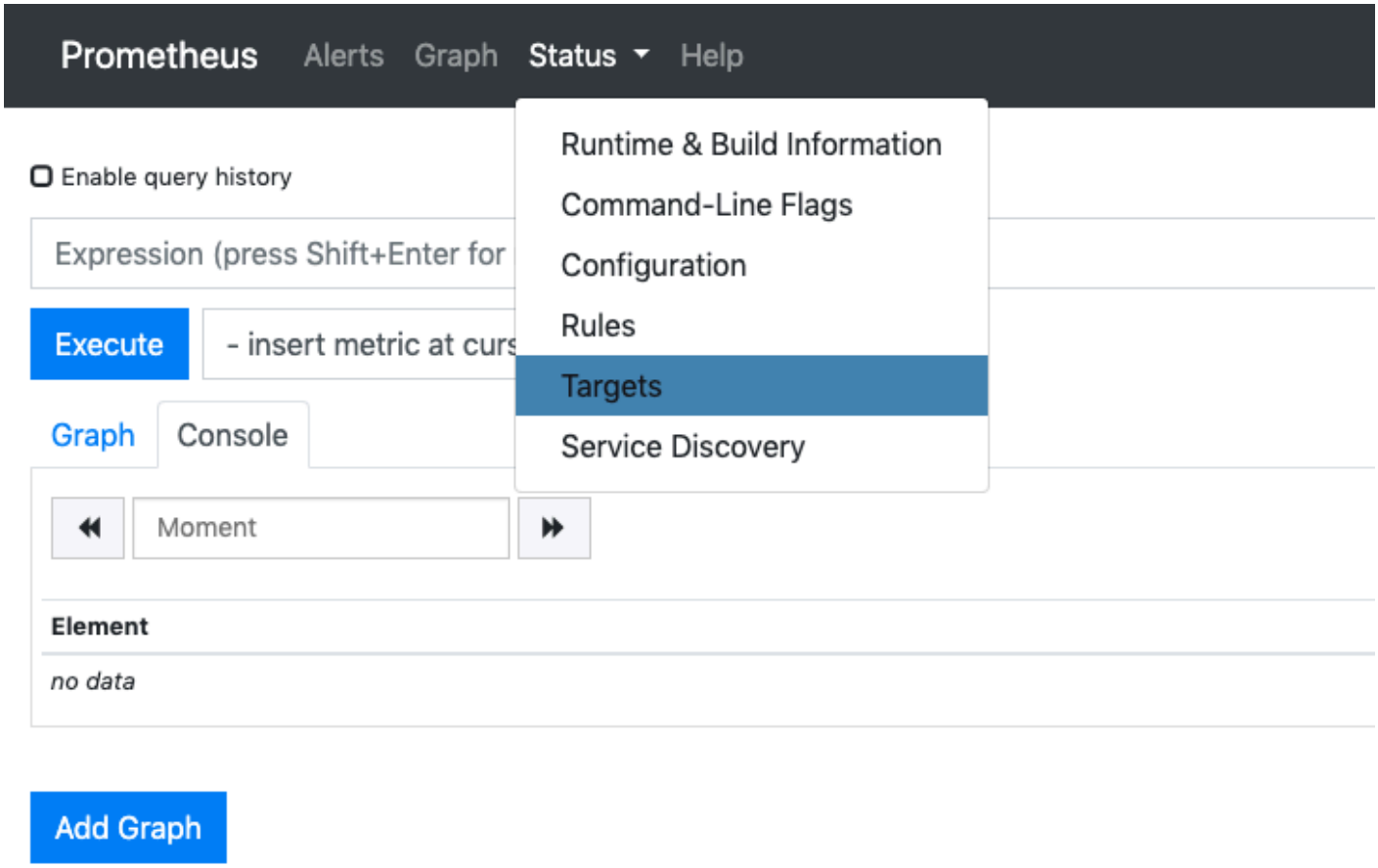
5. Utilice `kubectl` para el enrutamiento del puerto de la consola de Prometheus a su equipo local.

```
kubectl --namespace=prometheus port-forward deploy/prometheus-server 9090
```

6. Apunte un navegador web a `http://localhost:9090` para ver la consola de Prometheus.
7. Elija una métrica del menú - insert metric at cursor (- insertar métrica en el cursor) y elija Execute (Ejecutar). Elija la pestaña Graph (Gráfico) para mostrar la métrica con el paso del tiempo. La siguiente imagen muestra `container_memory_usage_bytes` a lo largo del tiempo.



8. En la barra de navegación superior, elija Status (Estado) y Targets (Destinos).



Se muestran todos los puntos de enlace de Kubernetes que están conectados a Prometheus mediante detección de servicios.

Obtención de métricas sin procesar del plano de control en formato Prometheus

El plano de control de Kubernetes expone una serie de métricas que se representan en un formato Prometheus. Estas métricas son útiles para el monitoreo y el análisis. Se exponen internamente a través de puntos de conexión de métricas, y se puede acceder a estos sin implementar Prometheus completamente. Sin embargo, la implementación de Prometheus permite analizar las métricas a lo largo del tiempo de forma más sencilla.

Para ver la salida de métricas sin procesar, reemplace `endpoint` y ejecute el siguiente comando.

```
kubectl get --raw endpoint
```

Este comando permite pasar cualquier ruta de punto de conexión y devuelve la respuesta sin procesar. La salida enumera las diferentes métricas línea por línea, y cada línea incluye un nombre de métrica, etiquetas y un valor.

```
metric_name{tag="value"[,...]} value
```

Cómo obtener métricas del servidor de API

El punto de conexión general del servidor de API se expone en el plano de control de Amazon EKS. Este punto de conexión resulta útil sobre todo cuando se examina una métrica específica.

```
kubectl get --raw /metrics
```

Un ejemplo de salida sería el siguiente.

```
[...]
# 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
```

Este resultado sin procesar devuelve literalmente lo que el servidor de API expone.

Métricas del plano de control con **`metrics.eks.amazonaws.com`**

En el caso de los clústeres de la versión `1.28` y posteriores de Kubernetes, Amazon EKS también expone métricas bajo el grupo de la API `metrics.eks.amazonaws.com`. Estas métricas incluyen componentes del plano de control, como `kube-scheduler` y `kube-controller-manager`.

> **Note**
>
> Si tiene una configuración de webhook que podría bloquear la creación de la nueva `v1.metrics.eks.amazonaws.com` del recurso `APIService` en el clúster, es posible que la característica de punto de conexión de métricas no se encuentre disponible. Para verificarlo en el registro de auditoría de `kube-apiserver`, busque la palabra clave `v1.metrics.eks.amazonaws.com`.

Cómo obtener métricas de **`kube-scheduler`**

Utilice el siguiente comando para obtener métricas de `kube-scheduler`

```
kubectl get --raw "/apis/metrics.eks.amazonaws.com/v1/ksh/container/metrics"
```

Un ejemplo de salida sería el siguiente.

```
# 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
[...]
```

Cómo obtener métricas de **`kube-controller-manager`**

Utilice el siguiente comando para obtener métricas de `kube-controller-manager`

```
kubectl get --raw "/apis/metrics.eks.amazonaws.com/v1/kcm/container/metrics"
```

Un ejemplo de salida sería el siguiente.

```
[...]
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
[...]
```

Comprender las métricas del programador y del administrador de controladores

En la siguiente tabla se describen las métricas del programador y del administrador de controladores que están disponibles para la extracción de estilos de Prometheus. Para obtener más información acerca de estas métricas, consulte Kubernetes Metrics Reference en la documentación de Kubernetes.

Métrica	Componente del plano de control	Descripción
scheduler_pending_pods	programador	La cantidad de pods que están a la espera de programarse en un nodo para su ejecución.
scheduler_schedule_attempts_total	programador	La cantidad de intentos realizados para programar pods.
scheduler_preemption_attempts_total	programador	La cantidad de intentos realizados por el programador para programar pods de mayor prioridad expulsando los de menor prioridad.

Métrica	Componente del plano de control	Descripción
scheduler_preemption_victims	programador	La cantidad de pods que se han seleccionado para expulsarse con el fin de liberar espacio para pods de mayor prioridad.
scheduler_pod_scheduling_attempts	programador	La cantidad de intentos de programar un pod correctamente.
scheduler_scheduling_attempt_duration_seconds	programador	Indica la velocidad con la que el programador es capaz de encontrar un espacio adecuado para la ejecución de un pod en función de varios factores, como la disponibilidad de recursos y las reglas de programación.
scheduler_pod_scheduling_sli_duration_seconds	programador	La latencia de extremo a extremo de un pod que se está programando, desde el momento en que entra en la cola de programación. Es posible que sean necesarios varios intentos de programación.
cronjob_controller_job_creation_skew_duration_seconds	administrador de controladores	El tiempo transcurrido entre el momento en que se programa la ejecución de un cronjob y el momento en que se crea el trabajo correspondiente.

Métrica	Componente del plano de control	Descripción
workqueue_depth	administrador de controladores	La profundidad actual de la cola.
workqueue_adds_total	administrador de controladores	Cantidad total de adiciones administradas por la cola de trabajo.
workqueue_queue_duration_seconds	administrador de controladores	El tiempo en segundos que un elemento permanece en la cola de trabajo antes de ser solicitado.
workqueue_work_duration_seconds	administrador de controladores	El tiempo en segundos que tarda en procesarse un elemento de la cola de trabajo.

Implementación de un sistema de extracción de Prometheus para extraer métricas de forma sistemática

Para implementar un sistema de extracción de Prometheus que extraiga sistemáticamente las métricas, utilice la siguiente configuración:

```
---
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
```

El permiso que sigue es necesario para que el pod acceda al nuevo punto de conexión de métricas.

```
{
  "effect": "allow",
  "apiGroups": [
    "metrics.eks.amazonaws.com"
  ],
  "resources": [
    "kcm/metrics",
    "ksh/metrics"
  ],
  "verbs": [
    "get"
  ] },
```

Para aplicar un parche al rol que se utiliza, puede utilizar el siguiente comando.

```
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"]
    }
  }
]'
```

A continuación, para ver el panel de Prometheus, puede redirigir mediante proxy el puerto del sistema de extracción de Prometheus al puerto local.

```
kubectl port-forward pods/prom-pod 9090:9090
```

Para el clúster de Amazon EKS, las métricas del plano de control principal de Kubernetes también se ingieren en las métricas de Amazon CloudWatch bajo el espacio de nombres de `AWS/EKS`. Para verlas, abra la consola de CloudWatch y seleccione Todas las métricas en el panel de navegación izquierdo. En la página de selección de Métricas, elija el espacio de nombres de `AWS/EKS` y una dimensión de métricas para el clúster.

Supervisión de datos de clústeres con Amazon CloudWatch

Amazon CloudWatch es un servicio de supervisión que recopila métricas y registros de los recursos en la nube. CloudWatch proporciona algunas métricas básicas de Amazon EKS de forma gratuita cuando se utiliza un clúster nuevo de la versión `1.28` o posterior. Sin embargo, al utilizar el operador de observabilidad de CloudWatch como complemento de Amazon EKS, se obtienen características de observabilidad mejoradas.

Métricas básicas en Amazon CloudWatch

En el caso de clústeres de la versión `1.28` o posterior de Kubernetes, obtendrá métricas suministradas por CloudWatch de forma gratuita en el espacio de nombres de `AWS/EKS`. En la siguiente tabla se presenta una lista de las métricas básicas disponibles para las versiones compatibles. Cada métrica que aparece en la lista tiene una frecuencia de un minuto.

Nombre de métrica	Descripción
apiserver_flowcontrol_current_executing_seats	El número de plazas que se utilizan actualmente para ejecutar las solicitudes de la API. La asignación de plazas se determina mediante las configuraciones priority_level y flow_schema de la característica de prioridad y equidad de la API de Kubernetes. Unidades: recuento Estadísticas válidas: Máximo
scheduler_schedule_attempts_total	El número total de intentos del programador para programar los pods en el clúster durante un periodo determinado. Esta métrica ayuda a supervisar la carga de trabajo del programador y puede indicar la presión de programación o los posibles problemas relacionados con la ubicación de los pods. Unidades: recuento Estadísticas válidas: Sum
scheduler_schedule_attempts_SCHEDULED	El número de intentos correctos del programador para programar los pods en los nodos del clúster durante un periodo determinado. Unidades: recuento Estadísticas válidas: Sum
scheduler_schedule_attempts_UNSCHEDULABLE	El número de intentos para programar pods que no se pudieron programar durante un periodo determinado debido a restricciones válidas, como la falta de CPU o memoria en un nodo.

Nombre de métrica	Descripción
	Unidades: recuento Estadísticas válidas: Sum
scheduler_schedule_attempts_ERROR	El número de intentos para programar pods que no se pudieron programar durante un periodo determinado debido a un problema interno del propio programador, como problemas de conectividad con el servidor de API. Unidades: recuento Estadísticas válidas: Sum
scheduler_pending_pods	El número total de pods pendientes que programará el programador en el clúster durante un periodo determinado. Unidades: recuento Estadísticas válidas: Sum
scheduler_pending_pods_ACTIVEQ	El número de pods pendientes en ActiveQ que están esperando su programación en el clúster durante un periodo determinado. Unidades: recuento Estadísticas válidas: Sum
scheduler_pending_pods_UNSCHEDULABLE	El número de pods pendientes que el programador intentó programar y fallaron, y que se mantienen en un estado no programable para volver a intentarlo. Unidades: recuento Estadísticas válidas: Sum

Nombre de métrica	Descripción
scheduler_pending_pods_BACKOFF	El número de pods pendientes en backoffQ en estado de retroceso a la espera de que venza su periodo de retroceso. Unidades: recuento Estadísticas válidas: Sum
scheduler_pending_pods_GATED	El número de pods pendientes que se encuentran actualmente en espera en un estado cerrado, ya que no se pueden programar hasta que cumplan las condiciones requeridas. Unidades: recuento Estadísticas válidas: Sum
apiserver_request_total	El número de solicitudes HTTP hechas en todos los servidores de API del clúster. Unidades: recuento Estadísticas válidas: Sum
apiserver_request_total_4XX	El número de solicitudes HTTP hechas a todos los servidores de API del clúster que generaron códigos de estado 4XX (error del cliente). Unidades: recuento Estadísticas válidas: Sum

Nombre de métrica	Descripción
apiserver_request_total_429	El número de solicitudes HTTP hechas a todos los servidores de API del clúster que generaron un código de estado 429, que se produce cuando los clientes superan los límites de umbrales de frecuencia. Unidades: recuento Estadísticas válidas: Sum
apiserver_request_total_5XX	El número de solicitudes HTTP hechas a todos los servidores de API del clúster que generaron códigos de estado 5XX (error del servidor). Unidades: recuento Estadísticas válidas: Sum
apiserver_request_total_LIST_PODS	El número de solicitudes pods LIST hechas a todos los servidores de API del clúster. Unidades: recuento Estadísticas válidas: Sum
apiserver_request_duration_seconds_PUT_P99	El percentil 99 de latencia de las solicitudes PUT se calcula a partir de todas las solicitudes de todos los servidores de API del clúster. Representa el tiempo de respuesta por debajo del cual se completa el 99 % de todas las solicitudes PUT. Unidades: segundos Estadísticas válidas: Promedio

Nombre de métrica	Descripción
apiserver_request_duration_seconds_PATCH_P99	El percentil 99 de latencia de las solicitudes PATCH se calcula a partir de todas las solicitud es de todos los servidores de API del clúster. Representa el tiempo de respuesta por debajo del cual se completa el 99 % de todas las solicitudes PATCH. Unidades: segundos Estadísticas válidas: Promedio
apiserver_request_duration_seconds_POST_P99	El percentil 99 de latencia de las solicitudes POST se calcula a partir de todas las solicitud es de todos los servidores de API del clúster. Representa el tiempo de respuesta por debajo del cual se completa el 99 % de todas las solicitudes POST. Unidades: segundos Estadísticas válidas: Promedio
apiserver_request_duration_seconds_GET_P99	El percentil 99 de latencia de las solicitudes GET se calcula a partir de todas las solicitud es de todos los servidores de API del clúster. Representa el tiempo de respuesta por debajo del cual se completa el 99 % de todas las solicitudes GET. Unidades: segundos Estadísticas válidas: Promedio

Nombre de métrica	Descripción
apiserver_request_duration_seconds_LIST_P99	El percentil 99 de latencia de las solicitudes LIST se calcula a partir de todas las solicitudes de todos los servidores de API del clúster. Representa el tiempo de respuesta por debajo del cual se completa el 99 % de todas las solicitudes LIST. Unidades: segundos Estadísticas válidas: Promedio
apiserver_request_duration_seconds_DELETE_P99	El percentil 99 de latencia de las solicitudes DELETE se calcula a partir de todas las solicitudes de todos los servidores de API del clúster. Representa el tiempo de respuesta por debajo del cual se completa el 99 % de todas las solicitudes DELETE. Unidades: segundos Estadísticas válidas: Promedio
apiserver_current_inflight_requests_MUTATING	El número de solicitudes mutantes (POST, PUT, DELETE, PATCH) que se están procesando actualmente en todos los servidores de API del clúster. Esta métrica representa las solicitudes en tránsito y que aún no se han procesado. Unidades: recuento Estadísticas válidas: Sum

Nombre de métrica	Descripción
`apiserver_current_inflight_requests_READONLY`	El número de solicitudes de solo lectura (`GET`, `LIST`) que se están procesando actualmente en todos los servidores de API del clúster. Esta métrica representa las solicitudes en tránsito y que aún no se han procesado. Unidades: recuento Estadísticas válidas: Sum
`apiserver_admission_webhook_request_total`	El número de solicitudes de webhook de admisión hechas en todos los servidores de API del clúster. Unidades: recuento Estadísticas válidas: Sum
`apiserver_admission_webhook_request_total_ADMIT`	El número de solicitudes mutantes de webhook de admisión hechas en todos los servidores de API del clúster. Unidades: recuento Estadísticas válidas: Sum
`apiserver_admission_webhook_request_total_VALIDATING`	El número de solicitudes de validación de webhook de admisión hechas en todos los servidores de API del clúster. Unidades: recuento Estadísticas válidas: Sum

Nombre de métrica	Descripción
`apiserver_admission_webhook_rejection_count`	El número de solicitudes de webhook de admisión hechas en todos los servidores de API del clúster que se rechazaron. Unidades: recuento Estadísticas válidas: Sum
`apiserver_admission_webhook_rejection_count_ADMIT`	El número de solicitudes mutantes de webhook de admisión hechas en todos los servidores de API del clúster que se rechazaron. Unidades: recuento Estadísticas válidas: Sum
`apiserver_admission_webhook_rejection_count_VALIDATING`	El número de solicitudes de validación de webhook de admisión hechas en todos los servidores de API del clúster que se rechazaro n. Unidades: recuento Estadísticas válidas: Sum
`apiserver_admission_webhook_admission_duration_seconds`	El percentil 99 de latencia de las solicitudes de webhook de admisión de terceros se calcula a partir de todas las solicitudes de todos los servidores de API del clúster. Representa el tiempo de respuesta por debajo del cual se completa el 99 % de todas las solicitudes de webhook de admisión de terceros. Unidades: segundos Estadísticas válidas: Promedio

Nombre de métrica	Descripción
apiserver_admission_webhook_admission_duration_seconds_ADMIT_P99	El percentil 99 de latencia de las solicitudes mutantes de webhook de admisión de terceros se calcula a partir de todas las solicitudes de todos los servidores de API del clúster. Representa el tiempo de respuesta por debajo del cual se completa el 99 % de todas las solicitudes mutantes de webhook de admisión de terceros. Unidades: segundos Estadísticas válidas: Promedio
apiserver_admission_webhook_admission_duration_seconds_VALIDATING_P99	El percentil 99 de latencia de las solicitud es de validación de webhook de admisión de terceros se calcula a partir de todas las solicitudes de todos los servidores de API del clúster. Representa el tiempo de respuesta por debajo del cual se completa el 99 % de todas las solicitudes de validación de webhook de admisión de terceros. Unidades: segundos Estadísticas válidas: Promedio
apiserver_storage_size_bytes	El tamaño físico en bytes del archivo de base de datos de almacenamiento etcd utilizado por los servidores de API del clúster. Esta métrica representa el espacio real en disco asignado al almacenamiento. Unidades: bytes Estadísticas válidas: Máximo

Amazon CloudWatch Observability Operator

Observabilidad de Amazon CloudWatch recopila datos de rastreo, métricas y registros en tiempo real. Los envía a Amazon CloudWatch y AWS X-Ray. Puede instalar este complemento para habilitar tanto CloudWatch Application Signals como CloudWatch Container Insights con una observabilidad mejorada para Amazon EKS. Esto le ayuda a monitorear el estado y el rendimiento de su infraestructura y aplicaciones en contenedores. El operador de observabilidad de Amazon CloudWatch está diseñado para instalar y configurar los componentes necesarios.

Amazon EKS admite el operador de observabilidad de CloudWatch como complemento de Amazon EKS. El complemento permite Información de contenedores en los nodos de trabajo de Linux y Windows del clúster. Para activar Información de contenedores en Windows, la versión del complemento de Amazon EKS debe ser `1.5.0` o posterior. Actualmente, CloudWatch Application Signals no es compatible con Amazon EKS en Windows.

Los siguientes temas describen cómo comenzar a utilizar el operador de observabilidad de CloudWatch para el clúster de Amazon EKS.

- Para obtener instrucciones sobre la instalación de este complemento, consulte Instalación del agente de CloudWatch con el complemento de EKS de observabilidad de Amazon CloudWatch o el gráfico de Helm en la Guía del usuario de Amazon CloudWatch.
- Para obtener más información sobre CloudWatch Application Signals, consulte Application Signals en la Guía del usuario de Amazon CloudWatch.
- Para obtener más información sobre Container Insights, consulte Using Container Insights en la Guía del usuario de Amazon CloudWatch.

Envío de los registros del plano de control a Registros de CloudWatch

El registro del plano de control de Amazon EKS proporciona registros de auditoría y diagnóstico directamente desde el plano de control de Amazon EKS a CloudWatch Logs en su cuenta. Estos registros hacen que le resulte más fácil asegura y ejecutar los clústeres. Puede seleccionar los tipos de registro exactos que necesita. Los registros se envían como secuencias de registro a un grupo para cada clúster de Amazon EKS en CloudWatch. Puede usar los filtros de suscripción de CloudWatch para analizar los registros en tiempo real o reenviarlos a otros servicios (los registros se codificarán en Base64 y se comprimirán en formato gzip). Para obtener más información, consulte Registros de Amazon CloudWatch.

Para comenzar a utilizar el registro del plano de control de Amazon EKS, elija los tipos de registro que desea habilitar para los clústeres nuevos o existentes de Amazon EKS. Puede habilitar o desactivar cada tipo de registro en función del clúster por medio de la Consola de administración de AWS, la CLI de AWS (versión `1.16.139` o posterior) o la API de Amazon EKS. Cuando se encuentran habilitados, los registros se envían automáticamente desde el clúster de Amazon EKS a CloudWatch Logs en la misma cuenta.

Cuando se utiliza el registro del plano de control de Amazon EKS, se facturan los precios estándar de Amazon EKS para cada clúster que ejecuta. Se cobra la ingesta de datos de CloudWatch Logs y los costos de almacenamiento estándar para cualquier registro enviado a CloudWatch Logs desde sus clústeres. También se cobran los recursos de AWS, como las instancias de Amazon EC2 o los volúmenes de Amazon EBS que aprovisiona como parte de su clúster.

Están disponibles los siguientes tipos de registro de plano de control de clúster. Cada tipo de registro se corresponde con un componente del plano de control de Kubernetes. Para obtener más información acerca de estos componentes, consulte los componentes de Kubernetes en la documentación de Kubernetes.

Servidor de la API**`api`** ()

> El servidor de la API del clúster es el componente del plano de control que expone la API de Kubernetes. Si habilita los registros del servidor de la API al lanzar el clúster o poco después, estos registros incluyen los indicadores del servidor de la API que se usaron para iniciar el servidor de la API. Para obtener más información, consulte kube-apiserver y la política de auditoría en la documentación de Kubernetes.

Auditoría**`audit`** ()

> Los registros de auditoría de Kubernetes ofrecen un registro de los usuarios individuales, administradores o componentes del sistema que han afectado el clúster. Para obtener más información, consulte la sección de auditorías en la documentación de Kubernetes.

Autenticador**`authenticator`** ()

> Los registros del autenticador son exclusivos de Amazon EKS. Estos registros representan el componente del plano de control que Amazon EKS utiliza para la autenticación de control de acceso basado en rol (RBAC) de Kubernetes mediante credenciales de IAM. Para obtener más información, consulte Administración de clústeres.

Administrador de controladores**`controllerManager`** ()

El administrador de controladores administra los bucles de control principal que se envían con Kubernetes. Para obtener más información, consulte kube-controller-manager en la documentación de Kubernetes.

Programador**`scheduler`** ()

El componente programador administra cuándo y dónde ejecutar pods en su clúster. Para obtener más información, consulte kube-scheduler en la documentación de Kubernetes.

Habilitación o deshabilitación de los registros del plano de control

De forma predeterminada, los registros del plano de control del clúster no se envían a los registros de CloudWatch. Debe habilitar cada tipo de registro de manera individual a fin de enviar registros para el clúster. Las tasas de ingesta, almacenamiento de archivos y análisis de datos de CloudWatch Logs se aplican a los registros del plano de control habilitados. Para obtener más información, consulte los precios de CloudWatch.

Para actualizar la configuración de registro del plano de control, Amazon EKS requiere hasta cinco direcciones IP disponibles en cada subred. Al habilitar un tipo de registro, los registros se envían con un nivel de detalle de registro de 2.

Es posible habilitar o deshabilitar los registros del plano de control mediante la Consola de administración de AWS o la AWS CLI.

Consola de administración de AWS

1. Abra la consola de Amazon EKS.
2. Elija el nombre del clúster para mostrar la información del clúster.
3. Seleccione la pestaña Observabilidad.
4. En la sección Registro del plano de control, elija Administrar registro.
5. Para cada tipo de registro individual, elija si el tipo de registro debe estar habilitado o deshabilitado. De forma predeterminada, cada tipo de registro está desactivado.
6. Elija Save changes (Guardar cambios) para terminar.

AWS CLI

1. Consulte su versión de la CLI de AWS con el siguiente comando.

```
aws --version
```

   Si su versión de la CLI de AWS es anterior a `1.16.139`, primero debe actualizar a la última versión. Para obtener instrucciones para configurar la CLI de AWS, consulte Instalación de la interfaz de línea de comandos de AWS en la Guía del usuario de la interfaz de línea de comandos de AWS.

2. Actualice la configuración de exportación del registro de plano de control del clúster con el siguiente comando de la CLI de AWS. Reemplace *`my-cluster`* con el nombre del clúster y especifique los valores de acceso de punto de conexión deseados.

> **Note**
>
> El siguiente comando envía todos los tipos de registros disponibles a CloudWatch Logs.

```
aws eks update-cluster-config \
    --region region-code \
    --name my-cluster \
    --logging '{"clusterLogging":[{"types":
["api","audit","authenticator","controllerManager","scheduler"],"enabled":true}]}'
```

   Un ejemplo de salida sería el siguiente.

```
{
    "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. Monitoree el estado de la actualización de la configuración del registro con el siguiente comando, utilizando el nombre del clúster y el ID de actualización devueltos por el comando anterior. Su actualización se habrá completado cuando el estado mostrado sea `Successful`.

```
aws eks describe-update \
    --region region-code\
    --name my-cluster \
    --update-id 883405c8-65c6-4758-8cee-2a7c1340a6d9
```

Un ejemplo de salida sería el siguiente.

```
{
    "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": []
    }
}
```

Visualización de registros del plano de control del clúster

Una vez que haya habilitado cualquiera de los tipos de registro del plano de control para el clúster de Amazon EKS, puede verlos en la consola de CloudWatch.

Para obtener más información sobre la visualización, el análisis y la administración de registros en CloudWatch, consulte la Guía del usuario de Amazon CloudWatch Logs.

1. Abra la consola de CloudWatch. Este enlace abre la consola y muestra los grupos de registro disponibles actualmente y los filtra con el prefijo `/aws/eks`.
2. Elija el clúster para el que desee ver registros. El formato del nombre del grupo de registros es `/aws/eks/my-cluster/cluster`.
3. Elija la secuencia de registro que desea ver. En la siguiente lista se describe el nombre de secuencia de registro para cada tipo de registro.

   > **Note**
   >
   > A medida que aumentan los datos de la secuencia de registro, se rotan los nombres de la secuencia de registro. Cuando hay mucho flujo de registro para un determinado tipo de registro, puede ver el último flujo de registro si busca el nombre del flujo de registro con la Last Event Time (Hora del último evento) más reciente.

   - Registros de componentes del servidor de API de Kubernetes (**`api`**): `kube-apiserver-1234567890abcdef01234567890abcde`
   - Auditoría (**`audit`**): `kube-apiserver-audit-1234567890abcdef01234567890abcde`
   - Autenticador (**`authenticator`**): `authenticator-1234567890abcdef01234567890abcde`
   - Administrador de controladores (**`controllerManager`**): `kube-controller-manager-1234567890abcdef01234567890abcde`
   - Programador (**`scheduler`**): `kube-scheduler-1234567890abcdef01234567890abcde`
4. Examine los eventos del flujo de registro.

   Por ejemplo, debería ver los indicadores iniciales del servidor de la API del clúster al ver la parte superior de `kube-apiserver-1234567890abcdef01234567890abcde` .

   > **Note**
   >
   > Si no ve los registros del servidor de API al principio de la secuencia del flujo de registro, es probable que el archivo de registro del servidor de API se haya rotado en el servidor antes de habilitar el registro del servidor de API en el servidor. Los archivos de registro que se rotan antes de habilitar el registro del servidor de la API no se pueden exportar a CloudWatch.

Sin embargo, puede crear un nuevo clúster con la misma versión de Kubernetes y habilitar el registro del servidor API cuando cree el clúster. Los clústeres con la misma versión de la plataforma tienen los mismos indicadores habilitados, por lo que sus indicadores deben coincidir con los indicadores del nuevo clúster. Cuando termine de ver los indicadores del nuevo clúster en CloudWatch, puede eliminar el nuevo clúster.

Registrar llamadas a la API como eventos de AWS CloudTrail

Amazon EKS se integra con AWS CloudTrail. CloudTrail es un servicio que proporciona un registro de las acciones realizadas por un usuario, un rol o un servicio de AWS en Amazon EKS. CloudTrail captura todas las llamadas a la API para Amazon EKS como eventos. Esto incluye llamadas realizadas desde la consola de Amazon EKS y las llamadas desde el código a las operaciones de la API de Amazon EKS.

Si crea un registro de seguimiento, puede habilitar la entrega continua de eventos de CloudTrail a un bucket de Amazon S3. Incluye eventos para Amazon EKS. Si no configura un registro de seguimiento, puede ver los eventos más recientes de la consola de CloudTrail en el Historial de eventos. Mediante la información recopilada por CloudTrail, puede determinar ciertos detalles sobre una solicitud. Por ejemplo, puede determinar cuándo se realizó la solicitud a Amazon EKS, la dirección IP desde la que se realizó la solicitud y quién la realizó.

Para obtener más información sobre CloudTrail, consulte la AWSGuía del usuario de CloudTrail.

Temas

- Consulta de referencias útiles para AWS CloudTrail
- Análisis de las entradas de archivos de registro de AWS CloudTrail
- Visualización de métricas para grupos de Amazon EC2 Auto Scaling

Consulta de referencias útiles para AWS CloudTrail

CloudTrail se habilita en su cuenta de AWS cuando la crea. Cuando se produce alguna actividad en Amazon EKS, dicha actividad se registra en un evento de CloudTrail junto con los eventos de los demás servicios de AWS en Event history (Historial de eventos). Puede ver, buscar y descargar los últimos eventos de la cuenta de AWS. Para obtener más información, consulte Ver eventos con el historial de eventos de CloudTrail.

Para mantener un registro continuo de los eventos de la cuenta de AWS, incluidos los eventos de Amazon EKS, cree un registro de seguimiento. Un registro de seguimiento permite a CloudTrail

enviar archivos de registro a un bucket de Amazon S3. De manera predeterminada, cuando se crea un registro de seguimiento en la consola, el registro de seguimiento se aplica a todas las regiones de AWS. El seguimiento registra los eventos de todas las regiones de AWS en la partición de AWS y envía los archivos de registro al bucket de Amazon S3 especificado. También es posible configurar otros servicios de AWS para analizar en profundidad y actuar en función de los datos de eventos recopilados en los registros de CloudTrail. Para obtener más información, consulte los siguientes recursos.

- Introducción a la creación de registros de seguimiento
- Servicios e integraciones compatibles con CloudTrail
- Configuración de notificaciones de Amazon SNS para CloudTrail
- Recibir archivos de registro de CloudTrail de varias regiones y Recibir archivos de registro de CloudTrail de varias cuentas

Todas las acciones de Amazon EKS se registran en CloudTrail y están documentadas en Amazon EKS API Reference (Referencia de la API de Amazon EKS). Por ejemplo, las llamadas a las secciones CreateCluster, ListClusters y DeleteCluster generan entradas en los archivos de registro de CloudTrail.

Cada entrada de registro o de evento contiene información sobre el tipo de identidad de IAM que ha realizado la solicitud y las credenciales que se han utilizado. Si se utilizaron credenciales temporales, la entrada muestra cómo se obtuvieron las credenciales.

Para obtener más información, consulte Elemento userIdentity de CloudTrail.

Análisis de las entradas de archivos de registro de AWS CloudTrail

Un registro de seguimiento es una configuración que permite la entrega de eventos como archivos de registros en un bucket de Amazon S3 que especifique. Los archivos de registro de CloudTrail pueden contener una o varias entradas de registro. Un evento representa una solicitud específica realizada desde una fuente y contiene información sobre la acción solicitada. Incluye información como la fecha y la hora de la acción y los parámetros de solicitud que se utilizaron. Los archivos de registro de CloudTrail no rastrean en orden la pila de las llamadas públicas a la API, por lo que estas no aparecen en ningún orden específico.

En el ejemplo siguiente, se muestra una entrada de registro de CloudTrail que ilustra la acción `CreateCluster`.

```
{
  "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"
}
```

Entradas de registro para roles vinculados al servicio de Amazon EKS

Los roles vinculados al servicio de Amazon EKS realizan llamadas a la API para los recursos de AWS. Aparecen las entradas de registro de CloudTrail con `username: AWSServiceRoleForAmazonEKS` y `username: AWSServiceRoleForAmazonEKSNodegroup` para las llamadas realizadas por los roles vinculados al servicio de Amazon EKS. Para obtener más información acerca de Amazon EKS y los roles vinculados a servicios, consulte the section called “Roles vinculados a servicios”.

En el siguiente ejemplo se incluye una entrada de registro de CloudTrail que muestra una acción `DeleteInstanceProfile` hecha por el rol vinculado al servicio de `AWSServiceRoleForAmazonEKSNodegroup`, que se indica en `sessionContext`.

```
{
    "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"
}
```

Visualización de métricas para grupos de Amazon EC2 Auto Scaling

Los grupos de nodos administrados por Amazon EKS tienen las métricas de grupos de Amazon EC2 Auto Scaling habilitadas de forma predeterminada y sin cargo adicional. El grupo de escalado automático envía datos de muestra a Amazon CloudWatch cada minuto. Estas métricas se pueden afinar con el nombre de los grupos de escalado automático. Con ellas, obtendrá una visibilidad continua del historial de su grupo de escalado automático que hace posible el funcionamiento de sus grupos de nodos administrados, como los cambios en el tamaño del grupo a lo largo del tiempo. Las métricas de grupos de escalado automático están disponibles en la consola de Auto Scaling o Amazon CloudWatch. Para obtener más información, consulte Monitor CloudWatch metrics for your Auto Scaling groups and instances.

Con la recopilación de métricas del grupo de escalado automático, podrá supervisar el escalado de los grupos de nodos administrados. Las métricas de grupo de escalado automático indican el tamaño mínimo, máximo y deseado de un grupo de escalado automático. Puede crear una alarma si el número de nodos de un grupo de nodos está por debajo del tamaño mínimo, lo que indicaría que

el grupo de nodos no funciona bien. El seguimiento del tamaño del grupo de nodos también es útil para ajustar el recuento máximo y así evitar que su plano de datos se quede sin capacidad.

Si prefiere que no se recopilen estas métricas, puede optar por deshabilitarlas todas o algunas de ellas. Por ejemplo, puede hacer esto para evitar interferencias en los paneles de CloudWatch. Para obtener más información, consulte Amazon CloudWatch metrics for Amazon EC2 Auto Scaling.

Envío de datos de métricas y rastreo con el operador ADOT

Amazon EKS admite el uso de Consola de administración de AWS, AWS CLI y la API de Amazon EKS para instalar y administrar el operador AWS Distro for OpenTelemetry (ADOT). Esto posibilita que las aplicaciones se ejecuten en Amazon EKS para enviar datos de métricas y seguimiento a varias opciones de servicio de monitorización, como Amazon CloudWatch, Prometheus y X-Ray.

Para obtener más información, consulte Getting Started with AWS Distro for OpenTelemetry using EKS Add-Ons en la documentación de AWS Distro para OpenTelemetry.

Consulta de los datos agregados sobre los recursos del clúster con el panel de EKS

image::images/eks-dashboard.png[screenshot of account level cluster metrics]

¿Qué es el panel de Amazon EKS?

El panel de Amazon EKS proporciona una visibilidad consolidada de sus clústeres de Kubernetes en varias regiones de AWS y cuentas de AWS. Con este panel, puede:

- Hacer un seguimiento de los clústeres programados para actualizaciones automáticas por fin de soporte en los próximos 90 días.
- Proyectar los costos del plano de control de EKS para los clústeres con soporte ampliado.
- Revisar los clústeres con información que necesite atención antes de actualizarlos.
- Identificar los grupos de nodos administrados que ejecutan versiones específicas de la AMI.
- Supervisar la distribución de los tipos de soporte de los clústeres (estándar en comparación con el soporte ampliado).

El panel de EKS se integra con Información del clúster de EKS para detectar problemas en sus clústeres, como el uso de API de Kubernetes obsoletas. Para obtener más información, consulte the section called “Información sobre clústeres”.

> **Note**
>
> El panel de EKS no es en tiempo real y se actualiza cada 12 horas. Para la supervisión de clústeres en tiempo real, consulte Supervisión de clústeres

¿Cómo utiliza el panel AWS Organizations?

El panel de Amazon EKS requiere la integración de AWS Organizations para funcionar. Aprovecha AWS Organizations para recopilar de forma segura la información del clúster en todas las cuentas. Esta integración proporciona una administración y gobernanza centralizadas a medida que su infraestructura de AWS se amplía.

Si AWS Organizations no está habilitado para su infraestructura, consulte la Guía del usuario de AWS Organizations para obtener instrucciones de configuración.

Acceso entre cuentas y regiones

El panel de EKS puede ver los recursos del clúster en cualquier cuenta que sea miembro de la organización de AWS. Para generar una lista de las cuentas de AWS de su organización, consulte Export details for all accounts in an organization.

La región de AWS us-east-1 genera el panel. Debe iniciar sesión en esta región para ver el panel. El panel agrega datos entre regiones de AWS, pero no incluye `GovCloud` ni las regiones de China.

Términos clave

- AWS Organization: una estructura unificada de administración para varias cuentas de AWS.
- Cuenta de administración: la cuenta principal que controla la AWS Organization.
- Cuenta de miembro: cualquier cuenta de la organización, excepto la cuenta de administración.
- Administrador delegado: una cuenta de miembro a la que se conceden permisos administrativos específicos entre cuentas. Dentro de la cuenta de administración, puede seleccionar una cuenta de administrador delegado por servicio de AWS.
- Acceso de confianza: autorización para que el panel de EKS acceda a la información del clúster en todas las cuentas de la organización.
- Rol vinculado al servicio (SLR): un tipo único de rol de IAM vinculado directamente a un servicio de AWS. El panel de EKS utiliza un SLR para leer la información sobre sus cuentas y organizaciones.

- Para obtener más información, consulte Terminology and concepts en la Guía del usuario de AWS Organizations.

Información general

1. Acceda a la cuenta de administración de su AWS Organization.
   - Los pasos para acceder a la cuenta de administración dependen de cómo haya configurado su organización de AWS. Por ejemplo, puede acceder a la cuenta de administración a través del Centro de identidades de AWS u Okta.
2. Active el acceso de confianza a través de la consola de EKS.
3. Asigne un administrador delegado mediante su ID de cuenta de AWS.
4. Cambie a la cuenta de administrador delegado.
5. Acceda a la consola de EKS mejorada con visibilidad en toda la organización.

Activación del panel de control de EKS mediante la consola de AWS

> **Important**
>
> Debe iniciar sesión en la cuenta de administración de su AWS Organization para activar el panel de EKS.

Acceso a la configuración del panel de EKS

1. Confirme lo siguiente:
   a. Tiene AWS Organizations activadas y configuradas.
   b. Ha iniciado sesión en la cuenta de administración de la organización.
   c. Está viendo la Consola de administración de AWS en la región us-east-1.
2. Vaya a la consola de EKS.
3. En la barra lateral izquierda, abra Configuración del panel.

Configuración del acceso al panel de Amazon EKS

1. Busque el ID de cuenta de AWS de la cuenta de AWS a la que desea permitir ver el panel de EKS.
   a. Este paso es opcional, pero se recomienda hacerlo. Si no lo hace, solo podrá acceder al panel desde la cuenta de administración. Como práctica recomendada, debe limitar el acceso a la cuenta de administración.
2. Haga clic en Activar el acceso confiable.
   a. Ahora puede ver el panel desde la cuenta de administración.
3. Haga clic en Registrar administrador delegado e ingrese el ID de la cuenta de AWS que utilizará para ver el panel.
   a. Ahora puede ver el panel desde la cuenta de administrador delegado o desde la cuenta de administración.

Para obtener información sobre los permisos necesarios para activar el panel, consulte Minimum IAM policies required.

Vista del panel de EKS

1. Inicie sesión en la cuenta de administrador delegado (recomendado) o en la cuenta de administración.
2. Inicie sesión en la región us-east-1.
3. Vaya al servicio de EKS y seleccione Panel en la barra lateral izquierda.

> **Note**
>
> Revise los permisos de IAM necesarios para ver el panel de EKS.

Configuración del panel

Puede configurar la vista del panel y filtrar los recursos.

Recursos disponibles

- Clústeres: vea información agregada sobre el estado y la ubicación de los clústeres de EKS.

  - Clústeres con problemas de estado.
  - Clústeres en el soporte extendido de EKS.
  - Desglose de los clústeres por versión de Kubernetes.
- Grupos de nodos administrados: revise los grupos de nodos administrados y las instancias de EC2.
  - Grupos de nodos por tipo de AMI, como Amazon Linux o Bottlerocket.
  - Problemas de estado de los grupos de nodos.
  - Distribución de tipos de instancias.
- Complementos: obtenga información sobre los complementos de Amazon EKS que ha instalado y su estado.
  - Número de instalaciones por complemento.
  - Complementos con problemas de estado.
  - Distribución de versiones por complemento.

Vistas disponibles

- Vista gráfica
  - Una vista de widget personalizable que muestra gráficos y visualizaciones del recurso seleccionado.
  - Los cambios en la vista gráfica, como la eliminación de un widget, son visibles para todos los usuarios del panel de EKS.
- Vista de recurso
  - Una vista de lista del recurso seleccionado, compatible con filtros.
- Vista de mapa
  - Vea la distribución geográfica del recurso seleccionado.

Filtrado del panel de EKS

Puede filtrar el panel de EKS por:

- AWSCuenta de
- Unidad organizativa, definida por AWS Organizations
- Región de AWS

Desactivación del panel de EKS mediante la consola de AWS

1. Confirme lo siguiente:
   a. Tiene AWS Organizations activadas y configuradas.
   b. Ha iniciado sesión en la cuenta de administración de la organización.
   c. Está viendo la Consola de administración de AWS en la región us-east-1.
2. Vaya a la consola de EKS.
3. En la barra lateral izquierda, abra Configuración del panel.
4. Haga clic en Desactivar el acceso confiable.

Solución de problemas con el panel

Problema al activar el panel de EKS

- Debe iniciar sesión en la cuenta de administración de una AWS Organization.
  - Si no dispone de una AWS Organization, cree una. Aprenda a crear y configurar una organización.
  - Si su cuenta de AWS ya es miembro de una AWS Organization, identifique al administrador de la organización.
- Debe iniciar sesión en la cuenta de AWS con permisos de IAM suficientes para crear y actualizar recursos de AWS Organizations.

Problema al ver el panel de EKS

- Debe iniciar sesión en una de las siguientes cuentas de AWS:
  - La cuenta de administración de la AWS Organization
  - Una cuenta de administrador delegado, identificada en la configuración del panel de EKS de la cuenta de administración.
- Si ha activado recientemente el panel de EKS, tenga en cuenta que el rellenado inicial de datos puede tardar hasta 12 horas.
- Intente volver a activar el panel mediante la CLI, incluida la creación del rol vinculado al servicio.

Movimiento inesperado de los widgets del panel

- El panel de EKS guarda la vista del widget configurable por cuenta de AWS. Si cambia la vista del widget, otras personas que usen la misma cuenta de AWS verán los cambios.

Configuración de la integración del panel de EKS con AWS Organizations

En esta sección se proporcionan instrucciones paso a paso para configurar la integración del panel de EKS con AWS Organizations. Aprenderá a activar y desactivar el acceso de confianza entre los servicios, así como a registrar y anular el registro de las cuentas de administrador delegado. Cada tarea de configuración se puede llevar a cabo mediante la consola de AWS o la AWS CLI.

Habilitar el acceso de confianza

El acceso de confianza autoriza al panel de EKS a acceder de forma segura a la información del clúster en todas las cuentas de su organización.

Uso de la consola de AWS

1. Inicie sesión en la cuenta de administración de la AWS Organization.
2. Vaya a la consola de EKS en la región us-east-1.
3. En la barra lateral izquierda, seleccione Configuración del panel.
4. Haga clic en Activar el acceso confiable.

> **Note**
>
> Al habilitar el acceso de confianza a través de la consola de EKS, el sistema crea automáticamente el rol vinculado al servicio de `AWSServiceRoleForAmazonEKSDashboard`. Esta creación automática no se produce si habilita el acceso de confianza mediante la AWS CLI o la consola de AWS Organizations.

Uso de la CLI de AWS

1. Inicie sesión en la cuenta de administración de la AWS Organization.
2. Ejecute los comandos siguientes:

```
aws iam create-service-linked-role --aws-service-name dashboard.eks.amazonaws.com
aws organizations enable-aws-service-access --service-principal eks.amazonaws.com
```

Deshabilitar el acceso de confianza

Al deshabilitar el acceso de confianza, se revoca el permiso del panel de EKS para acceder a la información del clúster en todas las cuentas de su organización.

Uso de la consola de AWS

1. Inicie sesión en la cuenta de administración de la AWS Organization.
2. Vaya a la consola de EKS en la región us-east-1.
3. En la barra lateral izquierda, seleccione Configuración del panel.
4. Haga clic en Desactivar el acceso confiable.

Uso de la CLI de AWS

1. Inicie sesión en la cuenta de administración de la AWS Organization.
2. Use el siguiente comando:

```
aws organizations disable-aws-service-access --service-principal eks.amazonaws.com
```

Habilitar una cuenta de administrador delegado

Un administrador delegado es una cuenta de miembro a la que se concede permiso para acceder al panel de EKS.

Uso de la consola de AWS

1. Inicie sesión en la cuenta de administración de la AWS Organization.
2. Vaya a la consola de EKS en la región us-east-1.
3. En la barra lateral izquierda, seleccione Configuración del panel.
4. Haga clic en Registrar administrador delegado.
5. Ingrese el ID de la cuenta de AWS que quiera elegir como administrador delegado.

6. Confirme el registro.

Uso de la CLI de AWS

1. Inicie sesión en la cuenta de administración de la AWS Organization.
2. Ejecute el siguiente comando, pero reemplace `123456789012` por su ID de cuenta:

```
aws organizations register-delegated-administrator --account-id 123456789012 --
service-principal eks.amazonaws.com
```

Desactivación de una cuenta de administrador delegado

Al desactivar un administrador delegado, se elimina el permiso de acceso de la cuenta al panel de EKS.

Uso de la consola de AWS

1. Inicie sesión en la cuenta de administración de la AWS Organization.
2. Vaya a la consola de EKS en la región us-east-1.
3. En la barra lateral izquierda, seleccione Configuración del panel.
4. Localice el administrador delegado en la lista.
5. Haga clic en Anular registro junto a la cuenta que quiera eliminar como administrador delegado.

Uso de la CLI de AWS

1. Inicie sesión en la cuenta de administración de la AWS Organization.
2. Ejecute el siguiente comando, pero reemplace `123456789012` por el ID de cuenta del administrador delegado:

```
aws organizations deregister-delegated-administrator --account-id 123456789012 --
service-principal eks.amazonaws.com
```

Políticas de IAM mínimas requeridas

En esta sección se describen las políticas de IAM mínimas necesarias para activar el acceso de confianza y delegar un administrador para la integración del panel de EKS con AWS Organizations.

Política para activar el acceso de confianza

Para activar el acceso de confianza entre el panel de EKS y AWS Organizations, necesita los siguientes permisos:

```
{
    "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"
        }
    ]
}
```

Política para delegar un administrador

Para registrar o anular el registro de un administrador delegado para el panel de EKS, necesita los siguientes permisos:

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "organizations:RegisterDelegatedAdministrator",
                "organizations:DeregisterDelegatedAdministrator",
                "organizations:ListDelegatedAdministrators"
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

Política para ver el panel de EKS

```
{
    "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**
>
> Estas políticas deben adjuntarse a la entidad principal de IAM (usuario o rol) en la cuenta de administración de su AWS Organization. Las cuentas de miembro no pueden activar el acceso de confianza ni delegar administradores.

Mejora de EKS con servicios de AWS integrados

Además de los servicios mencionados en otras secciones, Amazon EKS trabaja con más servicios de AWS para ofrecer soluciones adicionales. En este tema, se identifican algunos de los demás servicios que utiliza Amazon EKS para agregar funcionalidad o los servicios que Amazon EKS utiliza para realizar tareas.

Temas

- Copia de seguridad de Clústeres de EKS con AWS Backup
- Creación de recursos de Amazon EKS con AWS CloudFormation
- Conexión a los repositorios de Git con AWS CodeConnections
- Análisis de eventos de seguridad en EKS con Amazon Detective
- Detección de amenazas con Amazon GuardDuty
- Lanzamiento de clústeres EKS de baja latencia con zonas locales de AWS
- Evaluación de la resiliencia del clúster EKS con AWS Resilience Hub
- Administración de secretos de aplicaciones con AWS Secrets Manager
- Centralización y análisis de datos de seguridad de EKS con Security Lake
- Habilitación de conectividad segura entre clústeres con Amazon VPC Lattice

Copia de seguridad de Clústeres de EKS con AWS Backup

AWS Backup admite copias de seguridad de los clústeres de Amazon EKS, lo que incluye el estado del clúster de Kubernetes y el almacenamiento persistente adjunto al clúster de EKS mediante una reclamación de volúmenes persistentes (volúmenes de EBS, sistemas de archivos de EFS y buckets de S3). Una copia de seguridad de Amazon EKS creará un punto de recuperación compuesto, en el que habrá un punto de recuperación secundario para cada recurso del que se haga una copia de seguridad.

- Cómo hacer copias de seguridad de recursos: Introducción a AWS Backup
- Creación de copias de seguridad por tipo de recurso: Copias de seguridad de Amazon EKS
- Cómo restaurar los clústeres de Amazon EKS: Restauración de copias de seguridad de Amazon EKS

Para comenzar a través de la consola de EKS, la consola de AWS Backup o la CLI, asegúrese de que su rol de IAM tenga los siguientes permisos:

- Puede encontrarlos en la política administrada de AWS Backup AWSBackupServiceRolePolicyForBackup. Contiene los permisos necesarios para hacer copias de seguridad del clúster de Amazon EKS y del almacenamiento persistente de EBS y EFS.
- Si su clúster de EKS contiene un bucket de S3, tendrá que asegurarse de que se agreguen y activen las políticas y los requisitos previos que se indican a continuación para su bucket de S3, tal como se muestra en la documentación:
- AWSBackupServiceRolePolicyForS3Backup
- Requisitos previos para las copias de seguridad de S3

Asegúrese de que sus clústeres de EKS tengan la siguiente configuración:

- Modo de autorización del clúster de EKS establecido en API o API_AND_CONFIG_MAP para AWS Backup a fin de crear Entradas de acceso para acceder al clúster de EKS.

Creación de recursos de Amazon EKS con AWS CloudFormation

Amazon EKS está integrado con AWS CloudFormation, un servicio que lo ayuda a modelar y configurar los recursos de AWS para que pueda dedicar menos tiempo a crear y administrar sus recursos e infraestructura. Puede crear una plantilla que describa todos los recursos de AWS que desea, por ejemplo un clúster de Amazon EKS, y AWS CloudFormation se encarga de aprovisionar y configurar esos recursos.

Cuando utiliza AWS CloudFormation, puede volver a utilizar la plantilla para configurar los recursos de Amazon EKS de forma coherente y repetida. Solo tiene que describir los recursos una vez y, luego, aprovisionar los mismos recursos una y otra vez en varias cuentas y regiones de AWS.

Plantillas de Amazon EKS y AWS CloudFormation

Para aprovisionar y configurar los recursos de Amazon EKS y los servicios relacionados, debe comprender las plantillas de AWS CloudFormation. Las plantillas son archivos de texto con formato JSON o YAML. Estas plantillas describen los recursos que desea proporcionar en sus pilas de AWS CloudFormation. Si no está familiarizado con JSON o YAML, puede utilizar AWS CloudFormation Designer para ayudarle a comenzar a utilizar las plantillas de AWS CloudFormation. Para obtener

más información, consulte ¿Qué es AWS CloudFormation Designer? en la Guía del usuario de AWS CloudFormation.

Amazon EKS admite la creación de clústeres y grupos de nodos en AWS CloudFormation. Para obtener más información, incluidos ejemplos de plantillas JSON y YAML para los recursos de Amazon EKS, consulte la Referencia del tipo de recurso de Amazon EKS en la Guía del usuario de AWS CloudFormation.

Obtenga más información sobre AWS CloudFormation

Para obtener más información acerca de AWS CloudFormation, consulte los siguientes recursos:

- AWS CloudFormation
- Guía del usuario de AWS CloudFormation
- Guía del usuario de la interfaz de la línea de comandos de AWS CloudFormation

Conexión a los repositorios de Git con AWS CodeConnections

AWS CodeConnections proporciona una forma segura de conectar servicios de AWS a repositorios de código fuente de terceros. AWS CodeConnections es compatible con GitHub, GitLab, Bitbucket y otros proveedores. Para obtener más información y comenzar, consulte Cómo trabajar con conexiones.

Uso de AWS CodeConnections con Argo CD

Al usar la capacidad de EKS para Argo CD, puede elegir si desea usar AWS CodeConnections para activar la autenticación segura en los repositorios de Git sin administrar credenciales de larga duración ni tokens de acceso personal. AWS CodeConnections gestiona el flujo de autenticación de OAuth y administra la conexión con su proveedor de Git, lo que proporciona un enfoque seguro y administrable para acceder a sus repositorios de GitOps y manifiestos de aplicaciones almacenados en proveedores de Git de terceros.

Requisitos previos

- Un clúster de Amazon EKS con la capacidad de Argo CD creada
- Una conexión creada en AWS CodeConnections con el proveedor de Git
- Permisos de IAM configurados para que Argo CD utilice la conexión

Cómo configurar CodeConnections para el acceso al repositorio de Argo CD

1. Cree una conexión en la consola de CodeConnections:
   a. Abra la consola de CodeConnections.
   b. Elija Crear conexión.
   c. Seleccione su proveedor (GitHub, GitLab o Bitbucket) y siga el flujo de autenticación.
   d. Anote el ARN de la conexión para usarlo en la configuración de Argo CD.
2. Asegúrese de que el rol de capacidad de Argo CD tenga permisos para usar la conexión con una política basada en recursos:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codeconnections:UseConnection",
        "codeconnections:GetConnection"
      ],
      "Resource": "arn:aws:codeconnections:region:account-id:connection/connection-
id"
    }
  ]
}
```

3. Configure Argo CD para que haga referencia al recurso de CodeConnections cuando se agregue un repositorio con el punto de conexión del recurso de CodeConnections como URL del repositorio. La capacidad de Argo CD utiliza la conexión para autenticarse en su proveedor de Git sin requerir credenciales de larga duración.

Consideraciones sobre el uso de CodeConnections con Argo CD

Cuando utilice AWS CodeConnections con la capacidad de EKS para Argo CD, tenga en cuenta lo siguiente:

- La conexión de CodeConnections debe estar en la misma región de AWS que la del clúster de EKS.

- El rol de capacidad de Argo CD debe tener los permisos `codeconnections:UseConnection` y `codeconnections:GetConnection`.
- CodeConnections administra automáticamente el flujo de OAuth y el ciclo de vida de las credenciales.

Para obtener más información sobre cómo configurar el acceso al repositorio con Argo CD, consulte the section called “Configuración de repositorios”.

Análisis de eventos de seguridad en EKS con Amazon Detective

Amazon Detective ayuda a analizar, investigar e identificar rápidamente la causa raíz de resultados de seguridad o actividades sospechosas. Detective recopila automáticamente los datos de registro de sus recursos de AWS. A continuación, utiliza el machine learning, el análisis estadístico y la teoría de grafos para generar visualizaciones que lo ayuden a realizar investigaciones sobre la seguridad con mayor rapidez y de forma más eficaz. Las agregaciones de datos, los resúmenes y los contextos prediseñados de Detective ayudan a analizar y determinar rápidamente la naturaleza y el alcance de los posibles problemas de seguridad. Para obtener más información, consulte la Guía del usuario de Amazon Detective.

Detective organiza los datos de Kubernetes y AWS en resultados tales como:

- Detalles del clúster de Amazon EKS, incluyendo la identidad de IAM que creó el clúster y el rol de servicio del clúster. Puede investigar la actividad de la API de Kubernetes y AWS de estas identidades de IAM con Detective.
- Detalles del contenedor, como la imagen y el contexto de seguridad. También puede revisar los detalles de los pods finalizados.
- Actividad de la API de Kubernetes, lo que incluye tendencias generales de actividad de la API y detalles sobre llamadas a la API específicas. Por ejemplo, puede mostrar el número de llamadas a la API de Kubernetes procesadas correctamente y fallidas que se han emitido durante un intervalo de tiempo seleccionado. Además, la sección sobre llamadas a la API observadas recientemente puede resultar útil para identificar actividades sospechosas.

Los registros de auditoría de Amazon EKS son un paquete de orígenes de datos opcionales que se puede agregar a su gráfico de comportamiento de Detective. Puede ver los paquetes de orígenes opcionales disponibles y su estado en su cuenta. Para obtener más información, consulte Amazon EKS audit logs for Detective en la Guía del usuario de Amazon Detective.

Uso de Amazon Detective con Amazon EKS

Para poder revisar los resultados, Detective debe estar habilitado durante al menos 48 horas en la misma región de AWS donde se encuentre el clúster. Para obtener más información, consulte Setting up Amazon Detective en la Guía del usuario de Amazon Detective.

1. Abra la consola de Detective en https://console.aws.amazon.com/detective/.
2. En el panel de navegación izquierdo, seleccione Buscar.
3. Seleccione Elegir tipo y, a continuación, Clúster de EKS.
4. Escriba el nombre o ARN del clúster y, a continuación, seleccione Buscar.
5. En los resultados de la búsqueda, seleccione el nombre del clúster cuya actividad desea ver. Para obtener más información sobre lo que puede ver, consulte Actividad total de la Api de Kubernetes relacionada con un clúster de Amazon EKS en la Guía del usuario de Amazon Detective.

Detección de amenazas con Amazon GuardDuty

Amazon GuardDuty es un servicio de detección de amenazas que ayuda a proteger las cuentas, los contenedores, las cargas de trabajo y los datos de su entorno de AWS. Mediante modelos de machine learning (ML) y capacidades de detección de anomalías y amenazas, GuardDuty supervisa continuamente los diferentes orígenes de registro y la actividad en tiempo de ejecución para identificar y priorizar los posibles riesgos de seguridad y actividades maliciosas en su entorno.

Entre otras características, GuardDuty ofrece las siguientes dos características que detectan posibles amenazas para sus clústeres de EKS: Protección de EKS y Supervisión en tiempo de ejecución.

> **Note**
>
> Nuevo: el modo automático de Amazon EKS se integra con GuardDuty.

Protección de EKS

Esta característica proporciona una cobertura de detección de amenazas para ayudarlo a proteger los clústeres de Amazon EKS mediante la supervisión de los registros de auditoría de Kubernetes asociados. Los registros de auditoría de Kubernetes capturan las acciones secuenciales del clúster, incluidas las actividades de los usuarios, las aplicaciones que utilizan la

API de Kubernetes y el plano de control. Por ejemplo, GuardDuty puede identificar que un usuario no autenticado invocó las API llamadas para manipular los recursos de un clúster de Kubernetes.

Al activar la Protección de EKS, GuardDuty solo podrá acceder a los registros de auditoría de Amazon EKS para detectar amenazas de forma continua. Si GuardDuty identifica una amenaza potencial para el clúster, genera un resultado del registro de auditoría de Kubernetes asociado de un tipo específico. Para obtener más información sobre los tipos de resultados disponibles en los registros de auditoría de Kubernetes, consulte Kubernetes audit logs finding types en la Guía del usuario de Amazon GuardDuty.

Para obtener más información, consulte Protección de EKS en la Guía del usuario de Amazon GuardDuty.

Supervisión en tiempo de ejecución

Esta característica supervisa y analiza los eventos de archivos, redes y sistemas operativos para ayudarlo a detectar posibles amenazas en cargas de trabajo específicas de AWS en su entorno.

Cuando habilita la Supervisión del tiempo de ejecución e instala el agente GuardDuty en sus clústeres de Amazon EKS, GuardDuty comienza a supervisar los eventos de tiempo de ejecución asociados a este clúster. Tenga en cuenta que el agente de GuardDuty y la Supervisión en tiempo de ejecución no están disponibles para los Nodos híbridos de Amazon EKS, por lo que la Supervisión en tiempo de ejecución no se encuentra disponible para los eventos en tiempo de ejecución que se produzcan en los nodos híbridos. Si GuardDuty identifica una amenaza potencial para el clúster, genera un resultado de la supervisión del tiempo de ejecución asociado. Por ejemplo, una amenaza puede empezar por comprometer un único contenedor que ejecuta una aplicación web vulnerable. Es posible que esta aplicación web tenga permisos de acceso a los contenedores y las cargas de trabajo subyacentes. En este escenario, las credenciales mal configuradas podrían dar lugar a un acceso más amplio a la cuenta y a los datos almacenados en ella.

Para configurar Runtime Monitoring, instale el agente de GuardDuty en el clúster como complemento de Amazon EKS. Para obtener más información del complemento, consulte the section called “ Complementos de AWS”.

Para obtener más información, consulte Runtime Monitoring en la Guía del usuario de Amazon GuardDuty.

Lanzamiento de clústeres EKS de baja latencia con zonas locales de AWS

Una zona local de AWS es una extensión de una región de AWS cercana geográficamente a los usuarios. Las zonas locales tienen sus propias conexiones a Internet y admiten AWS Direct Connect. Los recursos creados en una zona local pueden prestar servicio a los usuarios locales con comunicaciones de baja latencia. Para obtener más información, consulte la Guía del usuario de las zonas locales de AWS y zonas locales en la Guía del usuario de Amazon EC2.

Amazon EKS admite ciertos recursos en zonas locales. Esto incluye grupos de nodos administrados, nodos de Amazon EC2 autoadministrados, volúmenes de Amazon EBS y equilibradores de carga de aplicaciones (ALB). Recomendamos que tenga en cuenta lo siguiente al utilizar zonas locales como parte del clúster de Amazon EKS.

- No se pueden crear nodos Fargate en zonas locales con Amazon EKS.
- El plano de control de Kubernetes administrado por Amazon EKS siempre se ejecuta en la región de AWS. El plano de control de Kubernetes administrado por Amazon EKS no se puede ejecutar en la zona local. Dado que las Local Zones aparecen como una subred dentro de la VPC, Kubernetes ve los recursos de la zona local como parte de esa subred.
- El clúster de Kubernetes de Amazon EKS se comunica con las instancias de Amazon EC2 que ejecuta en la región o zona local de AWS mediante interfaces de redes elásticas administradas por Amazon EKS. Para obtener más información sobre la arquitectura de redes de Amazon EKS, consulte Configurar redes.
- A diferencia de las subredes regionales, Amazon EKS no puede colocar interfaces de redes en las subredes de zona local. Esto significa que no debe especificar las subredes de zona local al crear el clúster. Sin embargo, puede tener nodos de trabajo en diferentes zonas locales conectadas al mismo clúster.

Evaluación de la resiliencia del clúster EKS con AWS Resilience Hub

AWS Resilience Hub evalúa la resiliencia de un clúster de Amazon EKS mediante el análisis de su infraestructura. AWS Resilience Hub utiliza la configuración de control de acceso basado en roles (RBAC) de Kubernetes para evaluar las cargas de trabajo de Kubernetes implementadas en el

clúster. Para obtener más información, consulte Habilitación del acceso de AWS Resilience Hub a un clúster de Amazon EKS en la Guía del usuario de AWS Resilience Hub.

Administración de secretos de aplicaciones con AWS Secrets Manager

AWS Secrets Manager lo ayuda a administrar credenciales, claves de API y otros secretos durante su ciclo de vida, acceder a ellos y rotarlos. Con Secrets Manager, puede proteger y administrar los secretos utilizados para acceder a los recursos en AWS, en servicios de terceros y en las instalaciones. Para obtener más información, consulte la Guía del usuario de AWS Secrets Manager.

Al utilizar la capacidad de EKS para Argo CD, Secrets Manager proporciona una forma segura de almacenar y recuperar las credenciales de repositorios de Git sin necesidad de codificar información confidencial en la configuración y los recursos de Argo CD. Esta integración es particularmente útil para administrar los tokens de acceso a repositorios privados y las claves de SSH que utiliza Argo CD para sincronizar aplicaciones de los repositorios de Git.

Uso de AWS Secrets Manager con Argo CD

Al utilizar la capacidad de EKS para Argo CD, puede almacenar las credenciales de repositorios de Git en Secrets Manager y configurar Argo CD para recuperarlas. Este enfoque es más seguro que almacenar las credenciales directamente en la configuración de Argo CD o utilizar tokens de acceso personal de larga duración.

Requisitos previos

- Un clúster de Amazon EKS con la capacidad de Argo CD activada
- Credenciales de repositorios de Git almacenadas en AWS Secrets Manager
- Permisos de IAM configurados para que Argo CD acceda a Secrets Manager

Configuración de Argo CD para que utilice Secrets Manager como credenciales de repositorios

1. Almacene credenciales de Git en Secrets Manager. Por ejemplo, haga lo siguiente para almacenar un token de acceso personal de GitHub:

```
aws secretsmanager create-secret \
  --name argocd/github-token \
  --secret-string '{"username":"git","password":"ghp_xxxxxxxxxxxx"}'
```

2. Asegúrese de que el rol de capacidad de Argo CD tenga permisos para recuperar el secreto:

```
{
  "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**
>
> El permiso de descifrado de KMS es obligatorio porque Secrets Manager cifra todos los secretos mediante AWS KMS. La condición restringe el descifrado solo a los secretos con el prefijo `argocd/`. Si utiliza la clave administrada de AWS predeterminada para Secrets Manager, este permiso es suficiente. En el caso de las claves de KMS administradas por el cliente, actualice el campo `Resource` con el ARN de clave específico.

3. Configure Argo CD para que utilice las credenciales de Secrets Manager. Para obtener información sobre cómo sincronizar los secretos de Secrets Manager con los secretos de

Kubernetes a los que Argo CD puede hacer referencia, consulte Secret Management en la documentación de Argo CD.

4. Cree una configuración de repositorios de Argo CD que haga referencia al ARN de secreto:

```
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
```

Para obtener más información sobre cómo configurar el acceso al repositorio con Argo CD, consulte the section called “Configuración de repositorios”.

Centralización y análisis de datos de seguridad de EKS con Security Lake

Amazon Security Lake es un servicio de lago de datos de seguridad totalmente administrado que le permite centralizar los datos de seguridad de varios orígenes, incluido Amazon EKS. Al integrar Amazon EKS con Security Lake, puede obtener información más detallada sobre las actividades que se llevan a cabo en sus recursos de Kubernetes y mejorar la seguridad de sus clústeres de Amazon EKS.

> **Note**
>
> Para obtener más información sobre el uso de Security Lake con Amazon EKS y la configuración de los orígenes de datos, consulte la documentación de Amazon Security Lake.

Ventajas de usar Security Lake con Amazon EKS

Datos de seguridad centralizados: Security Lake recopila y centraliza automáticamente los datos de seguridad de sus clústeres de Amazon EKS, junto con datos de otros servicios de AWS, proveedores

de SaaS, orígenes en las instalaciones y orígenes de terceros. Esto proporciona una visión completa de su postura de seguridad en toda su organización.

Formato de datos estandarizado: Security Lake convierte los datos recopilados al formato Open Cybersecurity Schema Framework (OCSF), que es un esquema estándar de código abierto. Esta normalización facilita el análisis y la integración con otras herramientas y servicios de seguridad.

Detección de amenazas mejorada: al analizar los datos de seguridad centralizados, incluidos los registros del plano de control de Amazon EKS, puede detectar actividades potencialmente sospechosas en sus clústeres de Amazon EKS de manera más eficaz. Esto ayuda a identificar y responder rápidamente a los incidentes de seguridad.

Administración de datos simplificada: Security Lake administra el ciclo de vida de sus datos de seguridad con configuraciones de retención y replicación personalizables. Esto simplifica las tareas de administración de datos y garantiza que retenga los datos necesarios para fines de cumplimiento y auditoría.

Habilitación de Security Lake en Amazon EKS

1. Active el registro del plano de control de Amazon EKS en sus clústeres de EKS. Consulte Habilitar y deshabilitar registros de plano de control para obtener instrucciones detalladas.
2. Agregue los registros de auditoría de Amazon EKS como origen en Security Lake. A continuación, Security Lake comenzará a recopilar información detallada sobre las actividades hechas en los recursos de Kubernetes que se ejecutan en sus clústeres de EKS.
3. Configure los ajustes de retención y replicación de sus datos de seguridad en Security Lake en función de sus requisitos.
4. Utilice los datos OCSF normalizados almacenados en Security Lake para responder a incidentes, llevar a cabo análisis de seguridad e integrarlos con otros servicios de AWS o herramientas de terceros. Por ejemplo, puede generar información sobre seguridad a partir de los datos de Amazon Security Lake mediante Amazon OpenSearch Ingestion.

Análisis de los registros de EKS en Security Lake

Security Lake normaliza los eventos de registro de EKS al formato OCSF, lo que facilita el análisis y la correlación de los datos con otros eventos de seguridad. Puede utilizar diversas herramientas y servicios, como Amazon Athena, Amazon QuickSight o herramientas de análisis de seguridad de terceros, para consultar y visualizar los datos normalizados.

Para obtener más información sobre la asignación de OCSF para los eventos de registro de EKS, consulte https://github.com/ocsf/examples/tree/main/mappings/markdown/ AWS /v1.1.0/EKS Audit Logs [referencia de asignación] en el repositorio de GitHub de OCSF.

Habilitación de conectividad segura entre clústeres con Amazon VPC Lattice

Amazon VPC Lattice es un servicio de redes de aplicaciones administrado totalmente e integrado directamente en la infraestructura de red de AWS que puede utilizar para conectar, proteger y supervisar sus servicios en varias cuentas y nubes privadas virtuales (VPC). Con Amazon EKS, puede aprovechar Amazon VPC Lattice mediante el uso del controlador de la API de AWS de puerta de enlace, una implementación de la API de puerta de enlace de Kubernetes. Con Amazon VPC Lattice, puede configurar la conectividad entre clústeres con semántica de Kubernetes estándar de forma sencilla y coherente. Para empezar a utilizar Amazon VPC Lattice con Amazon EKS, consulte la Guía del usuario del controlador de la API de puerta de enlace de AWS.

Conexión de un clúster de Kubernetes a una consola de administración de Amazon EKS con el conector de Amazon EKS

Puede utilizar Amazon EKS Connector a fin de registrar y conectar cualquier clúster de Kubernetes conforme a AWS y visualizarlo en la consola de Amazon EKS. Una vez conectado, puede ver el estado, la configuración y las cargas de trabajo del clúster en la consola de Amazon EKS. Puede utilizar esta característica para ver los clústeres conectados en la consola de Amazon EKS, pero no puede administrarlos. Amazon EKS Connector requiere un agente que sea un proyecto de código abierto en GitHub. Para obtener contenido técnico adicional, incluidas las preguntas frecuentes y las soluciones de problemas, consulte the section called “Solución de problemas del conector de EKS”.

Amazon EKS Connector puede conectar los siguientes tipos de clústeres de Kubernetes a Amazon EKS.

- Clústeres de Kubernetes en las instalaciones
- Clústeres autoadministrados que se ejecutan en Amazon EC2
- Clústeres administrados de otros proveedores de nube

Consideraciones sobre Amazon EKS Connector

Antes de utilizar Amazon EKS Connector, debe comprender lo siguiente:

- Debe tener privilegios administrativos en el clúster de Kubernetes para conectar el clúster a Amazon EKS.
- El clúster de Kubernetes debe tener nodos de trabajo Linux de 64 bits (x86) presentes antes de conectarse. No se admiten los nodos de trabajo de ARM}.
- Debe tener nodos de trabajo en el clúster de Kubernetes que tengan acceso saliente a los puntos de enlace `ssm.` y `ssmmessages.` de Systems Manager. Para obtener más información, consulte Puntos de enlace de Systems Manager en la Referencia general de AWS.
- De forma predeterminada, puede conectar hasta 10 clústeres en una región. Puede solicitar un aumento a través de la consola de cuotas de servicio. Para obtener más información, consulte Solicitud de un aumento de cuota.

- Solo se admiten las API `RegisterCluster`, `ListClusters`, `DescribeCluster` y `DeregisterCluster` de Amazon EKS para clústeres de Kubernetes externos.
- Debe tener los siguientes permisos para registrar un clúster:
  - eks:RegisterCluster
  - ssm:CreateActivation
  - ssm:DeleteActivation
  - iam:PassRole
- Debe tener los siguientes permisos para anular el registro de un clúster:
  - eks:DeregisterCluster
  - ssm:DeleteActivation
  - ssm:DeregisterManagedInstance

Roles de IAM necesarios para Amazon EKS Connector

El uso de Amazon EKS Connector requiere los dos roles de IAM siguientes:

- El rol vinculado al servicio de Amazon EKS Connector se crea al registrar un clúster por primera vez.
- Debe crear el rol de IAM de agente de Amazon EKS Connector. Para obtener más información, consulte the section called “Rol de IAM conector”.

A fin de habilitar el permiso de visualización de clúster y carga de trabajo para entidades principales de IAM, aplique el `eks-connector` y los roles de clúster de Amazon EKS Connector en el clúster. Siga los pasos en Concesión de acceso para ver los recursos del clúster de Kubernetes en una consola de Amazon EKS.

Conexión de un clúster externo de Kubernetes a la consola de administración de Amazon EKS

Puede conectar un clúster externo de Kubernetes en Amazon EKS mediante varios métodos en el siguiente proceso. Este proceso consta de dos pasos: registrar el clúster en Amazon EKS e instalar el agente `eks-connector` del clúster.

> **Important**
>
> Debe completar el segundo paso en un plazo de 3 días a partir de haber completado el primer paso, antes de que caduque el registro.

Consideraciones

Puede usar los manifiestos YAML al instalar el agente. Como alternativa, puede utilizar Helm si registra el clúster con la Consola de administración de AWS o la Interfaz de la línea de comandos de AWS. Sin embargo, no puede usar Helm para instalar el agente si registra el clúster con `eksctl`.

Requisitos previos

- Asegúrese de que se haya creado el rol de agente de Amazon EKS Connector. Siga los pasos que se indican en Creación del rol de agente conector de Amazon EKS.
- Debe tener los siguientes permisos para registrar un clúster:
  - `eks:RegisterCluster`
  - `ssm:CreateActivation`
  - `ssm:DeleteActivation`
  - `iam:PassRole`

Paso 1: registro del clúster

Para registrar un clúster en el conector de Amazon EKS, puede utilizar una de las siguientes herramientas:

- the section called “AWS CLI”
- the section called “Consola de administración de AWS”
- the section called “`eksctl`”

AWS CLI

1. La CLI de AWS debe estar instalada. Para instalarla o actualizarla, consulte Instalación de la CLI de AWS.

2. Para la configuración de Connector, especifique su rol de IAM de agente de Amazon EKS Connector. Para obtener más información, consulte the section called “Roles de IAM necesarios para 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
```

Un ejemplo de salida sería el siguiente.

```
{
    "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"
    }
}
```

Utilizará los valores de `aws-region`, `activationId` y `activationCode` en un paso posterior.

Consola de administración de AWS

1. Abra la consola de Amazon EKS.
2. Elija Add cluster (Agregar clúster) y seleccione Register (Registrar) para abrir la página de configuración.
3. En la sección Configure cluster (Configurar clúster), rellene los siguientes campos:
   - Nombre: un nombre único para el clúster.
   - Provider (Proveedor): elija esta opción para mostrar la lista desplegable de proveedores de clústeres de Kubernetes. Si no conoce al proveedor específico, seleccione Otro.

- EKS Connector role (Rol de EKS Connector): seleccione el rol que se va a utilizar para conectar el clúster.

4. Seleccione Register cluster (Registrar el clúster).
5. Se abre la página de información general del clúster. Si quiere usar el gráfico de Helm, copie el comando `helm install` y continúe con el siguiente paso. Si desea usar el manifiesto de YAML, elija Descargar archivo YAML para descargar el archivo de manifiesto en la unidad local.

> **Important**
>
> Esta es su única oportunidad para copiar el comando `helm install` o descargar este archivo. No navegue fuera de esta página, ya que no se podrá acceder al enlace y deberá anular el registro del clúster e iniciar los pasos desde el principio.

El comando o el archivo de manifiesto solo se puede utilizar una vez para el clúster registrado. Si elimina recursos del clúster de Kubernetes, debe volver a registrar el clúster y obtener un nuevo archivo de manifiesto.

Continúe en el paso siguiente para aplicar el archivo de manifiesto al clúster de Kubernetes.

`eksctl`

1. Se debe instalar la versión `0.68` de `eksctl` o posterior. Para instalarla o actualizarla, consulte the section called “Creación de un clúster (eksctl)”.
2. Registre el clúster al proporcionar un nombre, un proveedor y una región.

```
eksctl register cluster --name my-cluster --provider my-provider --region region-code
```

Ejemplo de salida:

```
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
```

De este modo, se crean archivos en el equipo local. Estos archivos deben aplicarse al clúster externo en un plazo de 3 días o el registro vence.

3. En una terminal que pueda acceder al clúster, aplique el archivo `eks-connector-binding.yaml`:

```
kubectl apply -f eks-connector-binding.yaml
```

Paso 2: Instalación del agente de **`eks-connector`**

Para instalar el agente `eks-connector`, utilice una de las siguientes herramientas:

- the section called “Helm”
- the section called “yaml”

Helm

> **Note**
>
> Si ha registrado el clúster con `eksctl`, utilice el método de manifiesto YAML en lugar del método de gráfico de Helm.

1. Si utilizó la CLI de AWS en el paso anterior, reemplace el `ACTIVATION_CODE` y `ACTIVATION_ID` en el siguiente comando por los valores de `activationId` y `activationCode`, respectivamente. Reemplace la `aws-region` por la región de AWS que utilizó en el paso anterior. Ejecute el siguiente comando para instalar el agente de `eks-connector` en el clúster del registro:

```
$ 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
```

Si ha utilizado la Consola de administración de AWS en el paso anterior, utilice el comando que copió del paso anterior y que contiene estos valores rellenados.

2. Compruebe el buen estado de la implementación de `eks-connector` instalado y espere a que el estado del clúster registrado en Amazon EKS sea `ACTIVE`.

yaml

Complete la conexión aplicando el archivo de manifiesto de Amazon EKS Connector al clúster de Kubernetes. Para ello, debe utilizar los métodos descritos anteriormente. Si el manifiesto no se aplica en un plazo de tres días, el registro de Amazon EKS Connector vence. Si la conexión del clúster vence, debe anularse el registro del clúster antes de volverlo a conectar.

1. Descargue el archivo YAML de Amazon EKS Connector.

```
curl -O https://amazon-eks.s3.us-west-2.amazonaws.com/eks-connector/manifests/eks-connector/latest/eks-connector.yaml
```

2. Edite el archivo YAML de Amazon EKS Connector para reemplazar todas las referencias de `%AWS_REGION%`, `%EKS_ACTIVATION_ID%` y `%EKS_ACTIVATION_CODE%` por `aws-region`, `activationId` y `activationCode` de la salida del paso anterior.

   El siguiente comando de ejemplo puede sustituir estos valores.

```
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**
>
> Asegúrese de que el código de activación esté en el formato base64.

3. En una terminal que pueda acceder al clúster, puede aplicar el archivo de manifiesto actualizado al ejecutar el siguiente comando:

```
kubectl apply -f eks-connector.yaml
```

4. Una vez que los archivos YAML de manifiesto y de enlace de roles de Amazon EKS Connector se hayan aplicado al clúster de Kubernetes, confirme que el clúster esté conectado.

```
aws eks describe-cluster \
     --name "my-first-registered-cluster" \
     --region AWS_REGION
```

   El resultado debe incluir `status=ACTIVE`.

5. (Opcional) Agregue etiquetas a su clúster. Para obtener más información, consulte the section called “Etiquetado de recursos”.

Pasos a seguir a continuación

Si tiene algún problema con estos pasos, consulte the section called “Solución de problemas del conector de EKS”.

Para conceder acceso adicional a entidades principales de IAM a la consola de Amazon EKS para ver los recursos de Kubernetes en un clúster conectado, consulte the section called “Concesión de acceso a clústeres”.

Concesión de acceso para ver los recursos del clúster de Kubernetes en una consola de Amazon EKS

Conceda acceso a entidades principales de IAM a la consola de Amazon EKS para ver la información sobre los recursos de Kubernetes que se ejecutan en el clúster conectado.

Requisitos previos

La entidad principal de IAM que utiliza para acceder a la Consola de administración de AWS debe cumplir con los siguientes requisitos:

- Debe tener el permiso de IAM `eks:AccessKubernetesApi`.
- La cuenta del servicio de Amazon EKS Connector puede suplantar la entidad principal de IAM en el clúster. Esto permite que Amazon EKS Connector asigne la entidad principal de IAM a un usuario de Kubernetes.

Para crear y aplicar el rol del clúster de Amazon EKS Connector

1. Descargue la plantilla de rol de clúster `eks-connector`.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/eks-connector/manifests/eks-connector-console-roles/eks-connector-clusterrole.yaml
```

2. Edición del archivo YAML de la plantilla de roles del clúster. Sustituya las referencias de `%IAM_ARN%` por el nombre de recurso de Amazon (ARN) de la entidad principal de IAM.
3. Aplique el archivo YAML de rol de clúster de Amazon EKS Connector al clúster de Kubernetes.

```
kubectl apply -f eks-connector-clusterrole.yaml
```

Para que una entidad principal de IAM visualice los recursos de Kubernetes en la consola de Amazon EKS, la entidad de principal debe estar asociada a un `role` o `clusterrole` de Kubernetes con los permisos necesarios para leer estos recursos. Para obtener más información, consulte Utilización de la autorización de RBAC en la documentación de Kubernetes.

Para configurar una entidad principal de IAM para que acceda al clúster conectado

1. Puede descargar uno de estos archivos de manifiesto de ejemplo para crear un `clusterrole` y `clusterrolebinding` o un `role` y `rolebinding`, respectivamente:

   Visualización de recursos de Kubernetes en todos los espacios de nombres

   - El rol del clúster de `eks-connector-console-dashboard-full-access-clusterrole` da acceso a todos los espacios de nombres y recursos que se pueden visualizar en la consola. Puede cambiar el nombre de `role`, `clusterrole` y su enlace correspondiente antes de aplicarlos al clúster. Utilice el siguiente comando para descargar un archivo de muestra.

     ```
     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
     ```

   Visualización de recursos de Kubernetes en un espacio de nombres específico

   - El espacio de nombres en este archivo es `default`, así que si desea especificar un espacio de nombres diferente, edite el archivo antes de aplicarlo al clúster. Utilice el siguiente comando para descargar un archivo de muestra.

```
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. Edite el archivo YAML de acceso completo o acceso restringido para reemplazar las referencias de `%IAM_ARN%` por el nombre de recurso de Amazon (ARN) de la entidad principal de IAM.
3. Aplique los archivos YAML de acceso completo o acceso restringido a su clúster de Kubernetes. Reemplace el valor del archivo YAML por el suyo propio.

```
kubectl apply -f eks-connector-console-dashboard-full-access-group.yaml
```

Para ver los recursos de Kubernetes en el clúster conectado, consulte the section called “Acceso a recursos del clúster”. Datos de algunos tipos de recursos de la pestaña Recursos no está disponible para clústeres conectados.

Anulación del registro de un clúster de Kubernetes desde la consola de Amazon EKS

Si ya ha terminado de usar un clúster conectado, puede anular su registro. Una vez que se anule el registro, el clúster ya no estará visible en la consola de Amazon EKS.

Debe tener los siguientes permisos para llamar a la API deregisterCluster:

- `eks:DeregisterCluster`
- `ssm:DeleteActivation`
- `ssm:DeregisterManagedInstance`

Este proceso consta de dos pasos: anular el registro del clúster en Amazon EKS y desinstalar el agente eks-connector del clúster.

Anulación del registro del clúster de Kubernetes

Para anular el registro de un clúster del conector Amazon EKS, puede utilizar una de estas herramientas:

- the section called “AWS CLI”

- the section called “Consola de administración de AWS”
- the section called “`eksctl`”

AWS CLI

1. La CLI de AWS debe estar instalada. Para instalarla o actualizarla, consulte Instalación de la CLI de AWS.
2. Asegúrese de que se haya creado el rol de agente de Amazon EKS Connector.
3. Anule el registro del clúster conectado.

```
aws eks deregister-cluster \
    --name my-cluster \
    --region region-code
```

Consola de administración de AWS

1. Abra la consola de Amazon EKS.
2. Seleccione Clusters (Clústeres).
3. En la página Clusters (Clústeres), seleccione el clúster conectado y seleccione Deregister (Anular registro).
4. Confirme que desea anular el registro del clúster.

`eksctl`

1. Instale la versión de `eksctl 0.68` o una posterior. Para instalarla o actualizarla, consulte the section called “Creación de un clúster (eksctl)”.
2. Asegúrese de que se haya creado el rol de agente de Amazon EKS Connector.
3. Anule el registro del clúster conectado:

```
eksctl deregister cluster --name my-cluster
```

Limpieza de los recursos del clúster de Kubernetes

Para desinstalar el agente de `eks-connector`, utilice uno de los siguientes comandos:

- the section called “helm”
- the section called “yaml”

helm

Ejecute el siguiente comando para desinstalar el agente.

```
helm -n eks-connector uninstall eks-connector
```

yaml

1. Elimine el archivo YAML de Amazon EKS Connector del clúster de Kubernetes.

   ```
   kubectl delete -f eks-connector.yaml
   ```

2. Si ha creado un `clusterrole` o `clusterrolebindings` para que entidades principales de IAM adicionales accedan al clúster, asegúrese de eliminarlas del clúster de Kubernetes.

Solución de problemas del conector de Amazon EKS

En este tema, se tratan algunos de los errores comunes que pueden producirse mientras se usa Amazon EKS Connector y se incluyen las instrucciones sobre cómo resolverlos e implementar soluciones alternativas.

Solución de problemas básicos

En esta sección se describen los pasos para diagnosticar problemas de Amazon EKS Connector.

Verificación del estado de Amazon EKS Connector

Compruebe el estado de Amazon EKS Connector, ingrese:

```
kubectl get pods -n eks-connector
```

Inspección de los registros de Amazon EKS Connector

El pod de Amazon EKS Connector consta de tres contenedores. Para recuperar los registros completos de todos estos contenedores y poder inspeccionarlos, ejecute los siguientes comandos:

- `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
```

Obtención del nombre real del clúster

Los clústeres de Amazon EKS se identifican de forma inequívoca a través del `clusterName` dentro de una cuenta de AWS y una región de AWS. Si tiene varios clústeres conectados en Amazon EKS, puede confirmar qué clúster de Amazon EKS se ha registrado en el clúster de Kubernetes actual. Para ello, ingrese lo siguiente para saber cuál es el `clusterName` del clúster actual.

```
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/"
```

Comandos diversos

Los siguientes comandos son útiles para recuperar la información que necesita para solucionar problemas.

- Utilice el siguiente comando para recopilar imágenes que utilizan los pods en Amazon EKS Connector.

```
kubectl get pods -n eks-connector -o jsonpath="{.items[*].spec.containers[*].image}"
 | tr -s '[[:space:]]' '\n'
```

- Utilice el siguiente comando para determinar los nombres de los nodos en los que se ejecuta Amazon EKS Connector.

```
kubectl get pods -n eks-connector -o jsonpath="{.items[*].spec.nodeName}" | tr -s
 '[[:space:]]' '\n'
```

- Ejecute el siguiente comando para obtener las versiones de cliente y servidor de Kubernetes.

```
kubectl version
```

- Ejecute el siguiente comando para obtener información acerca de los nodos.

```
kubectl get nodes -o wide --show-labels
```

Problema de Helm: 403 Prohibido

Si ha recibido el siguiente error al ejecutar los comandos de instalación de Helm:

```
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
```

Puede ejecutar la siguiente línea para solucionarlo:

```
docker logout public.ecr.aws
```

Error de la consola: El clúster está atascado en el estado Pending

Si después de registrar el clúster, este se queda atascado en el estado `Pending` en la consola de Amazon EKS, tal vez se deba a que Amazon EKS Connector todavía no conectó correctamente el clúster a AWS. Para un clúster registrado, el estado `Pending` significa que la conexión todavía no se ha establecido correctamente. Para resolver este problema, asegúrese de haber aplicado el manifiesto al clúster de Kubernetes de destino. Si lo aplicó al clúster, pero este sigue en estado `Pending`, puede que StatefulSet `eks-connector` esté en mal estado. Para solucionar este problema, consulte the section called “Los pods de Amazon EKS Connector se están bloqueando en bucle” en este tema.

Error de la consola: El usuario system:serviceaccount:eks-connector:eks-connector no puede suplantar los usuarios de recursos en el grupo de la API en el ámbito del clúster.

Amazon EKS Connector utiliza la suplantación de usuarios de Kubernetes para actuar en nombre de las entidades principales de IAM desde la Consola de administración de AWS. A cada entidad principal de IAM que accede a la API de Kubernetes desde la cuenta de servicio de AWS `eks-connector` se le debe conceder permiso para suplantar al usuario de Kubernetes correspondiente con un ARN de IAM como nombre de usuario de Kubernetes. En los ejemplos siguientes, el ARN de IAM se asigna a un usuario de Kubernetes.

- El usuario de IAM *`john`* de la cuenta de AWS *`111122223333`* está asignado a un usuario de Kubernetes. Según las prácticas recomendadas de IAM, se recomienda conceder permisos a los roles en lugar de a los usuarios.

  ```
  arn:aws:iam::111122223333:user/john
  ```

- El rol de IAM *`admin`* de la cuenta de AWS *`111122223333`* está asignado a un usuario de Kubernetes:

  ```
  arn:aws:iam::111122223333:role/admin
  ```

  El resultado es el ARN de un rol de IAM, en lugar del ARN de la sesión de AWS STS.

Para obtener instrucciones sobre cómo configurar `ClusterRole` y `ClusterRoleBinding` a fin de conceder a la cuenta de servicio de `eks-connector` el privilegio para suplantar al usuario asignado, consulte the section called “Concesión de acceso a clústeres”. Asegúrese de que, en la plantilla, `%IAM_ARN%` se sustituya con el ARN de IAM de la entidad principal de IAM de la Consola de administración de AWS.

Error de la consola: […] está prohibido: el usuario […] no puede publicar el recurso […] en grupo de la API en el ámbito del clúster

Considere el siguiente problema. Amazon EKS Connector ha suplantado correctamente a la entidad principal de IAM solicitante de la Consola de administración de AWS en el clúster de Kubernetes de

destino. Sin embargo, la entidad principal suplantada no tiene permiso de RBAC en las operaciones de la API de Kubernetes.

Para resolver este problema, existen dos métodos para conceder permisos a usuarios adicionales. Si anteriormente instaló eks-connector mediante el gráfico de Helm, puede conceder acceso a los usuarios fácilmente ejecutando el siguiente comando. Sustituya `userARN1` y `userARN2` por una lista de los ARN de los roles de IAM para permitir el acceso a la visualización de los recursos de Kubernetes:

```
helm upgrade eks-connector oci://public.ecr.aws/eks-connector/eks-connector-chart \
    --reuse-values \
    --set 'authentication.allowedUserARNs={userARN1,userARN2}'
```

O bien, como administrador del clúster, otorgue el nivel adecuado de privilegios de RBAC a los usuarios individuales de Kubernetes. Para obtener más información y ejemplos, consulta the section called “Concesión de acceso a clústeres”.

Error de la consola: Amazon EKS no puede comunicarse con el servidor de la API del clúster de Kubernetes. El clúster debe estar en estado ACTIVE (ACTIVO) para que la conexión se establezca de forma correcta. Intente establecer la conexión en unos minutos.

Si el servicio Amazon EKS no puede comunicarse con Amazon EKS Connector en el clúster de destino, tal vez se deba a alguno de los siguientes motivos:

- Amazon EKS Connector no funciona bien en el clúster de destino.
- La conectividad es deficiente o la conexión se ha interrumpido entre el clúster de destino y la región de AWS.

Para resolver este problema, consulte los registros de Amazon EKS Connector. Si no aparece un error para Amazon EKS Connector, vuelva a intentar establecer la conexión después de unos minutos. Si experimenta regularmente una alta latencia o una conectividad intermitente para el clúster de destino, considere volver a registrar el clúster en una región de AWS que se encuentre en una ubicación más cercana.

Los pods de Amazon EKS Connector se están bloqueando en bucle

Hay muchas razones que pueden provocar que un pod de Amazon EKS Connector entre en el estado `CrashLoopBackOff`. Es probable que este problema afecte al contenedor `connector-init`. Compruebe el estado del pod de Amazon EKS Connector.

```
kubectl get pods -n eks-connector
```

Un ejemplo de salida sería el siguiente.

```
NAME              READY   STATUS                  RESTARTS   AGE
eks-connector-0   0/2     Init:CrashLoopBackOff   1          7s
```

Si el resultado es similar al resultado anterior, consulte the section called “Inspección de los registros de Amazon EKS Connector” para solucionar el problema.

Error al iniciar eks-connector: InvalidActivation

Cuando inicia Amazon EKS Connector por primera vez, este registra un `activationId` y un `activationCode` en Amazon Web Services. El registro puede fallar, lo que podría provocar que el contenedor `connector-init` se bloquee con un error similar al siguiente.

```
F1116 20:30:47.261469       1 init.go:43] failed to initiate eks-connector:
 InvalidActivation:
```

Para solucionar este problema, tenga en cuenta las siguientes causas y correcciones recomendadas:

- Es posible que el registro haya fallado porque el `activationId` y el `activationCode` no están en el archivo de manifiesto. En este caso, asegúrese de que se devolvieron los valores correctos de la operación `RegisterCluster` de la API y de que el `activationCode` se encuentra en el archivo de manifiesto. El `activationCode` se agrega a los secretos de Kubernetes, por lo que debe estar codificado en `base64`. Para obtener más información, consulte the section called “Paso 1: registro del clúster”.
- Es posible que el registro haya fallado porque la activación venció. Esto se debe a que, por razones de seguridad, debe activar Amazon EKS Connector en un plazo de tres días después del registro del clúster. Para resolver este problema, asegúrese de haber aplicado el manifiesto de Amazon EKS Connector al clúster de Kubernetes de destino antes de la fecha y la hora de

vencimiento. Para confirmar la fecha de vencimiento de la activación, llame a la operación de la API `DescribeCluster`.

```
aws eks describe-cluster --name my-cluster
```

En la siguiente respuesta de ejemplo, la fecha y la hora de vencimiento se registran como `2021-11-12T22:28:51.101000-08:00`.

```
{
    "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"
        }
    }
}
```

Si `activationExpiry` ya transcurrió, anule el registro del clúster y vuelva a registrarlo. Hacer esto genera una nueva activación.

El nodo del clúster no tiene conectividad de salida

Para que funcione correctamente, Amazon EKS Connector requiere conectividad de salida a varios puntos de conexión de AWS. No se puede conectar un clúster privado sin conectividad de salida a una región de AWS de destino. Para resolver este problema, debe agregar la conectividad de salida necesaria. Para obtener información sobre los requisitos de los conectores, consulte the section called “Consideraciones sobre Amazon EKS Connector”.

Pods de Amazon EKS Connector con el estado `ImagePullBackOff`

Si ejecuta el comando `get pods` y los pods se encuentran en el estado `ImagePullBackOff`, no pueden funcionar correctamente. Si los pods de Amazon EKS Connector se encuentran en el estado `ImagePullBackOff`, no pueden funcionar correctamente. Compruebe el estado de los pods de Amazon EKS Connector.

```
kubectl get pods -n eks-connector
```

Un ejemplo de salida sería el siguiente.

```
NAME              READY   STATUS                  RESTARTS   AGE
eks-connector-0   0/2     Init:ImagePullBackOff   0          4s
```

El archivo de manifiesto predeterminado de Amazon EKS Connector hace referencia a imágenes de Galería pública de Amazon ECR. Es posible que el clúster de Kubernetes de destino no pueda extraer imágenes de Galería pública de Amazon ECR. Resuelva el problema de extracción de imágenes de Galería pública de Amazon ECR o considere la posibilidad de reflejar las imágenes en el registro de contenedores privado de su elección.

Preguntas frecuentes sobre el conector de AWS

P: ¿Cómo funciona la tecnología subyacente detrás de Amazon EKS Connector?

R: Amazon EKS Connector se basa en el agente de AWS Systems Manager (Systems Manager). Amazon EKS Connector se ejecuta como `StatefulSet` en el clúster de Kubernetes. Establece una conexión y actúa como proxy para la comunicación entre el servidor de la API de su clúster y Amazon Web Services. Lo hace para mostrar los datos del clúster en la consola de Amazon EKS hasta que desconecte el clúster de AWS. El agente de Systems Manager es un proyecto de código abierto. Para obtener más información acerca de este proyecto, consulte la página de proyecto de GitHub.

P: Tengo un clúster de Kubernetes en las instalaciones que quiero conectar. ¿Debo abrir los puertos del firewall para conectarlo?

R: No, no es necesario abrir ningún puerto de firewall. El clúster de Kubernetes solo requiere conexión de salida con las regiones de AWS. Los servicios de AWS nunca acceden a los

recursos en la red en las instalaciones. Amazon EKS Connector se ejecuta en el clúster e inicia la conexión con AWS. Cuando se completa el registro del clúster, AWS solo emite comandos para Amazon EKS Connector después de que usted inicie una acción desde la consola de Amazon EKS que requiere información del servidor de la API de Kubernetes en el clúster.

P: ¿Qué datos envía Amazon EKS Connector desde mi clúster a AWS?

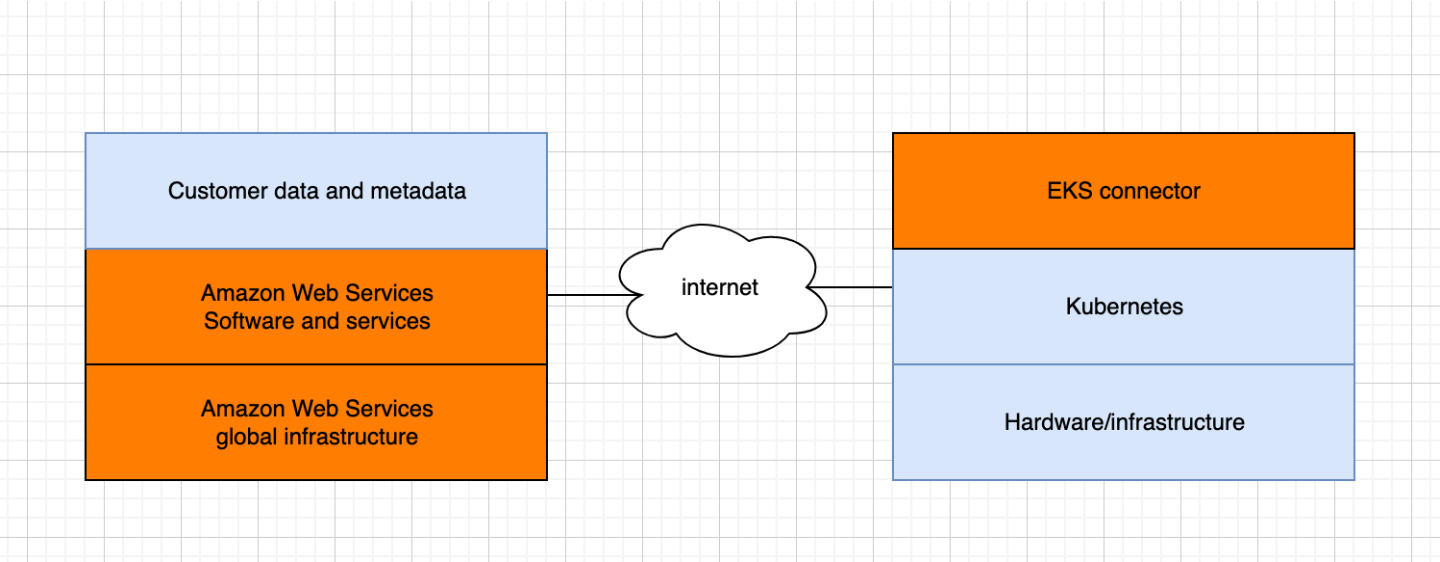
R: Amazon EKS Connector envía la información técnica necesaria para que su clúster se registre en AWS. También envía metadatos del clúster y la carga de trabajo para las características de la consola de Amazon EKS que solicitan los clientes. Amazon EKS Connector solo recopila o envía estos datos si usted inicia una acción desde la consola o la API de Amazon EKS que requiere que los datos se envíen a AWS. Aparte del número de versión de Kubernetes, AWS no almacena ningún dato de forma predeterminada. Los almacena solo si usted lo autoriza.

P: ¿Puedo conectar un clúster fuera de una región de AWS?

R: Sí, puede conectar un clúster desde cualquier ubicación con Amazon EKS. Además, su servicio Amazon EKS se puede ubicar en cualquier región de AWS comercial pública de AWS. Esto funciona con una conexión de red válida entre el clúster y la región de AWS de destino. Le recomendamos elegir la región de AWS más cercana a la ubicación del clúster para optimizar el rendimiento de la interfaz del usuario. Por ejemplo, si tiene un clúster en ejecución en Tokio, conecte el clúster a la región de AWS ubicada en Tokio (es decir, la región `ap-northeast-1` de AWS) para obtener una latencia baja. Puede conectar un clúster desde cualquier ubicación con Amazon EKS en cualquiera de las regiones de AWS comerciales públicas, excepto las regiones de China o GovCloud de AWS.

Comprensión de la seguridad de Amazon EKS Connector

Amazon EKS Connector es un componente de código abierto que se ejecuta en el clúster de Kubernetes. Este clúster se puede ubicar fuera del entorno AWS. Esto crea consideraciones adicionales para las responsabilidades de seguridad. El siguiente diagrama ilustra esta configuración. Naranja representa responsabilidades de AWS y azul representa las responsabilidades del cliente:



En este tema se describen las diferencias en el modelo de responsabilidad si el clúster conectado está fuera de AWS.

Responsabilidades de AWS

- Mantenimiento, creación y entrega de Amazon EKS Connector, que es un componente de código abierto que se ejecuta en el clúster de Kubernetes de un cliente y se comunica con AWS.
- Mantenimiento de la seguridad de la comunicación del nivel de aplicaciones y del transporte entre el clúster de Kubernetes conectado y servicios de AWS.

Responsabilidades del cliente

- Seguridad específica del clúster de Kubernetes, específicamente en las siguientes líneas:
  - Los secretos de Kubernetes deben estar cifrados y protegidos correctamente.
  - Bloquee el acceso al espacio de nombres de `eks-connector`.
- Configuración de permisos de control de acceso basado en roles (RBAC) para administrar el acceso de entidades principales de IAM desde AWS. Para obtener instrucciones, consulte the section called “Concesión de acceso a clústeres”.
- Instalación y actualización de Amazon EKS Connector.
- Mantenimiento del hardware, el software y la infraestructura que admite el clúster de Kubernetes conectado.

- Seguridad de sus cuentas de AWS (por ejemplo, protegiendo sus credenciales de usuario raíz seguro).

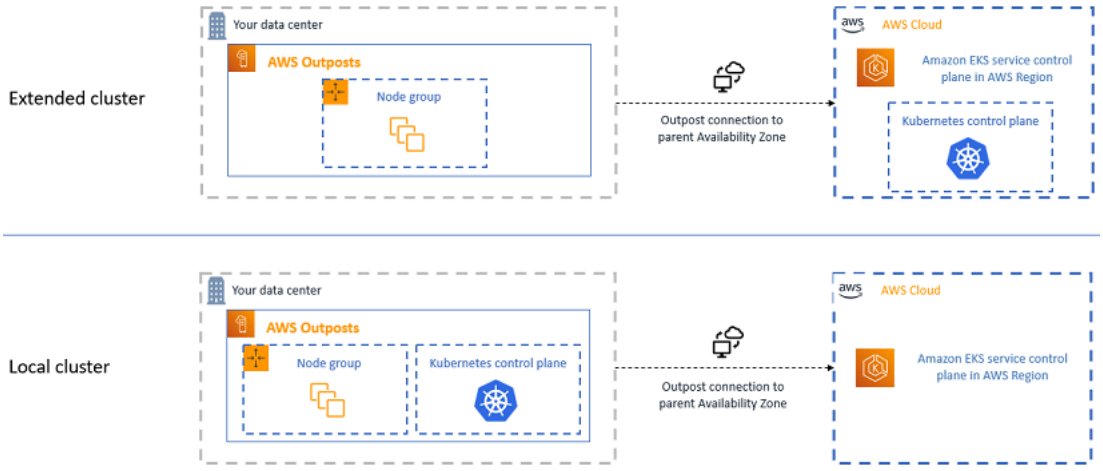
Implementación de Amazon EKS en las instalaciones con AWS Outposts

Puede utilizar Amazon EKS para ejecutar aplicaciones de Kubernetes en las instalaciones con AWS Outposts. Puede implementar Amazon EKS en Outposts de las siguientes formas:

- Clústeres extendidos: ejecute el plano de control de Kubernetes en una región de AWS y los nodos en su Outpost.
- Clústeres locales: ejecute el plano de control de Kubernetes y nodos en el Outpost.

Para ambas opciones de implementación, el plano de control de Kubernetes está completamente administrado por AWS. Puede usar las mismas API, herramientas y consola de Amazon EKS que usa en la nube para crear y ejecutar Amazon EKS en Outposts.

En el siguiente diagrama, se muestran estas opciones de implementación.



Cuándo usar cada opción de implementación

Tanto los clústeres locales como los extendidos son opciones de implementación de uso general y se pueden usar para una variedad de aplicaciones.

Con los clústeres locales, puede ejecutar todo el clúster de Amazon EKS de forma local en Outposts. Esta opción puede mitigar el riesgo de tiempo de inactividad de las aplicaciones que puede resultar de la desconexión temporal de la red a la nube. Estas desconexiones de red pueden deberse a cortes de fibra o eventos climáticos. Dado que todo el clúster de Amazon EKS se ejecuta de forma local en Outposts, las aplicaciones permanecen disponibles. De esta manera, puede realizar

operaciones de clúster durante las desconexiones de red a la nube. Para obtener más información, consulte the section called “Preparación para las desconexiones”. Si le preocupa la calidad de la conexión de red de sus Outposts a la región de AWS principal y requiere una alta disponibilidad durante desconexiones de red, use la opción de implementación de clúster local.

Los clústeres extendidos le permiten conservar la capacidad de su Outpost porque el plano de control de Kubernetes se ejecuta en la región de AWS principal. Esta puede ser la opción más adecuada si puede invertir en una conectividad de red confiable y redundante desde su Outpost hasta la región de AWS. La calidad de la conexión de red es fundamental para esta opción. La forma en que Kubernetes administra las desconexiones de red entre el plano de control de Kubernetes y los nodos puede provocar un tiempo de inactividad de la aplicación. Para obtener más información sobre el comportamiento de Kubernetes, consulte Scheduling, Preemption, and Eviction en la documentación de Kubernetes.

Comparación de las opciones de implementación

En la siguiente tabla, se comparan las diferencias entre las dos opciones.

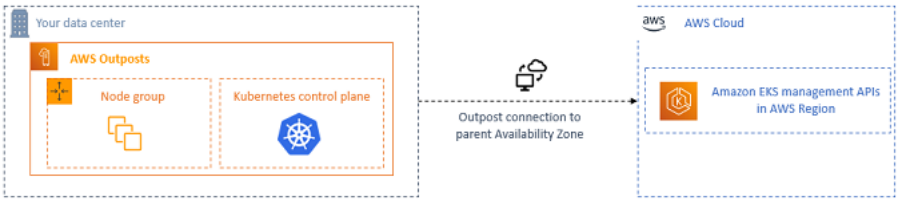
Característica	Clúster extendido	Clúster local
Ubicación del plano de control de Kubernetes	Región de AWS	Outpost
Cuenta del plano de control de Kubernetes	Cuenta de AWS	Su cuenta
Disponibilidad regional	consulte Service endpoints	Este de EE. UU. (Ohio), Este de EE. UU. (Norte de Virginia), Oeste de EE. UU. (Norte de California), Oeste de EE. UU. (Oregón), Asia-Pacífico (Seúl), Asia-Pacífico (Singapur), Asia-Pacífico (Sídney), Asia-Pacífico (Tokio), Canadá (centro), Europa (Fráncfort), Europa (Irlanda), Europa (Londres), Medio Oriente (Baréin) y América del Sur (São Paulo).

Característica	Clúster extendido	Clúster local
Versiones secundarias de Kubernetes	eks/latest/userguide/kubernetes-versions.html[Supported Amazon EKS versions, type="documentation"].	eks/latest/userguide/kubernetes-versions.html[Supported Amazon EKS versions, type="documentation"].
Versiones de la plataforma	Consulte Versiones de la plataforma de EKS.	Consulte the section called “Versiones de la plataforma de EKS”
Factores de forma de Outpost	Bastidores de Outpost	Bastidores de Outpost
Interfaces de usuario	Consola de administración de AWS, AWS CLI, API de Amazon EKS, `eksctl`, AWS CloudFormation y Terraform	Consola de administración de AWS, AWS CLI, API de Amazon EKS, `eksctl`, AWS CloudFormation y Terraform
Políticas administradas	AmazonEKSClusterPolicy y the section called “Política administrada de AWS: AmazonEKSServiceRolePolicy”	AmazonEKSLocalOutpostClusterPolicy y the section called “Política administrada de AWS: AmazonEKSLocalOutpostServiceRolePolicy”
VPC y subredes del clúster	Consulte the section called “Requisitos de VPC y subred”	Consulte the section called “Creación de una VPC y de subredes”
Acceso al punto de conexión del clúster	Público o privado o ambos	Solo privada
Autenticación del servidor de la API de Kubernetes	AWS Identity and Access Management (IAM) y OIDC	IAM y certificados `x.509`
Tipos de nodos	Solo autoadministrado	Solo autoadministrado
Tipos de computación de nodos	Amazon EC2 bajo demanda	Amazon EC2 bajo demanda

Característica	Clúster extendido	Clúster local
Tipos de almacenamiento de nodos	gp2 de Amazon EBS y SSD de NVMe local	gp2 de Amazon EBS y SSD de NVMe local
AMI optimizadas para Amazon EKS	Amazon Linux, Windows y Bottlerocket	Solo Amazon Linux
Versiones de IP	Sólo `IPv4`	Sólo `IPv4`
Complementos	Complementos de Amazon EKS o complementos autoadministrados	Solo complementos autoadministrados
Interfaz de red de contenedores predeterminada	Complemento CNI de Amazon VPC para Kubernetes	Complemento CNI de Amazon VPC para Kubernetes
Registros del plano de control de Kubernetes	Registros de Amazon CloudWatch	Registros de Amazon CloudWatch
Equilibrio de carga	Utilice el Controlador del equilibrador de carga de AWS para aprovisionar solo equilibradores de carga de aplicación (no equilibradores de carga de red)	Utilice el Controlador del equilibrador de carga de AWS para aprovisionar solo equilibradores de carga de aplicación (no equilibradores de carga de red)
Cifrado de sobre para secretos	Consulte the section called “Habilitación del cifrado de secretos”	No compatible
Roles de IAM para cuentas de servicio	Consulte the section called “Credenciales con IRSA”	No compatible
Solución de problemas	Consulte Solución de problemas	Consulte the section called “Solución de problemas de clústeres”

Creación de los clústeres locales de Amazon EKS en AWS Outposts para obtener alta disponibilidad

Puede utilizar clústeres locales para ejecutar todo su clúster de Amazon EKS de forma local en AWS Outposts. Esto ayuda a mitigar el riesgo de tiempo de inactividad de las aplicaciones que puede resultar de la desconexión temporal de la red a la nube. Estas desconexiones pueden deberse a cortes de fibra o eventos climáticos. Dado que todo el clúster de Kubernetes se ejecuta de forma local en Outposts, las aplicaciones permanecen disponibles. De esta manera, puede realizar operaciones de clúster durante las desconexiones de red a la nube. Para obtener más información, consulte the section called “Preparación para las desconexiones”. En el siguiente diagrama, se muestra una implementación de clúster local.



Los clústeres locales suelen estar disponibles para su uso con los bastidores de Outposts.

Regiones de AWS compatibles

Puede crear clústeres locales en las siguientes regiones de AWS: Este de EE. UU. (Ohio), Este de EE. UU. (Norte de Virginia), Oeste de EE. UU. (Norte de California), Oeste de EE. UU. (Oregón), Asia-Pacífico (Seúl), Asia-Pacífico (Singapur), Asia-Pacífico (Sídney), Asia-Pacífico (Tokio), Canadá (centro), Europa (Fráncfort), Europa (Irlanda), Europa (Londres), Medio Oriente (Baréin) y América del Sur (São Paulo). Para obtener información detallada sobre las características admitidas, consulte the section called “Comparación de las opciones de implementación”.

Implementación de un clúster de Amazon EKS en AWS Outposts

En este tema, se proporciona información general sobre lo que debe tener en cuenta al ejecutar un clúster local en un Outpost. El tema también contiene instrucciones sobre cómo implementar un clúster local en un Outpost.

> **Important**
>
> - Estas consideraciones no se replican en la documentación relacionada de Amazon EKS. Si otros temas de la documentación de Amazon EKS entran en conflicto con las

consideraciones que se presentan aquí, siga las consideraciones que se indican a continuación.

- Estas consideraciones están sujetas a modificaciones y pueden cambiar con frecuencia. Por lo tanto, le recomendamos que revise este tema con regularidad.
- Muchas de las consideraciones son diferentes a las consideraciones para crear un clúster en la nube de AWS.

- Los clústeres locales solo admiten bastidores de Outpost. Un único clúster local puede ejecutarse en varios bastidores de Outpost físicos que comprenden un único Outpost lógico. Un único clúster local no puede ejecutarse en varios Outposts lógicos. Cada Outpost lógico tiene un único ARN de Outpost.
- Los clústeres locales ejecutan y administran el plano de control de Kubernetes en su cuenta en Outpost. No se pueden ejecutar cargas de trabajo en las instancias del plano de control de Kubernetes ni modificar los componentes del plano de control de Kubernetes. Estos nodos están administrados por el servicio de Amazon EKS. Los cambios en el plano de control de Kubernetes no persisten a través de las acciones de administración automáticas de Amazon EKS, como la aplicación de parches.
- Los clústeres locales admiten complementos autoadministrados y grupos de nodos autoadministrados de Amazon Linux. El complemento CNI de Amazon VPC para los complementos Kubernetes, kube-proxy y CoreDNS se instala automáticamente en los clústeres locales.
- Los clústeres locales requieren el uso de Amazon EBS en Outposts. Su Outpost debe tener Amazon EBS disponible para el almacenamiento del plano de control de Kubernetes. Los Outposts admiten únicamente volúmenes `gp2` de Amazon EBS.
- Los `PersistentVolumes` de Kubernetes basadas en Amazon EBS se admiten mediante el controlador de CSI de Amazon EBS.
- Las instancias del plano de control de los clústeres locales se configuran en topología apilada de alta disponibilidad. Dos de las tres instancias del plano de control deben estar en buen estado en todo momento para mantener el quórum. Si se pierde el quórum, contacte a soporte de AWS, ya que se requerirán algunas acciones del lado del servicio para habilitar las nuevas instancias administradas.

Requisitos previos

- Familiaridad con las opciones de implementación de Outposts. Seleccione los tipos de instancias y los grupos de ubicación para los clústeres de Amazon EKS en AWS Outposts en función de las consideraciones de capacidad y los requisitos y las consideraciones de VPC.
- Un Outpost existente. Para obtener más información, consulte ¿Qué es AWS Outposts?
- La herramienta de la línea de comandos de `kubectl` está instalada en su equipo o AWS CloudShell. La versión puede ser la misma o hasta una versión secundaria anterior o posterior a la versión de Kubernetes de su clúster. Por ejemplo, si la versión del clúster es `1.29`, puede usar la versión `1.28`, `1.29` o `1.30` de `kubectl` con él. Para instalar o actualizar `kubectl`, consulte the section called “Configure kubectl y eksctl”.
- La versión `2.12.3` o posterior, o bien, la versión `1.27.160` o posterior de la AWS interfaz de la línea de comandos (AWS CLI) instalada y configurada en su dispositivo o AWS CloudShell. Para comprobar su versión actual, utilice `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente. Para actualizarla, consulte Instalación de la CLI de AWS en su directorio de usuarios principal en la Guía del usuario de AWS CloudShell.
- Una entidad principal de IAM (usuario o rol) con permisos para `create` y `describe` un clúster de Amazon EKS. Para obtener más información, consulte the section called “Creación de un clúster local de Kubernetes en un Outpost” y the section called “Enumeración o descripción de todos los clústeres”.

Cuando se crea un clúster local de Amazon EKS, la entidad principal de IAM que crea el clúster se agrega de manera permanente. La entidad principal de IAM se agrega específicamente a la tabla de autorizaciones RBAC de Kubernetes como administrador. Esta entidad tiene permisos `system:masters`. La identidad de esta entidad no está visible en la configuración del clúster. Por lo tanto, es importante anotar la entidad que creó el clúster y asegurarse de no eliminarlo nunca. Al principio, solo la entidad principal que creó el servidor puede hacer llamadas al servidor de la API de Kubernetes con `kubectl`. Si usa la consola para crear el clúster, debe asegurarse de que las mismas credenciales de IAM se encuentren en la cadena de credenciales del SDK de AWS al ejecutar comandos de `kubectl` en el clúster. Una vez que se crea el clúster, puede conceder acceso a su clúster a otras entidades principales de IAM.

Creación de un clúster local de Amazon EKS

Puede crear un clúster local con las siguientes herramientas que se describen en esta página:

- the section called “eksctl”
- the section called “Consola de administración de AWS”

También puede utilizar la AWS CLI, la API de Amazon EKS, los AWS SDK, AWS CloudFormation o Terraform para crear clústeres en Outposts.

`eksctl`

Para crear un clúster con **`eksctl`**

1. Instale la versión `0.215.0` o posterior de la herramienta de línea de comandos de `eksctl` instalada en su dispositivo o AWS CloudShell. Para instalar o actualizar `eksctl`, consulte la sección de Instalación en la documentación de `eksctl`.
2. Copie los siguientes contenidos en su dispositivo. Reemplace los siguientes valores y, a continuación, ejecute el comando modificado para crear el archivo `outpost-control-plane.yaml`:
   - Reemplace *`region-code`* por la región de AWS admitida en la que desea crear su clúster.
   - Reemplace *`my-cluster`* por el nombre de su clúster. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster.
   - Reemplace *`vpc-ExampleID1`* y *`subnet-ExampleID1`* con los ID de su VPC y subred existentes. La VPC y la subred deben cumplir los requisitos en Creación de una VPC y subredes para clústeres de Amazon EKS en AWS Outposts.
   - Reemplace *`uniqueid`* con el ID de su Outpost.
   - Reemplace *`m5.large`* con un tipo de instancia disponible en su Outpost. Antes de elegir un tipo de instancia, consulte the section called “Consideraciones de capacidad”. Se implementan tres instancias del plano de control. No puede cambiar este número.

```
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
```

Para obtener una lista completa de todas las opciones y valores predeterminados disponibles, consulte Soporte de AWS Outposts y Esquema del archivo de configuración en la documentación de `eksctl`.

3. Para crear el clúster, use el archivo de configuración que creó en el paso anterior. `eksctl` crea una VPC y una subred en su Outpost para implementar el clúster.

```
eksctl create cluster -f outpost-control-plane.yaml
```

El aprovisionamiento de clústeres tarda varios minutos. Mientras se crea el clúster, aparecen varias líneas de salida. La última línea de salida es similar a la siguiente línea de ejemplo.

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

> **Tip**
>
> Para ver la mayoría de las opciones que se pueden especificar al crear un clúster con `eksctl`, utilice el comando `eksctl create cluster --help`. Para ver todas las opciones disponibles, puede utilizar un archivo `config`. Para obtener más información,

consulte Uso de archivos de configuración y el esquema de archivos de configuración en la documentación de `eksctl`. Puede encontrar ejemplos de archivos de configuración en GitHub.

El comando `eksctl` creó automáticamente una entrada de acceso para la entidad principal de IAM (usuario o rol) que creó el clúster y concedió al administrador de la entidad principal de IAM permisos para los objetos de Kubernetes en el clúster. Si no desea que el creador del clúster tenga acceso de administrador a los objetos de Kubernetes en el clúster, agregue el siguiente texto al archivo de configuración anterior: `bootstrapClusterCreatorAdminPermissions: false` (al mismo nivel que `metadata`, `vpc` y `outpost`). Si agregó la opción, después de crear el clúster, tendrá que crear una entrada de acceso para al menos una entidad principal de IAM; de lo contrario, ninguna entidad principal de IAM tendrá acceso a los objetos de Kubernetes en el clúster.

Consola de administración de AWS

Para crear el clúster con la Consola de administración de AWS

1. Necesita una VPC y subred existentes que cumplan con los requisitos de Amazon EKS. Para obtener más información, consulte the section called “Creación de una VPC y de subredes”.
2. Si ya tiene un rol de IAM del clúster local o va a crear su clúster con `eksctl`, puede omitir este paso. Por defecto, `eksctl` crea un rol.
   a. Ejecute el siguiente comando para crear un archivo de política de confianza JSON de IAM.

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

b. Creación del rol de IAM del clúster de Amazon EKS. Para crear un rol de IAM, a la entidad principal de IAM que está creando el rol se le debe asignar la acción `iam:CreateRole` (permiso).

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

c. Adjunte la política administrada de Amazon EKS con el nombre AmazonEKSLocalOutPostClusterPolicy al rol. Para adjuntar una política de IAM a una entidad principal de IAM, se debe asignar una de las siguientes acciones de IAM (permisos) a la entidad principal que adjunta la política: `iam:AttachUserPolicy` o `iam:AttachRolePolicy`.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSLocalOutpostClusterPolicy --role-name myAmazonEKSLocalClusterRole
```

3. Abra la consola de Amazon EKS.
4. En la parte superior de la pantalla de la consola, asegúrese de haber seleccionado una región de AWS admitida.
5. Elija Agregar clúster y, a continuación, elija Crear.
6. En la página Configurar clúster, rellene o seleccione los valores para los siguientes campos:
   - Ubicación del plano de control de Kubernetes: elija AWS Outposts.
   - ID de Outpost: elija el ID del Outpost en el que desea crear su plano de control.
   - Tipo de instancia: seleccione un tipo de instancia. Solo se muestran los tipos de instancias disponibles en su Outpost. En la lista desplegable, cada tipo de instancia describe para cuántos nodos se recomienda el tipo de instancia. Antes de elegir un tipo de instancia, consulte the section called “Consideraciones de capacidad”. Todas las réplicas se implementan con el mismo tipo de instancia. Después de crear el clúster, no se puede cambiar el tipo de instancia. Se implementan tres instancias del plano de control. No puede cambiar este número.
   - Nombre: un nombre único para el clúster. Debe ser único en la cuenta de AWS. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster.

- Versión de Kubernetes: elija la versión de Kubernetes que desea utilizar para el clúster. Le recomendamos seleccionar la versión más reciente, a menos que necesite usar una versión anterior.
- Rol de servicio de clúster: elija el rol de IAM del clúster de Amazon EKS que creó en un paso anterior para permitir que el plano de control de Kubernetes administre los recursos de AWS.
- Acceso de administrador del clúster de Kubernetes: si desea que la entidad principal de IAM (rol o usuario) que está creando el clúster tenga acceso de administrador a los objetos de Kubernetes en el clúster, acepte la opción predeterminada (Permitir). Amazon EKS crea una entrada de acceso para la entidad principal de IAM y concede permisos de administrador de clúster a la entrada de acceso. Para obtener más información acerca de las entradas de acceso, consulte . the section called “Entradas de los registros”.

  Si desea que una entidad principal de IAM diferente a la entidad principal que creó el clúster tenga acceso de administrador a los objetos del clúster de Kubernetes, seleccione la opción No permitir. Después de crear el clúster, cualquier entidad principal de IAM que tenga permisos de IAM para crear entradas de acceso puede agregar una entrada de acceso para cualquier entidad principal de IAM que necesite acceder a los objetos del clúster de Kubernetes. Para obtener más información sobre los permisos necesarios de IAM, consulte Acciones definidas por Amazon Elastic Kubernetes Service en la Referencia de autorización de servicios. Si elige la opción No permitir y no crea ninguna entrada de acceso, ninguna entidad principal de IAM tendrá acceso a los objetos del clúster de Kubernetes.
- Etiquetas: (opcional) agregue las etiquetas a su clúster. Para obtener más información, consulte the section called “Etiquetado de recursos”. Cuando haya terminado con esta página, seleccione Siguiente.

7. En la página Especificar red, seleccione valores para los siguientes campos:
   - VPC: elija una VPC existente. La VPC debe tener un número suficiente de direcciones IP disponibles para el clúster, los nodos y otros recursos de Kubernetes que desee crear. La VPC debe cumplir los Requisitos y consideraciones de la VPC.
   - Subredes: de forma predeterminada, se preseleccionan todas las subredes disponibles de la VPC especificada en el campo anterior. Las subredes que elija deben cumplir los Requisitos y consideraciones de las subredes.
   - Grupos de seguridad: (opcional) especifique uno o varios grupos de seguridad que desea que Amazon EKS asocie a las interfaces de red que crea. Amazon EKS crea automáticamente un grupo de seguridad que habilita la comunicación entre el clúster y la VPC. Amazon EKS asocia este grupo de seguridad, y el que elija, a las interfaces de red que crea. Para obtener

más información acerca del grupo de seguridad de clúster que crea Amazon EKS, consulte the section called “Requisitos del grupo de seguridad”. Puede modificar las reglas del grupo de seguridad en el clúster que crea Amazon EKS. Si elige agregar sus propios grupos de seguridad, no puede cambiar los que elija tras la creación del clúster. Para que los hosts en las instalaciones se comuniquen con el punto de conexión del clúster, debe permitir el tráfico saliente desde el grupo de seguridad del clúster. Para los clústeres que no tienen una conexión a Internet de entrada y salida (también conocidos como clústeres privados), debe realizar una de las siguientes acciones:

- Agregue el grupo de seguridad asociado a los puntos de conexión de VPC requeridos. Para obtener más información acerca de los puntos de conexión requeridos, consulte the section called “Uso de los puntos de conexión de VPC de interfaz” en Acceso de subred a los servicios de AWS.
- Modifique el grupo de seguridad que creó Amazon EKS para permitir el tráfico del grupo de seguridad asociado a los puntos de conexión de VPC. Cuando haya terminado con esta página, seleccione Siguiente.

8. En la página Configurar observabilidad, si lo desea, puede elegir qué opciones de métricas y registro de plano de control quiere activar. De forma predeterminada, cada tipo de registro está desactivado.
   - Para obtener más información sobre la opción de las métricas de Prometheus, consulte the section called “Paso 1: activación de métricas de Prometheus”.
   - Para obtener más información sobre las opciones de Registro de plano de control, consulte the section called “Registros del plano de control”. Cuando haya terminado con esta página, seleccione Siguiente.
9. En la página Revisar y crear, revise la información que introdujo o seleccionó en las páginas anteriores. Si necesita realizar cambios, seleccione Edit (Editar). Cuando esté satisfecho, seleccione Crear. El campo Estado muestra CREANDO mientras se aprovisiona el clúster.

   El aprovisionamiento de clústeres tarda varios minutos.

Visualización del clúster local de Amazon EKS

1. Una vez creado el clúster, puede ver las instancias del plano de control de Amazon EC2 que se crearon.

```
aws ec2 describe-instances --query 'Reservations[*].Instances[*].{Name:Tags[?
Key==`Name`]|[0].Value}' | grep my-cluster-control-plane
```

Un ejemplo de salida sería el siguiente.

```
"Name": "my-cluster-control-plane-id1"
"Name": "my-cluster-control-plane-id2"
"Name": "my-cluster-control-plane-id3"
```

Cada instancia tiene un taint `node-role.eks-local.amazonaws.com/control-plane` aplicado, de modo que nunca se programen cargas de trabajo en las instancias del plano de control. Para obtener más información sobre las taints, consulte Taints and Tolerations en la documentación de Kubernetes. Amazon EKS supervisa continuamente el estado de los clústeres locales. Realizamos acciones de administración automáticas, como la aplicación de parches de seguridad y la reparación de instancias en mal estado. Cuando los clústeres locales se desconectan de la nube, realizamos acciones para garantizar que el clúster vuelva a un buen estado al volver a conectarse.

2. Si ha creado el clúster mediante `eksctl`, puede omitir este paso. `eksctl` completa este paso. Habilite `kubectl` para comunicarse con el clúster agregando un nuevo contexto al archivo `kubectl config`. Para obtener instrucciones acerca de cómo crear y actualizar el archivo, consulte the section called “Acceso al clúster con kubectl”.

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

Un ejemplo de salida sería el siguiente.

```
Added new context arn:aws:eks:region-code:111122223333:cluster/my-cluster to /home/
username/.kube/config
```

3. Para conectarse al servidor de API de Kubernetes de su clúster local, debe tener acceso a la puerta de enlace local de la subred o conectarse desde la VPC. Para obtener más información acerca de la conexión de un bastidor de Outpost a su red en las instalaciones, consulte Cómo funcionan las puertas de enlace locales para bastidores en la Guía del usuario de AWS Outposts. Si usa el enrutamiento directo de VPC y la subred de Outpost tiene una ruta a la puerta de enlace local, las direcciones IP privadas de las instancias del plano de control de Kubernetes se transmiten automáticamente a través de la red local. El punto de conexión del servidor de la

API de Kubernetes del clúster local está alojado en Amazon Route 53 (Route 53). Los servidores de DNS públicos pueden resolver el punto de conexión del servicio de API en las direcciones IP privadas de los servidores de la API de Kubernetes.

Las instancias del plano de control de Kubernetes de los clústeres locales se configuran con interfaces de red elásticas estáticas con direcciones IP privadas fijas que no cambian a lo largo del ciclo de vida del clúster. Es posible que las máquinas que interactúan con el servidor de la API de Kubernetes no tengan conectividad con Route 53 durante desconexiones de red. Si este es el caso, recomendamos configurar `/etc/hosts` con las direcciones IP privadas estáticas para continuar con las operaciones. También recomendamos configurar servidores de DNS locales y conectarlos a su Outpost. Para obtener más información, consulte la documentación de AWSOutposts. Ejecute el siguiente comando para confirmar que se ha establecido la comunicación con el clúster.

```
kubectl get svc
```

Un ejemplo de salida sería el siguiente.

```
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   28h
```

4. (Opcional) Pruebe la autenticación en su clúster local cuando se encuentra desconectado de la nube de AWS. Para obtener instrucciones, consulte the section called “Preparación para las desconexiones”.

Recursos internos

Amazon EKS crea los siguientes recursos en su clúster. Los recursos son para uso interno de Amazon EKS. Para que el clúster funcione correctamente, no edite ni modifique estos recursos.

- Los siguientes pods de reflejo:
  - `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`*

- Los siguientes complementos autoadministrados:
  - `kube-system/coredns`
  - `kube-system/ kube-proxy` (no se crean hasta que agregue su primer nodo)
  - `kube-system/aws-node` (no se crea hasta que agregue su primer nodo). Los clústeres locales usan el complemento CNI de Amazon VPC para Kubernetes para redes de clústeres. No cambie la configuración de las instancias del plano de control (pods denominados `aws-node-controlplane-*`). Hay variables de configuración que puede usar para cambiar el valor predeterminado para cuando el complemento crea interfaces de red nuevas. Para obtener más información, consulte la documentación en GitHub.
- Los siguientes servicios:
  - `default/kubernetes`
  - `kube-system/kube-dns`
- Una `PodSecurityPolicy` denominada `eks.system`
- Una `ClusterRole` denominada `eks:system:podsecuritypolicy`
- Una `ClusterRoleBinding` denominada `eks:system`
- Además del grupo de seguridad del clúster, Amazon EKS crea un grupo de seguridad en su cuenta de AWS llamado `eks-local-internal-do-not-use-or-edit-`*`cluster-name`*`-`*`uniqueid`*. Este grupo de seguridad permite que el tráfico fluya libremente entre los componentes de Kubernetes que se ejecutan en las instancias del plano de control.

Siguientes pasos recomendados:

- Conceda a la entidad principal de IAM que creó el clúster los permisos necesarios para ver los recursos de Kubernetes en la Consola de administración de AWS
- Conceda acceso a las entidades de IAM al clúster. Si desea que las entidades vean los recursos de Kubernetes en la consola de Amazon EKS, conceda los Permisos requeridos a las entidades.
- Configure el registro para el clúster
- Familiarícese con lo que sucede durante las desconexiones de red.
- Agregue nodos al clúster
- Considere la posibilidad de configurar un plan de copia de seguridad para su `etcd`. Amazon EKS no admite la copia de seguridad ni la restauración automatizadas de `etcd` para clústeres locales. Para obtener más información, consulte Backing up an etcd cluster en la documentación

de Kubernetes. Las dos opciones principales son usar `etcdctl` para automatizar la toma de instantáneas o usar la copia de seguridad del volumen de almacenamiento de Amazon EBS.

Información sobre las versiones de la plataforma de Amazon EKS y Kubernetes para AWS Outposts

Las versiones de la plataforma de clúster local representan las capacidades del clúster de Amazon EKS en AWS Outposts. Las versiones incluyen los componentes que se ejecutan en el plano de control de Kubernetes, cuyos indicadores del servidor de la API de Kubernetes están habilitados. También incluyen la versión del parche de Kubernetes actual. Cada versión secundaria de Kubernetes tiene una o varias versiones de la plataforma de asociados. Las versiones de la plataforma para las diferentes versiones secundarias de Kubernetes son independientes. Las versiones de plataforma para los clústeres locales y los clústeres de Amazon EKS en la nube son independientes.

Cuando está disponible una nueva versión secundaria de Kubernetes para clústeres locales, como `1.31`, la versión inicial de la plataforma para esa versión secundaria de Kubernetes comienza con `eks-local-outposts.1`. Sin embargo, Amazon EKS lanza nuevas versiones de la plataforma de forma periódica para habilitar nuevos ajustes de plano de control de Kubernetes y proporcionar revisiones de seguridad.

Cuando están disponibles nuevas versiones de la plataforma de clústeres locales para una versión secundaria:

- El número de versión de la plataforma se incrementa (`eks-local-outposts.n+1`).
- Amazon EKS actualiza automáticamente todos los clústeres locales existentes a la última versión de la plataforma para su versión secundaria de Kubernetes correspondiente. Las actualizaciones automáticas de las versiones de la plataforma existentes se implementan de forma incremental. El proceso de implementación consiste en reemplazar las instancias administradas del plano de control de Kubernetes que se ejecutan en el Outpost, una a la vez, hasta que las tres instancias se sustituyan por otras nuevas.
- El proceso de reemplazo de instancias del plano de control de Kubernetes dejará de avanzar si existe el riesgo de que se interrumpa el servicio. Amazon EKS solo intentará reemplazar una instancia si las otras dos instancias del plano de control de Kubernetes estén en buen estado y cumplen todas las condiciones de preparación como un nodo de clúster.
- Por lo general, el lanzamiento de una versión de la plataforma tarda menos de 30 minutos en completarse. Si el estado de un clúster es `UPDATING` durante un periodo prolongado, consulte

the section called “Solución de problemas de clústeres” y busque ayuda de AWS Support. Nunca cancele manualmente las instancias del plano de control de Kubernetes a menos que AWS Support se lo indique.

- Amazon EKS puede publicar una nueva AMI de nodo con la versión de parche correspondiente. Todas las versiones de parche son compatibles entre el plano de control de Kubernetes y las AMI de nodo para una única versión secundaria de Kubernetes.

Las nuevas versiones de la plataforma no introducen cambios bruscos ni provocan interrupciones de servicio.

Los clústeres locales siempre se crean con la última versión de la plataforma disponible (`eks-local-outposts.n`) para la versión de Kubernetes especificada.

Las versiones de la plataforma actuales y recientes se describen en las siguientes tablas.

Para recibir notificaciones de todos los cambios en el archivo de origen de esta página de documentación específica, puede suscribirse a la siguiente URL con un lector de RSS:

```
https://github.com/awsdocs/amazon-eks-user-guide/commits/mainline/latest/ug/outposts/eks-outposts-platform-versions.adoc.atom
```

Versión de Kubernetes **`1.31`**

Los siguientes controladores de admisión están habilitados para todas las versiones de plataforma `1.31`: `CertificateApproval`, `CertificateSigning`, `CertificateSubjectRestriction`, `ClusterTrustBundleAttest`, `DefaultIngressClass`, `DefaultStorageClass`, `DefaultTolerationSeconds`, `ExtendedResourceToleration`, `LimitRanger`, `MutatingAdmissionWebhook`, `NamespaceLifecycle`, `NodeRestriction`, `PersistentVolumeClaimResize`, `PodSecurity`, `Priority`, `ResourceQuota`, `RuntimeClass`, `ServiceAccount`, `StorageObjectInUseProtection`, `TaintNodesByCondition`, `ValidatingAdmissionPolicy` y `ValidatingAdmissionWebhook`.

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.31.12	eks-local-outposts.5	Nueva versión de la plataforma con mejoras y correcciones de seguridad kube-proxy actualizada a v1.31.10. AWS Autenticador de IAM actualizado a v0.7.4. Se actualizó el complemento CNI de Amazon VPC para Kubernetes a v1.20.2. Se actualizó la versión de Bottlerocket a la v1.47.0.	3 de octubre de 2025
1.31.9	eks-local-outposts.4	Nueva versión de la plataforma con mejoras y correcciones de seguridad kube-proxy actualizada a v1.31.9. AWS Autenticador de IAM actualizado a v0.7.2. Se ha actualizado el complemento CNI de Amazon VPC para Kubernetes a la versión v1.20.0. Se ha actualizado la	9 de agosto de 2025

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
		versión de Bottlerocket a `v1.43.0`.	
`1.31.7`	`eks-local-outposts.3`	Nueva versión de la plataforma con mejoras y correcciones de seguridad kube-proxy actualizada a `v1.31.9`. AWS Autenticador de IAM actualizado a `v0.7.1`. Se actualizó el complemento CNI de Amazon VPC para Kubernetes a `v1.19.5`. Se actualizó la versión de Bottlerocket a la `v1.40.0`.	19 de junio de 2025
`1.31.6`	`eks-local-outposts.2`	Nueva versión de la plataforma con mejoras y correcciones de seguridad. Se actualizó la versión de Bottlerocket a la `v1.36.0`.	24 de abril de 2025

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.31.6	eks-local-outposts.1	Lanzamiento inicial de la versión v1.31 de Kubernetes para los clústeres locales de Amazon EKS en Outposts.	9 de abril de 2025

Versión de Kubernetes **1.30**

Los siguientes controladores de admisión están habilitados para todas las versiones de plataforma 1.30: CertificateApproval, CertificateSigning, CertificateSubjectRestriction, ClusterTrustBundleAttest, DefaultIngressClass, DefaultStorageClass, DefaultTolerationSeconds, ExtendedResourceToleration, LimitRanger, MutatingAdmissionWebhook, NamespaceLifecycle, NodeRestriction, PersistentVolumeClaimResize, PodSecurity, Priority, ResourceQuota, RuntimeClass, ServiceAccount, StorageObjectInUseProtection, TaintNodesByCondition, ValidatingAdmissionPolicy y ValidatingAdmissionWebhook.

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.30.14	eks-local-outposts.7	Nueva versión de la plataforma con mejoras y correcciones de seguridad kube-proxy actualizada a v1.30.14. AWS Autenticador de IAM actualizado a v0.7.4. Se actualizó	3 de octubre de 2025

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
		el complemento CNI de Amazon VPC para Kubernetes a `v1.20.2`. Se actualizó la versión de Bottlerocket a la `v1.47.0`.	
`1.30.13`	`eks-local-outposts.6`	Nueva versión de la plataforma con mejoras y correcciones de seguridad kube-proxy actualizada a `v1.30.13`. AWS Autenticador de IAM actualizado a `v0.7.2`. Se actualizó el complemento CNI de Amazon VPC para Kubernetes a `v1.20.0`. Se actualizó la versión de Bottlerocket a la `v1.43.0`.	09 de agosto de 2025

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.30.11	eks-local-outposts.5	Nueva versión de la plataforma con mejoras y correcciones de seguridad kube-proxy actualizada a v1.30.11. AWS Autenticador de IAM actualizado a v0.7.1. Se ha actualizado el complemento CNI de Amazon VPC para Kubernetes a la versión v1.19.5. Se ha actualizado la versión de Bottlerocket a v1.40.0.	19 de junio de 2025
1.30.10	eks-local-outposts.4	Nueva versión de la plataforma con mejoras y correcciones de seguridad. Se actualizó la versión de Bottlerocket a la v1.36.0.	24 de abril de 2025

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.30.10	eks-local-outposts.3	Nueva versión de la plataforma con mejoras y correcciones de seguridad kube-proxy actualizada a v1.30.10. AWS Autenticador de IAM actualizado a v0.6.29. Se actualizó el complemento CNI de Amazon VPC para Kubernetes a v1.19.2. Se actualizó CoreDNS a v1.11.4. AWS Se actualizó el administrador de controladores en la nube a v1.30.8. Se actualizó la versión de Bottlerocket a la v1.34.0.	27 de marzo de 2025

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.30.7	eks-local-outposts.2	Nueva versión de la plataforma con mejoras y correcciones de seguridad kube-proxy actualizada a v1.30.7. AWS Autenticador de IAM actualizado a v0.6.28. Se actualizó el complemento CNI de Amazon VPC para Kubernetes a v1.19.0. Se actualizó la versión de Bottlerocket a la v1.29.0.	10 de enero de 2025
1.30.5	eks-local-outposts.1	Lanzamiento inicial de la versión v1.30 de Kubernetes para los clústeres locales de Amazon EKS en Outposts.	13 de noviembre de 2024

Versión de Kubernetes **1.29**

Los siguientes controladores de admisión están habilitados para todas las versiones de plataforma 1.29: CertificateApproval, CertificateSigning, CertificateSubjectRestriction, ClusterTrustBundleAttest, DefaultIngressClass, DefaultStorageClass, DefaultTolerationSeconds, ExtendedResourceToleration, LimitRanger, MutatingAdmissionWebhook,

NamespaceLifecycle, NodeRestriction, PersistentVolumeClaimResize, PodSecurity, Priority, ResourceQuota, RuntimeClass, ServiceAccount, StorageObjectInUseProtection, TaintNodesByCondition, ValidatingAdmissionPolicy y ValidatingAdmissionWebhook.

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
v1.29.15	eks-local-outposts.10	Nueva versión de la plataforma con mejoras y correcciones de seguridad. AWS Autenticador de IAM actualizado a v0.7.4. Se actualizó el complemento CNI de Amazon VPC para Kubernetes a v1.20.2. Se actualizó la versión de Bottlerocket a la v1.47.0.	3 de octubre de 2025
v1.29.15	eks-local-outposts.9	Nueva versión de la plataforma con mejoras y correcciones de seguridad. AWS Autenticador de IAM actualizado a v0.7.2. Se actualizó el complemento CNI de Amazon VPC para Kubernetes a v1.20.0. Se actualizó la versión	9 de agosto de 2025

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
		de Bottlerocket a la `v1.43.0`.	
`v1.29.15`	`eks-local-outposts.8`	Nueva versión de la plataforma con mejoras y correcciones de seguridad kube-proxy actualizada a `v1.29.15`. AWS Autenticador de IAM actualizado a `v0.7.1`. Se actualizó el complemento CNI de Amazon VPC para Kubernetes a `v1.19.5`. Se actualizó la versión de Bottlerocket a la `v1.40.0`.	19 de junio de 2025
`v1.29.14`	`eks-local-outposts.7`	Nueva versión de la plataforma con mejoras y correcciones de seguridad. Se actualizó la versión de Bottlerocket a la `v1.36.0`.	24 de marzo de 2025

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
v1.29.14	eks-local-outposts.6	Nueva versión de la plataforma con mejoras y correcciones de seguridad kube-proxy actualizada a v1.29.14. Se actualizó el complemento CNI de Amazon VPC para Kubernetes a v1.19.2. Se actualizó CoreDNS a v1.11.4. AWS Se actualizó el administrador de controladores en la nube a v1.29.8. Se actualizó la versión de Bottlerocket a la v1.34.0.	27 de marzo de 2025

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
v1.29.11	eks-local-outposts.5	Nueva versión de la plataforma con mejoras y correcciones de seguridad kube-proxy actualizada a v1.29.11. Se actualizó el complemento CNI de Amazon VPC para Kubernetes a v1.19.0. Se actualizó la imagen de CoreDNS a v1.11.3. Se actualizó la versión de Bottlerocket a la v1.29.0.	10 de enero de 2025
1.29.9	eks-local-outposts.4	Nueva versión de la plataforma con mejoras y correcciones de seguridad kube-proxy actualizada a v1.29.9. AWS Autenticador de IAM actualizado a v0.6.26. Se actualizó la versión de Bottlerocket a la v1.26.0.	8 de noviembre de 2024

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.29.6	eks-local-outposts.3	Nueva versión de la plataforma con mejoras y correcciones de seguridad. Se actualizó la versión de Bottlerocket a la v1.22.0.	22 de octubre de 2024
1.29.6	eks-local-outposts.2	Nueva versión de la plataforma con mejoras y correcciones de seguridad. Se actualizó la versión de Bottlerocket a la v1.21.0.	27 de agosto de 2024
1.29.6	eks-local-outposts.1	Lanzamiento inicial de la versión v1.29 de Kubernetes para los clústeres locales de Amazon EKS en Outposts.	20 de agosto de 2024

Versión de Kubernetes **1.28**

Los siguientes controladores de admisión están habilitados para todas las versiones de plataforma 1.28: CertificateApproval, CertificateSigning, CertificateSubjectRestriction, ClusterTrustBundleAttest, DefaultIngressClass, DefaultStorageClass, DefaultTolerationSeconds, ExtendedResourceToleration, LimitRanger, MutatingAdmissionWebhook, NamespaceLifecycle, NodeRestriction, PersistentVolumeClaimResize, PodSecurity, Priority, ResourceQuota, RuntimeClass, ServiceAccount,

`StorageObjectInUseProtection`, `TaintNodesByCondition`, `ValidatingAdmissionPolicy` y `ValidatingAdmissionWebhook`.

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
`1.28.15`	`eks-local-outposts.17`	Nueva versión de la plataforma con mejoras y correcciones de seguridad. AWS Autenticador de IAM actualizado a `v0.7.4`. Se actualizó el complemento CNI de Amazon VPC para Kubernetes a `v1.20.2`. Se actualizó la versión de Bottlerocket a la `v1.47.0`.	3 de octubre de 2025
`1.28.15`	`eks-local-outposts.16`	Nueva versión de la plataforma con mejoras y correcciones de seguridad. AWS Autenticador de IAM actualizado a `v0.7.2`. Se actualizó el complemento CNI de Amazon VPC para Kubernetes a `v1.20.0`. Se actualizó la versión de Bottlerocket a la `v1.43.0`.	9 de agosto de 2025

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.28.15	eks-local-outposts.15	Nueva versión de la plataforma con mejoras y correcciones de seguridad. AWS Autenticador de IAM actualizado a v0.7.1. Se actualizó el complemento CNI de Amazon VPC para Kubernetes a v1.19.5. Se actualizó CoreDNS a v1.11.4. Se actualizó la versión de Bottlerocket a la v1.40.0.	19 de junio de 2025
1.28.15	eks-local-outposts.14	Nueva versión de la plataforma con mejoras y correcciones de seguridad. Se actualizó la versión de Bottlerocket a la v1.36.0.	24 de abril de 2025

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.28.15	eks-local-outposts.13	Nueva versión de la plataforma con mejoras y correcciones de seguridad. Se actualizó la compilación kube-proxy a v1.28.15. Se actualizó el complemento CNI de Amazon VPC para Kubernetes a v1.19.0. AWS Se actualizó el administrador de controladores en la nube a v1.28.11.	27 de marzo de 2025
1.28.15	eks-local-outposts.12	Nueva versión de la plataforma con mejoras y correcciones de seguridad kube-proxy actualizada a v1.28.15. Se actualizó el complemento CNI de Amazon VPC para Kubernetes a v1.19.0. Se actualizó la versión de Bottlerocket a la v1.29.0.	10 de enero de 2025

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.28.14	eks-local-outposts.11	Nueva versión de la plataforma con mejoras y correcciones de seguridad kube-proxy actualizada a v1.28.14. AWS Autenticador de IAM actualizado a v0.6.26. Se actualizó la imagen de CoreDNS a v1.11.1. Se actualizó la versión de Bottlerocket a la v1.26.0.	8 de noviembre de 2024
1.28.10	eks-local-outposts.10	Nueva versión de la plataforma con mejoras y correcciones de seguridad. Se actualizó la versión de Bottlerocket a la v1.22.0.	22 de octubre de 2024
1.28.10	eks-local-outposts.9	Nueva versión de la plataforma con mejoras y correcciones de seguridad. Se actualizó la versión de Bottlerocket a la v1.21.0.	27 de agosto de 2024

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.28.10	eks-local-outposts.8	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	30 de julio de 2024
1.28.10	eks-local-outposts.6	Nueva versión de la plataforma con mejoras y correcciones de seguridad kube-proxy actualizada a v1.28.10. AWS Autenticador de IAM actualizado a v0.6.20. Se actualizó la versión de Bottlerocket a la v1.20.2.	19 de junio de 2024
1.28.6	eks-local-outposts.5	Se actualizó la versión de Bottlerocket a la v1.19.3, que incluye las correcciones de errores más recientes para admitir el arranque local en Outposts.	18 de abril de 2024

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.28.6	eks-local-outposts.4	Nueva versión de la plataforma con mejoras y correcciones de seguridad. Soporte restaurado o arranque local en Outposts. Se degradó la versión de Bottlerocket a v1.15.1 por motivos de compatibilidad.	2 de abril de 2024
1.28.6	eks-local-outposts.3	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	22 de marzo de 2024

Versión de Kubernetes	Versión de la plataforma de Amazon EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.28.6	eks-local-outposts.2	Nueva versión de la plataforma con mejoras y correcciones de seguridad kube-proxy actualizada a v1.28.6. AWS Autenticador de IAM actualizado a v0.6.17. Complemento CNI de Amazon VPC para Kubernetes adaptado a la versión v1.13.2 por motivos de compatibilidad. Se actualizó la versión de Bottlerocket a la v1.19.2.	8 de marzo de 2024
1.28.1	eks-local-outposts.1	Lanzamiento inicial de la versión v1.28 de Kubernetes para los clústeres locales de Amazon EKS en Outposts.	4 de octubre de 2023

Creación de una VPC y subredes para clústeres de Amazon EKS en AWS Outposts

Al crear un clúster local, especifica una VPC y al menos una subred privada que se ejecuta en Outposts. En este tema, se proporciona una descripción general de los requisitos y las consideraciones de la VPC y las subredes para el clúster local.

Requisitos y consideraciones de la VPC

Al crear un clúster local, la VPC que especifique debe cumplir los siguientes requisitos y consideraciones:

- Asegúrese de que la VPC tenga suficientes direcciones IP para el clúster local, los nodos y otros recursos de Kubernetes que desee crear. Si la VPC que desea utilizar no tiene suficientes direcciones IP, aumente el número de direcciones IP disponibles. Puede hacerlo asociando bloques adicionales de enrutamiento entre dominios sin clase (CIDR) con su VPC. Puede asociar bloques de CIDR privados (RFC 1918) y públicos (no RFC 1918) a su VPC antes o después de crear el clúster. Un clúster puede tardar hasta cinco horas en reconocer un bloque de CIDR asociado a una VPC.
- La VPC no puede tener prefijos IP ni bloques de CIDR IPv6 asignados. Debido a estas restricciones, la información incluida en Asignación de más direcciones IP a los nodos de Amazon EKS con prefijos y the section called “IPv6” no aplica a su VPC.
- La VPC tiene un nombre de host de DNS y la resolución de DNS habilitados. Sin estas características, se produce un error en la creación del clúster local y tendrá que habilitarlas y volver a crear el clúster. A fin de obtener más información, consulte Atributos de DNS para su VPC en la Guía del usuario de Amazon VPC.
- Para acceder al clúster local a través de la red local, la VPC debe estar asociada a la tabla de enrutamiento de la puerta de enlace local del Outpost. Para obtener más información, consulte Asociaciones de la VPC en la Guía del usuario de AWS Outposts.

Requisitos y consideraciones de la subred

Debe especificar al menos una subred privada al crear el clúster. Si especifica más de una subred, las instancias del plano de control de Kubernetes se distribuyen de manera uniforme en las subredes. Si se especifica más de una subred, las subredes deben existir en el mismo Outpost. Además, las subredes también deben tener las rutas y los permisos de grupos de seguridad adecuados para comunicarse entre sí. Las subredes que especifique al crear un clúster local deben cumplir los siguientes requisitos:

- Todas las subredes están en el mismo Outpost lógico.
- En conjunto, las subredes tienen al menos tres direcciones IP disponibles para las instancias del plano de control de Kubernetes. Si se especifican tres subredes, cada subred debe tener al menos una dirección IP disponible. Si se especifican dos subredes, cada subred debe tener al menos

dos direcciones IP disponibles. Si se especifica una subred, la subred debe tener al menos tres direcciones IP disponibles.

- Las subredes tienen una ruta a la puerta de enlace local del bastidor del Outpost para acceder al servidor de la API de Kubernetes a través de su red local. Si las subredes no tienen una ruta a la puerta de enlace local del bastidor del Outpost, debe comunicarse con su servidor de la API de Kubernetes desde dentro de la VPC.
- Las subredes deben utilizar nombres basados en direcciones IP. La nomenclatura basada en recursos de Amazon EC2 no es compatible con Amazon EKS.

Acceso a los servicios de AWS de la subred

Las subredes privadas del clúster local en Outposts deben poder comunicarse con los servicios de AWS regionales. Para ello, puede utilizar una puerta de enlace NAT para el acceso saliente a Internet o, si desea mantener la privacidad de todo el tráfico dentro de su VPC, mediante los puntos de conexión de la interfaz de la VPC.

Uso de una puerta de enlace NAT

Las subredes privadas del clúster local en Outposts deben tener una tabla de enrutamiento asociada con una ruta a una puerta de enlace NAT en una subred pública en la zona de disponibilidad principal del Outpost. La subred pública debe tener una ruta hacia una puerta de enlace de Internet. La puerta de enlace NAT permite el acceso a Internet saliente y evita las conexiones entrantes no solicitadas desde Internet hacia las instancias del Outpost.

Uso de los puntos de conexión de VPC de interfaz

Si las subredes privadas del clúster local en Outposts no tienen una conexión a Internet de salida o si desea mantener todo el tráfico privado dentro de su VPC, debe crear los siguientes puntos de conexión de VPC de interfaz y un punto de conexión de puerta de enlace en una subred regional antes de crear el clúster.

Punto de conexión	Tipo de punto de conexión
`com.amazonaws.` *`region-code`* `.ssm`	Interfaz
`com.amazonaws.` *`region-code`* `.ssmmessages`	Interfaz

Punto de conexión	Tipo de punto de conexión
com.amazonaws. *region-code* .ec2messages	Interfaz
com.amazonaws. *region-code* .ec2	Interfaz
com.amazonaws. *region-code* .secretsmanager	Interfaz
com.amazonaws. *region-code* .logs	Interfaz
com.amazonaws. *region-code* .sts	Interfaz
com.amazonaws. *region-code* .ecr.api	Interfaz
com.amazonaws. *region-code* .ecr.dkr	Interfaz
com.amazonaws. *region-code* .s3	Puerta de enlace

Los puntos de conexión debe cumplir los siguientes requisitos:

- Deben crearse en una subred privada ubicada en la zona de disponibilidad principal de su Outpost
- Deben tener habilitados los nombres de DNS privados
- Deben tener un grupo de seguridad adjunto que permita el tráfico HTTPS entrante desde el rango CIDR de la subred de Outpost privada.

La creación de puntos de conexión incurre en costos. Para obtener más información, consulte Precios de AWS PrivateLink. Si sus pods necesitan acceso a otros servicios de AWS, debe crear puntos de conexión adicionales. Para obtener una lista completa de puntos de conexión, consulte Servicios de AWS que se integran con AWS PrivateLink.

Creación de una VPC

Puede crear una VPC que cumpla los requisitos anteriores mediante una de las siguientes plantillas de AWS CloudFormation:

- Plantilla 1: esta plantilla crea una VPC con una subred privada en el Outpost y una subred pública en la región de AWS. La subred privada tiene una ruta a Internet a través de una puerta de enlace

NAT que reside en la subred pública de la región de AWS. Esta plantilla se puede usar para crear un clúster local en una subred con acceso a Internet de salida.

- Plantilla 2: esta plantilla crea una VPC con una subred privada en el Outpost y el conjunto mínimo de puntos de conexión de VPC necesarios para crear un clúster local en una subred que no tenga acceso a Internet de entrada o salida (también denominada subred privada).

Preparación de los clústeres locales de Amazon EKS en AWS Outposts para las desconexiones de la red

Puede seguir usando su clúster local de Amazon EKS en un Outpost si su red local ha perdido la conectividad con la nube de AWS. En este tema, se explica cómo preparar el clúster local para las desconexiones de red y las consideraciones relacionadas.

- Los clústeres locales permiten la estabilidad y las operaciones continuas durante las desconexiones de red temporales e imprevistas. AWS Outposts sigue siendo una oferta completamente conectada que funciona como una extensión de la nube de AWS en su centro de datos. En caso de que la red se desconecte entre su Outpost y la nube de AWS, le recomendamos que intente restablecer la conexión. Para obtener instrucciones, consulte lista de verificación para la solución de problemas de red de los bastidores de AWS en la Guía del usuario de AWS Outposts. Para obtener más información acerca de cómo solucionar problemas de los clústeres locales, consulte the section called “Solución de problemas de clústeres”.
- Los Outposts emiten una métrica `ConnectedStatus` que puede usar para supervisar el estado de conectividad de su Outpost. Para obtener más información, consulte Métricas de Outposts en la Guía del usuario de AWS Outposts.
- Los clústeres locales utilizan IAM como mecanismo de autenticación predeterminado mediante el Autenticador de AWS Identity and Access Management para Kubernetes. IAM no está disponible durante las desconexiones de red. Entonces, los clústeres locales admiten un mecanismo de autenticación alternativo mediante certificados `x.509` que puede usar para conectarse a su clúster durante las desconexiones de red. Para obtener información acerca de cómo obtener y usar un certificado `x.509` para su clúster, consulte the section called “Autenticación en el clúster local durante una desconexión de red”.
- Si no puede acceder a Route 53 durante las desconexiones de red, considere la posibilidad de utilizar servidores de DNS locales en su entorno en las instalaciones. Las instancias del plano de control de Kubernetes usan direcciones IP estáticas. Puede configurar los hosts que usa para conectarse a su clúster con el nombre de host y las direcciones IP del punto de conexión como

alternativa al uso de servidores de DNS locales. Para obtener más información, consulte DNS en la Guía del usuario de AWS Outposts.

- Si prevé aumentos en el tráfico de aplicaciones durante las desconexiones de red, puede aprovisionar capacidad de computación sobrante en su clúster cuando se conecte a la nube. Las instancias de Amazon EC2 están incluidas en el precio de AWS Outposts. Por lo tanto, la ejecución de instancias de repuesto no afecta al costo de uso de AWS.
- Durante las desconexiones de red, para permitir las operaciones de creación, actualización y escalado de las cargas de trabajo, las imágenes del contenedor de la aplicación deben ser accesibles a través de la red local y el clúster debe tener capacidad suficiente. Los clústeres locales no alojan un registro de contenedores en su nombre. Las imágenes del contenedor se almacenan en caché en los nodos si los pods se ejecutaron anteriormente en esos nodos. Si normalmente extrae las imágenes del contenedor de su aplicación de Amazon ECR en la nube, considere la posibilidad de ejecutar una caché o un registro local. Una caché o un registro local es útil si necesita crear, actualizar y escalar operaciones para los recursos de las cargas de trabajo durante desconexiones de red.
- Los clústeres locales usan Amazon EBS como clase de almacenamiento predeterminada para los volúmenes persistentes y el controlador de CSI de Amazon EBS para administrar el ciclo de vida de los volúmenes persistentes de Amazon EBS. Durante las desconexiones de red, los pods basados en Amazon EBS no se pueden crear, actualizar ni escalar. Esto se debe a que estas operaciones requieren llamadas a la API de Amazon EBS en la nube. Si está implementando cargas de trabajo con estado en clústeres locales y necesita crear, actualizar o escalar operaciones durante las desconexiones de red, considere la posibilidad de utilizar un mecanismo de almacenamiento alternativo.
- Las instantáneas de Amazon EBS no se pueden crear ni eliminar si AWS Outposts no puede acceder a las API pertinentes en la región de AWS (como las API de Amazon EBS o Amazon S3).
- Al integrar ALB (Ingress) con AWS Certificate Manager (ACM), los certificados se cargan y almacenan en la memoria de la instancia ALB Compute de AWS Outposts. La terminación actual de TLS seguirá funcionando en caso de que se desconecte de la región de AWS. Las operaciones de mutación en este contexto fallarán (como las nuevas definiciones de entrada, las nuevas operaciones de API de certificados basadas en ACM, la escala de computación de ALB o la rotación de certificados). Para obtener más información, consulte Solución de problemas de renovación administrada de certificados en la Guía del usuario de AWS.
- Los registros del plano de control de Amazon EKS se almacenan en caché de forma local en instancias del plano de control de Kubernetes durante las desconexiones de red. Si se conecta nuevamente, los registros se enviarán a Registros de CloudWatch en la región de AWS principal.

Puede utilizar Prometheus, Grafana o las soluciones de socios de Amazon EKS para supervisar el clúster de forma local mediante el punto de conexión de métricas del servidor de la API de Kubernetes o utilizar Fluent Bit para registros.

- Si está usando el controlador del equilibrador de carga de AWS en Outposts para el tráfico de aplicaciones, los pods existentes encabezados por el controlador del equilibrador de carga de AWS seguirán recibiendo tráfico durante las desconexiones de red. Los nuevos pods creados durante las desconexiones de red no reciben tráfico hasta que el Outpost se vuelve a conectar a la nube de AWS. Considere la posibilidad de configurar el recuento de réplicas de sus aplicaciones mientras esté conectado a la nube de AWS para satisfacer sus necesidades de escalado durante las desconexiones de red.
- El complemento CNI de Amazon VPC para Kubernetes establece el valor predeterminado en el modo de IP secundario. Está configurado con `WARM_ENI_TARGET=1`, que permite al complemento mantener "una interfaz de red elástica completa" de las direcciones IP disponibles. Considere cambiar los valores `WARM_ENI_TARGET`, `WARM_IP_TARGET` y `MINIMUM_IP_TARGET` de acuerdo con sus necesidades de escalado durante un estado desconectado. Para obtener más información, consulte el archivo readme para el complemento en GitHub. Para obtener una lista del número máximo de pods admitidos por cada tipo de instancia, consulte el archivo eni-max-pods.txt en GitHub.

Autenticación en el clúster local durante una desconexión de red

AWS Identity and Access Management (IAM) no está disponible durante las desconexiones de red. No es posible autenticar en su clúster local con las credenciales de IAM mientras esté desconectado. Sin embargo, puede conectarse a su clúster a través de su red local mediante certificados `x509` cuando esté desconectado. Debe descargar y almacenar un certificado cliente `X509` para usar durante las desconexiones. En este tema, aprenderá a crear y utilizar el certificado para autenticarse en su clúster cuando está en estado desconectado.

1. Crear una solicitud de firma de certificado.
   a. Genere una solicitud de firma de certificado.

```
openssl req -new -newkey rsa:4096 -nodes -days 365 \
    -keyout admin.key -out admin.csr -subj "/CN=admin"
```

   b. Cree una solicitud de firma de certificado en 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. Cree una solicitud de firma de certificado con `kubectl`.

```
kubectl create -f admin-csr.yaml
```

3. Compruebe el estado de la solicitud de firma de certificado.

```
kubectl get csr admin-csr
```

   Un ejemplo de salida sería el siguiente.

```
NAME       AGE   REQUESTOR                      CONDITION
admin-csr  11m   kubernetes-admin               Pending
```

   Kubernetes ha creado la solicitud de firma de certificado.

4. Apruebe la solicitud de firma de certificado.

```
kubectl certificate approve admin-csr
```

5. Vuelva a comprobar el estado de la solicitud de firma de certificado para aprobarla.

```
kubectl get csr admin-csr
```

   Un ejemplo de salida sería el siguiente.

```
NAME       AGE   REQUESTOR                     CONDITION
admin-csr  11m   kubernetes-admin              Approved
```

6. Recupere y verifique el certificado.

a. Recupere el certificado.

```
kubectl get csr admin-csr -o jsonpath='{.status.certificate}' | base64 --decode >
 admin.crt
```

b. Verifique el certificado.

```
cat admin.crt
```

7. Cree un enlace de roles de clúster para un usuario `admin`.

```
kubectl create clusterrolebinding admin --clusterrole=cluster-admin \
    --user=admin --group=system:masters
```

8. Genere un kubeconfig con ámbito de usuario para un estado desconectado.

   Puede generar un archivo `kubeconfig` con los certificados de `admin` descargados. Reemplace *`my-cluster`* y *`apiserver-endpoint`* en los siguientes comandos.

```
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. Visualización del archivo `kubeconfig`.

```
kubectl get nodes --kubeconfig admin.kubeconfig
```

10. Si ya tiene servicios en producción en su Outpost, omita este paso. Si Amazon EKS es el único servicio que se ejecuta en su Outpost y el Outpost no está en producción actualmente, puede simular una desconexión de red. Antes de iniciar la producción con el clúster local, simule una desconexión para asegurarse de que puede acceder al clúster cuando esté en estado desconectado.

    a. Aplique las reglas de firewall en los dispositivos de red que conectan su Outpost a la región de AWS. Esto desconecta el enlace de servicio del Outpost. No puede crear ninguna instancia nueva. Las instancias que se están ejecutando actualmente pierden la conectividad con la región de AWS e Internet.

    b. Puede probar la conexión a su clúster local mientras está desconectado mediante el certificado `x509`. Asegúrese de cambiar su `kubeconfig` a la `admin.kubeconfig` que creó en un paso anterior. Reemplace *`my-cluster`* por el nombre de su clúster local.

    ```
    kubectl config use-context admin@my-cluster --kubeconfig admin.kubeconfig
    ```

    Si observa algún problema con sus clústeres locales mientras están desconectados, le recomendamos que abra un tique de soporte.

Seleccione los tipos de instancias y los grupos de ubicación para los clústeres de Amazon EKS en AWS en función de las consideraciones de capacidad

En este tema, se proporciona orientación para seleccionar el tipo de instancia del plano de control de Kubernetes y (opcionalmente) para usar grupos de ubicación para cumplir con los requisitos de alta disponibilidad del clúster local de Amazon EKS en un Outpost.

Antes de seleccionar un tipo de instancia (como `m5`, `c5` o `r5`) para usarlo en el plano de control de Kubernetes de su clúster local en Outposts, confirme los tipos de instancias disponibles en su configuración de Outpost. Una vez que haya identificado los tipos de instancias disponibles, seleccione el tamaño de la instancia (como `large`, `xlarge` o `2xlarge`) en función de la cantidad de nodos que requieren sus cargas de trabajo. La siguiente tabla proporciona recomendaciones para elegir el tamaño de una instancia.

> **Note**
>
> Los tamaños de las instancias deben estar establecidos en sus Outposts. Asegúrese de tener capacidad suficiente para tres instancias del tamaño disponible en sus Outposts durante la vida útil de su clúster local. Para obtener una lista de los tipos de instancias de Amazon EC2 disponibles, consulte las secciones de computación y almacenamiento en las características de los bastidores de AWS Outposts.

Número de nodos	Tamaño de instancia del plano de control de Kubernetes
1–20	`large`
21–100	`xlarge`
101–250	`2xlarge`
251–500	`4xlarge`

El almacenamiento para el plano de control de Kubernetes requiere 246 GB de almacenamiento de Amazon EBS por cada clúster local para cumplir con la IOPS requerida para `etcd`. Los volúmenes de Amazon EBS se aprovisionan automáticamente cuando se crea el clúster local.

Ubicación del plano de control

Si no especifica un grupo de ubicación en la propiedad `OutpostConfig.ControlPlanePlacement.GroupName`, las instancias de Amazon EC2 aprovisionadas para su plano de control de Kubernetes no reciben ninguna obligación específica de ubicación de hardware en toda la capacidad subyacente disponible en su Outpost.

Puede usar grupos de ubicación para cumplir con los requisitos de alta disponibilidad de su clúster local de Amazon EKS en un Outpost. Al especificar un grupo de ubicaciones durante la creación del clúster, se influye en la ubicación de las instancias del plano de control de Kubernetes. Las instancias se distribuyen en un hardware subyacente independiente (bastidores o hosts), lo que minimiza el impacto correlacionado de las instancias en caso de errores de hardware.

El tipo de distribución que puede configurar depende de la cantidad de bastidores de Outpost que tenga en su implementación.

- Implementaciones con uno o dos bastidores físicos en un único Outpost lógico: debe tener al menos tres hosts configurados con el tipo de instancia que elija para las instancias del plano de control de Kubernetes. Un grupo con ubicación distribuida que utilice la distribución por host garantiza que todas las instancias del plano de control de Kubernetes se ejecuten en distintos hosts dentro de los bastidores subyacentes disponibles en su implementación de Outpost.
- Implementaciones con tres o más bastidores físicos en un único Outpost lógico: debe tener al menos tres hosts configurados con el tipo de instancia que elija para las instancias del plano de control de Kubernetes. Un grupo con ubicación distribuida que utilice la distribución por bastidor garantiza que todas las instancias del plano de control de Kubernetes se ejecutan en distintos bastidores en su implementación de Outpost. También puede utilizar el grupo de ubicación de dispersión a nivel de host de servidor tal y como se describe en la opción anterior.

Usted es responsable de crear el grupo de ubicación deseado. Especifique el grupo de ubicación al llamar a la API de `CreateCluster`. Para más información sobre los grupos de colocación y cómo crearlos, consulte Grupos de ubicación en la Guía del usuario de Amazon EC2.

- Cuando se especifica un grupo de ubicación, debe haber capacidad distribuida disponible en su Outpost para crear correctamente un clúster local de Amazon EKS. La capacidad varía en función de si se utiliza el tipo de distribución de host o de bastidores. Si no hay suficiente capacidad, el clúster permanece en el estado de `Creating`. Puede comprobar el `Insufficient Capacity Error` en el campo de estado de la respuesta de la API de DescribeCluster. Debe liberar capacidad para que el proceso de creación progrese.
- Durante las actualizaciones de la plataforma y la versión del clúster local de Amazon EKS, las instancias del plano de control de Kubernetes del clúster se sustituyen por nuevas instancias mediante una estrategia de actualización continua. Durante este proceso de reemplazo, se termina cada instancia del plano de control, lo que libera su ranura respectiva. Se aprovisiona una nueva instancia actualizada en su lugar. Es posible que la instancia actualizada se coloque en la ranura que se publicó. Si la ranura la consume otra instancia no relacionada y no queda más capacidad que respete el requisito de topología de dispersión requerido, el clúster permanece en el estado de `Updating`. Puede comprobar el `Insufficient Capacity Error` correspondiente en el campo de estado de la respuesta de la API de DescribeCluster. Debe liberar capacidad para que el proceso de actualización pueda avanzar y restablecer los niveles de alta disponibilidad anteriores.

- Puede crear un máximo de 500 grupos de ubicación por cuenta en cada región de AWS. Para obtener más información, consulte Normas generales y limitaciones en la Guía del usuario de Amazon EC2.

Solución de problemas de los clústeres locales de Amazon EKS en AWS Outposts

En este tema, se tratan algunos errores habituales que pueden aparecer al usar clústeres locales y cómo solucionarlos. Si bien los clústeres locales son similares a los clústeres de Amazon EKS en la nube, existen algunas diferencias en la forma en que Amazon EKS los administra.

> **Important**
>
> Nunca termine ninguna instancia del plano de control de `Kubernetes` del clúster local de EKS administrada que se ejecute en Outpost a menos que AWS Support lo indique explícitamente. La terminación de estas instancias supone un riesgo para la disponibilidad del servicio del clúster local, incluida la pérdida del clúster local en caso de que se terminen varias instancias simultáneamente. Las instancias del plano de control de `Kubernetes` del clúster local de EKS se identifican mediante la etiqueta `eks-local:controlplane-name` de la consola de instancias de EC2.

Comportamiento de la API

Los clústeres locales se crean mediante la API de Amazon EKS, pero se ejecutan de forma asincrónica. Esto significa que las solicitudes a la API de Amazon EKS se devuelven inmediatamente para los clústeres locales. Sin embargo, estas solicitudes pueden tener éxito, responder rápido a los errores debido a errores de validación de entradas o fallar y tener errores de validación descriptivos. Este comportamiento es similar a la API de Kubernetes.

Los clústeres locales no pasan a un estado `FAILED`. Amazon EKS intenta conciliar el estado del clúster con el estado deseado solicitado por el usuario de forma continua. Como resultado, un clúster local puede permanecer en el estado `CREATING` durante un periodo prolongado hasta que se resuelva el problema subyacente.

Describir el campo de estado del clúster

Los problemas del clúster local se pueden descubrir utilizando el comando describe-cluster de la CLI de AWS para Amazon EKS. Los problemas de los clústeres locales aparecen en el campo `cluster.health` de la respuesta del comando `describe-cluster`. El mensaje contenido en este campo incluye un código de error, un mensaje descriptivo y los ID de los recursos relacionados. Esta información se encuentra disponible solo a través de la API y la CLI de AWS para Amazon EKS. En el siguiente ejemplo, reemplace *my-cluster* por el nombre de su clúster local.

```
aws eks describe-cluster --name my-cluster --query 'cluster.health'
```

Un ejemplo de salida sería el siguiente.

```
{
    "issues": [
        {
            "code": "ConfigurationConflict",
            "message": "The instance type 'm5.large' is not supported in Outpost 'my-
outpost-arn'.",
            "resourceIds": [
                "my-cluster-arn"
            ]
        }
    ]
}
```

Si el problema no se puede solucionar, es posible que tenga que eliminar el clúster local y crear uno nuevo. Por ejemplo, intentar aprovisionar un clúster con un tipo de instancia que no está disponible en su Outpost. La siguiente tabla incluye errores comunes relacionados con el estado.

Situación de error	Código	Mensaje	ResourceIds
No se encontraron las subredes proporcionadas.	`ResourceNotFound`	`The subnet ID` *subnet-id* `does not exist`	Todos los ID de subred proporcionados
Las subredes proporcionadas	`ConfigurationConflict`	`Subnets specified must`	Todos los ID de subred proporcionados

Situación de error	Código	Mensaje	ResourceIds
no pertenecen a la misma VPC.		`belong to the same VPC`	
Algunas subredes proporcionadas no pertenecen al Outpost especificado.	`ConfigurationConflict`	`Subnet subnet-id expected to be in outpost-arn , but is in other-outpost-arn`	ID de subred problemático
Algunas subredes proporcionadas no pertenecen a ningún Outpost.	`ConfigurationConflict`	`Subnet subnet-id is not part of any Outpost`	ID de subred problemático
Algunas subredes proporcionadas no tienen suficientes direcciones libres para crear interfaces de red elásticas para instancias del plano de control.	`ResourceLimitExceeded`	`The specified subnet does not have enough free addresses to satisfy the request.`	ID de subred problemático
El tipo de instancia del plano de control especificado no es compatible con su Outpost.	`ConfigurationConflict`	`The instance type type is not supported in Outpost outpost-arn`	ARN del clúster

Situación de error	Código	Mensaje	ResourceIds
Terminó una instancia de Amazon EC2 del plano de control o `run-instance` se ejecutó correctamente, pero el estado observó cambios en `Terminated`. Esto puede suceder durante un periodo después de que el Outpost se vuelva a conectar y los errores internos de Amazon EBS provoquen una falla en el flujo de trabajo interno de Amazon EC2.	`InternalFailure`	`EC2 instance state "Terminated" is unexpected`	ARN del clúster
La capacidad de su Outpost es insuficiente. Esto también puede ocurrir durante la creación del clúster si un Outpost está desconectado de la región de AWS.	`ResourceLimitExceeded`	`There is not enough capacity on the Outpost to launch or start the instance.`	ARN del clúster
Su cuenta superó la cuota del grupo de seguridad.	`ResourceLimitExceeded`	Mensaje de error devuelto por la API de Amazon EC2	ID de la VPC de destino
Su cuenta superó la cuota de la interfaz de red elástica.	`ResourceLimitExceeded`	Mensaje de error devuelto por la API de Amazon EC2	ID de subred de destino

Situación de error	Código	Mensaje	ResourceIds
No se pudo acceder a las instancias del plano de control a través de AWS Systems Manager. Para obtener una resolución, consulte the section called “No se puede acceder a las instancias del plano de control a través de AWS Systems Manager”.	`ClusterUnreachable`	No se puede acceder a las instancias del plano de control de Amazon EKS mediante SSM. Verifique su SSM y la configuración de red y consulte la documentación de solución de problemas de EKS en Outposts.	ID de instancia de Amazon EC2
Se produjo un error al obtener detalles de un grupo de seguridad administrado o una interfaz de red elástica.	Basado en el código de error del cliente Amazon EC2.	Mensaje de error devuelto por la API de Amazon EC2	Todos los ID de grupo de seguridad administrados
Se produjo un error al autorizar o revocar las reglas de entrada de los grupos de seguridad. Esto se aplica a los grupos de seguridad del clúster y del plano de control.	Basado en el código de error del cliente Amazon EC2.	Mensaje de error devuelto por la API de Amazon EC2	ID de grupo de seguridad problemático
Se produjo un error al eliminar una interfaz de red elástica para una instancia del plano de control	Basado en el código de error del cliente Amazon EC2.	Mensaje de error devuelto por la API de Amazon EC2	ID de interfaz de red elástica problemático

En la siguiente tabla, se enumeran los errores de otros servicios de AWS que se presentan en el campo de estado de la respuesta `describe-cluster`.

Código de error de Amazon EC2	Código del problema del estado del clúster	Descripción
`AuthFailure`	`AccessDenied`	Este error se puede producir por diversas razones. La razón más común es cuando una etiqueta que el servicio usa para determinar el alcance de la política de roles vinculados al servicio se elimina accidentalmente de la instancia del plano de control. Si esto ocurre, Amazon EKS ya no podrá administrar ni supervisar estos recursos de AWS.
`UnauthorizedOperation`	`AccessDenied`	Este error se puede producir por diversas razones. La razón más común es cuando una etiqueta que el servicio usa para determinar el alcance de la política de roles vinculados al servicio se elimina accidentalmente de la instancia del plano de control. Si esto ocurre, Amazon EKS ya no podrá administrar ni supervisar estos recursos de AWS.
`InvalidSubnetID.NotFound`	`ResourceNotFound`	Este error se produce cuando no se encuentra el ID de subred de las reglas de

Código de error de Amazon EC2	Código del problema del estado del clúster	Descripción
		entrada de un grupo de seguridad.
`InvalidPermission.NotFound`	`ResourceNotFound`	Este error se produce cuando los permisos de las reglas de entrada de un grupo de seguridad no son correctos.
`InvalidGroup.NotFound`	`ResourceNotFound`	Este error se produce cuando no se encuentra el grupo de reglas de entrada de un grupo de seguridad.
`InvalidNetworkInterfaceID.NotFound`	`ResourceNotFound`	Este error se produce cuando no se encuentra el ID de la interfaz de red para las reglas de entrada de un grupo de seguridad.
`InsufficientFreeAddressesInSubnet`	`ResourceLimitExceeded`	Este error se produce cuando se supera la cuota de recursos de subred.
`InsufficientCapacityOnOutpost`	`ResourceLimitExceeded`	Este error se produce cuando se supera la cuota de capacidad del outpost.
`NetworkInterfaceLimitExceeded`	`ResourceLimitExceeded`	Este error se produce cuando se supera la cuota de la interfaz de red elástica.
`SecurityGroupLimitExceeded`	`ResourceLimitExceeded`	Este error se produce cuando se supera la cuota del grupo de seguridad.

Código de error de Amazon EC2	Código del problema del estado del clúster	Descripción
`VcpuLimitExceeded`	`ResourceLimitExceeded`	Esto se observa al crear una instancia de Amazon EC2 en una cuenta nueva. El error podría ser similar al siguiente: “: `"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 devuelve este código de error si el tipo de instancia especificado no es compatible con Outpost.
Todas las demás fallas	`InternalFailure`	Ninguno

Incapaz de crear o modificar clústeres

Los clústeres locales requieren permisos y políticas diferentes a los de los clústeres de Amazon EKS alojados en la nube. Cuando un clúster no se puede crear y produce un error de `InvalidPermissions`, compruebe que el rol de clúster que está utilizando tenga la política administrada AmazonEKSLocalOutpostClusterPolicy adjunta. Todas las demás llamadas a la API requieren el mismo conjunto de permisos que los clústeres de Amazon EKS en la nube.

El clúster está atascado en el estado **`CREATING`**

La cantidad de tiempo que tarda en crearse un clúster local varía según varios factores. Estos factores incluyen la configuración de la red, la configuración de Outpost y la configuración del clúster. En general, se crea un clúster local y cambia al estado `ACTIVE` en un plazo de 15 a 20 minutos. Si un clúster local permanece en el estado `CREATING`, puede llamar a `describe-cluster` para solicitar información sobre la causa en el campo de resultado `cluster.health`.

Los problemas más comunes son los siguientes:

- El clúster no puede conectarse a la instancia del plano de control desde la región de AWS en la que se encuentra Systems Manager. Para verificarlo, llame a `aws ssm start-session --target` *`instance-id`* desde un host bastión dentro de la región. Si ese comando no funciona, compruebe si Systems Manager se está ejecutando en la instancia del plano de control. Otra solución alternativa es eliminar el clúster y, a continuación, volver a crearlo.
- Las instancias del plano de control no se pueden crear debido a los permisos de claves de KMS para los volúmenes de EBS. Con las claves de KMS administradas por el usuario para los volúmenes de EBS cifrados, las instancias del plano de control se terminarán si no se puede acceder a la clave. Si las instancias se terminan, cambie a una clave de KMS administrada de AWS o asegúrese de que su política de claves administradas por el usuario conceda los permisos necesarios para el rol de clúster.
- Es posible que las instancias del plano de control de Systems Manager no tengan acceso a Internet. Compruebe si la subred que proporcionó al crear el clúster tiene una puerta de enlace NAT y una VPC con puerta de enlace de Internet. Use el analizador de accesibilidad de la VPC para verificar que la instancia del plano de control puede llegar a la puerta de enlace de Internet. Para obtener más información, consulte Introducción al Analizador de accesibilidad de la VPC.
- Faltan políticas en el ARN de rol que ha proporcionado. Verifique si se eliminó la política administrada de AWS AmazonEKSLocalOutpostClusterPolicy del rol. Esto también puede ocurrir si una pila de AWS CloudFormation está mal configurada.
- Todas las subredes proporcionadas deben estar asociadas al mismo Outpost y poder comunicarse entre sí. Cuando se especifican varias subredes al crear un clúster, Amazon EKS intenta distribuir las instancias del plano de control en varias subredes.
- Los grupos de seguridad administrados de Amazon EKS se aplican en la interfaz de red elástica. Sin embargo, otros elementos de configuración, como las reglas del firewall NACL, pueden entrar en conflicto con las reglas de la interfaz de red elástica.

Falta o está mal configurada la configuración de DNS de la VPC y de la subred

Revise Creación de una VPC y subredes para clústeres de Amazon EKS en AWS Outposts.

El clúster está atascado en el estado **`UPDATING`**

Amazon EKS actualiza automáticamente todos los clústeres locales existentes a las últimas versiones de la plataforma para su versión secundaria de Kubernetes correspondiente. Para obtener más información sobre las versiones de la plataforma, consulte the section called “Versiones de la plataforma de EKS”.

Durante la implementación de la versión de la plataforma automática, el estado del clúster cambia a `UPDATING`. El proceso de actualización consiste en sustituir todas las instancias del plano de control de Kubernetes por otras nuevas que contengan los últimos parches de seguridad y correcciones de errores publicados para la versión secundaria de Kubernetes correspondiente. En general, el proceso de actualización de la plataforma de un clúster local se completa en menos de 30 minutos y el clúster vuelve al estado `ACTIVE`. Si un clúster local permanece en el estado `UPDATING` durante un periodo de tiempo extenso, puede llamar a `describe-cluster` para consultar información sobre la causa en el campo de resultado `cluster.health`.

Amazon EKS garantiza que al menos 2 de cada 3 instancias del plano de control de Kubernetes sean nodos de clúster en buen estado y operativos para mantener la disponibilidad del clúster local y evitar la interrupción del servicio. Si un clúster local se bloquea en el estado `UPDATING`, suele ser porque hay algún problema de infraestructura o configuración que impide garantizar la disponibilidad mínima de las dos instancias en caso de que el proceso continúe. Por lo tanto, el proceso de actualización deja de avanzar para proteger la interrupción del servicio del clúster local.

Es importante solucionar los problemas de un clúster local atascado en el estado `UPDATING` y abordar la causa principal para que el proceso de actualización pueda completarse y restaurar el clúster local a `ACTIVE` con la alta disponibilidad de las tres instancias del plano de control de Kubernetes.

No termine ninguna instancia de `Kubernetes` del clúster local de EKS administrada que se ejecute en Outposts a menos que AWS Support lo indique explícitamente. Esto es especialmente importante en el caso de los clústeres locales atascados en el estado `UPDATING`, ya que existe una alta probabilidad de que otro nodo del plano de control no esté completamente en buen estado y, si se termina una instancia incorrecta, se podría interrumpir el servicio y correr el riesgo de perder los datos del clúster local.

Los problemas más comunes son los siguientes:

- Una o más instancias del plano de control no pueden conectarse a Systems Manager debido a un cambio en la configuración de la red desde que se creó el clúster local por primera vez. Para verificarlo, llame a `aws ssm start-session --target instance-id` desde un host bastión dentro de la región. Si ese comando no funciona, compruebe si Systems Manager se está ejecutando en la instancia del plano de control.
- Las instancias del nuevo plano de control no se pueden crear debido a los permisos de claves de KMS para los volúmenes de EBS. Con las claves de KMS administradas por el usuario para los volúmenes de EBS cifrados, las instancias del plano de control se terminarán si no se puede acceder a la clave. Si las instancias se terminan, cambie a una clave de KMS administrada de AWS o asegúrese de que su política de claves administradas por el usuario conceda los permisos necesarios para el rol de clúster.
- Es posible que las instancias del plano de control de Systems Manager hayan perdido el acceso a Internet. Compruebe si la subred que se proporcionó al crear el clúster tenga una puerta de enlace de NAT y una VPC con puerta de enlace de Internet. Use el analizador de accesibilidad de la VPC para verificar que la instancia del plano de control puede llegar a la puerta de enlace de Internet. Para obtener más información, consulte Introducción al Analizador de accesibilidad de la VPC. Si sus redes privadas no tienen una conexión a Internet saliente, asegúrese de que todos los puntos de conexión de VPC y de puerta de enlace necesarios sigan presentes en la subred regional de su clúster (consulte the section called “Acceso a los servicios de AWS de la subred”).
- Faltan políticas en el ARN de rol que ha proporcionado. Verifique no se haya eliminado la política administrada de AWS AmazonEKSLocalOutpostClusterPolicy del rol.
- Es posible que una de las nuevas instancias del plano de control de Kubernetes haya sufrido un error de arranque inesperado. Envié un ticket con el Centro de AWS Support para obtener más orientación sobre la solución de problemas y la recopilación de informes en este caso excepcional.

No puede asociar nodos a un clúster

- Problemas con las AMI:
  - Está utilizando una AMI incompatible. Solo se admiten las AMI de Amazon Linux 2 optimizadas para Amazon EKS (`amazon-linux-2`, `amazon-linux-2-gpu`, `amazon-linux-2-arm64`). Si intenta unir nodos de AL2023 a LocalClusters de EKS en AWS Outposts, los nodos no podrán unirse al clúster. Para obtener más información, consulte Creación de nodos de Amazon Linux en AWS Outposts.
  - Si utilizó una plantilla de AWS CloudFormation para crear sus nodos, asegúrese de que no estaba utilizando una AMI no compatible.

- Falta el `ConfigMap` del Autenticador de IAM de AWS: si falta, tiene que crearlo. Para obtener más información, consulte the section called “Aplique el `ConfigMap` de `aws-auth` en su clúster” .
- Se ha usado un grupo de seguridad incorrecto: asegúrese de usar `eks-cluster-sg-`*`cluster-name`*`-`*`uniqueid`* para el grupo de seguridad de sus nodos de trabajo. AWS CloudFormation cambia el grupo de seguridad seleccionado para permitir un nuevo grupo de seguridad cada vez que se utilice la pila.
- Siguientes pasos inesperados de VPC de enlace privado: se pasan datos de CA incorrectos (`--b64-cluster-ca`) o del punto de conexión de la API (`--apiserver-endpoint`).

Recopilación de registros

Cuando un Outpost se desconecta del servidor al que está asociado la región de AWS, es probable que el clúster de Kubernetes siga funcionando con normalidad. Sin embargo, si el clúster no funciona correctamente, siga los pasos de solución de problemas que se indican en Preparación de los clústeres locales de Amazon EKS en AWS Outposts para las desconexiones de la red. Si encuentra algún problema, póngase en contacto con AWS Support. AWS Support puede guiarlo para descargar y ejecutar una herramienta de recopilación de registros. De esta forma, puede recopilar registros de sus instancias del plano de control del clúster de Kubernetes y enviarlas a AWS Support para una investigación en mayor profundidad.

No se puede acceder a las instancias del plano de control a través de AWS Systems Manager

Cuando no se puede acceder a las instancias del plano de control de Amazon EKS a través de AWS Systems Manager, Amazon EKS muestra el siguiente error para su clúster.

```
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.
```

Para resolver este problema, asegúrese de que su VPC y las subredes cumplen con los requisitos que están en Creación de una VPC y subredes para clústeres de Amazon EKS en AWS Outposts y que ha completado los pasos de Configuración del administrador de sesiones en la Guía del usuario de AWS Systems Manager.

Creación de nodos de Amazon Linux en AWS Outposts

Important

Los clústeres locales de Amazon EKS en Outposts solo admiten nodos creados desde las siguientes AMI de Amazon Linux 2023 optimizadas para Amazon EKS:

- Amazon Linux 2023 estándar (`amazon-linux-2023/x86_64/standard`)
- Amazon Linux 2023 acelerado por NVIDIA (`amazon-linux-2023/x86_64/nvidia`)
- Amazon Linux 2023 acelerado por Neuron (`amazon-linux-2023/x86_64/neuron`)

AWS finalizó el soporte para las AMI optimizadas para AL2 y aceleradas para AL2 de AKS el 26 de noviembre de 2025. Aunque aún podrá utilizar las AMI EKS AL2 después de la fecha de fin de soporte (EOS) (26 de noviembre de 2025), EKS dejará de publicar nuevas versiones de Kubernetes o actualizaciones de las AMI AL2, incluso versiones secundarias, revisiones y correcciones de errores después de esta fecha. Consulte esta página para obtener más información sobre la obsolescencia de AL2.

En este tema, se describe cómo puede lanzar grupos de escalado automático de nodos de Amazon Linux en un Outpost que se registrarán con el clúster de Amazon EKS. El clúster puede estar en la nube de AWS o en un Outpost.

- Un Outpost existente. Para obtener más información, consulte ¿Qué es AWS Outposts?
- Un clúster existente de Amazon EKS. Para implementar un clúster en la nube de AWS, consulte the section called “Creación de un clúster ”. Para implementar un clúster en un Outpost, consulte the section called “Ejecución de clústeres locales”.
- Supongamos que está creando sus nodos en un clúster en la nube de AWS y tiene subredes en la región de AWS donde tiene habilitados AWS Outposts, AWS Wavelength y las zonas locales de AWS. Esas subredes no deberían haberse transferido al crear el clúster. Si está creando los nodos en un clúster en un Outpost, debe haber proporcionado una subred de Outpost al crear el clúster.
- (Recomendado para clústeres en la nube de AWS) El complemento CNI de Amazon VPC para Kubernetes configurado con su propio rol de IAM que tiene asociada la política de IAM necesaria. Para obtener más información, consulte the section called “Configuración para IRSA”. Los clústeres locales no admiten roles de IAM para cuentas de servicio.

Puede crear un grupo de nodos autoadministrados de Amazon Linux con `eksctl` o la Consola de administración de AWS (con una plantilla de AWS CloudFormation). También puede usar Terraform.

Puede crear un grupo de nodos autoadministrado para el clúster local con las siguientes herramientas que se describen en esta página:

- the section called “`eksctl`”
- the section called “Consola de administración de AWS”

Important

- Un grupo de nodos autoadministrados incluye instancias de Amazon EC2 en su cuenta. Estas instancias no se actualizan de forma automática cuando usted o Amazon EKS actualizan la versión del plano de control en su nombre. Un grupo de nodos autoadministrados no tiene indicaciones en la consola de que necesita actualizarse. Puede ver la versión de `kubelet` instalada en un nodo al seleccionar el nodo en la lista de Nodos en la pestaña Overview (Información general) del clúster para determinar qué nodos deben actualizarse. Debe actualizar los nodos de forma manual. Para obtener más información, consulte the section called “Métodos de actualización”.
- Los certificados que utiliza kubelet en sus nodos autoadministrados se emiten con un año de validez. De forma predeterminada, la rotación de certificados no está habilitada (consulte: https://kubernetes.io/docs/reference/config-api/kubelet-config.v1beta1/#kubelet-config-k8s-io-v1beta1-KubeletConfiguration), lo que significa que, si tiene un nodo autoadministrado en funcionamiento durante más de un año, ya no podrá autenticarse en la API de Kubernetes.
- Como práctica recomendada, recomendamos a los clientes que actualicen periódicamente sus grupos de nodos autoadministrados para recibir los CVE y los parches de seguridad de la última AMI optimizada para Amazon EKS. La actualización de la AMI utilizada en los grupos de nodos autoadministrados también desencadena la recreación de los nodos y garantiza que no se produzcan problemas debido a la caducidad de los certificados de kubelet.
- Como alternativa, también puede habilitar la rotación de certificados de cliente (consulte: https://kubernetes.io/docs/tasks/tls/certificate-rotation/) al crear los grupos de nodos autoadministrados para asegurarse de que los certificados de kubelet se renueven a medida que el certificado actual se acerca a su fecha de caducidad.

eksctl

Para inicializar nodos de Linux autoadministrados mediante **eksctl**

1. Instale la versión `0.215.0` o posterior de la herramienta de línea de comandos de `eksctl` instalada en su dispositivo o AWS CloudShell. Para instalar o actualizar `eksctl`, consulte la sección de Instalación en la documentación de `eksctl`.
2. Si su clúster está en la nube de AWS y la política de IAM administrada AmazonEKS_CNI_Policy se adjunta a su Rol de IAM de nodo de Amazon EKS, recomendamos asignarlo a un rol de IAM asociado a la cuenta de servicios de `aws-node` de Kubernetes en su lugar. Para obtener más información, consulte the section called “Configuración para IRSA”. Si el clúster está en su Outpost, la política debe estar asociada a su rol de nodo.
3. El siguiente comando crea un grupo de nodos en un clúster existente. El clúster debe haberse creado con `eksctl`. Sustituya *`al-nodes`* por un nombre para su grupo de nodos. El nombre del grupo de nodos no puede tener más de 63 caracteres. Debe empezar por una letra o un dígito, pero también puede incluir guiones y guiones bajos como caracteres no iniciales. Reemplace *`my-cluster`* por el nombre de su clúster. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster. Si el clúster existe en un Outpost, reemplace *`id`* con el ID de una subred de Outpost. Si su clúster existe en la nube de AWS, reemplace *`id`* con el ID de una subred que no especificó cuando creó el clúster. Reemplace los valores de ejemplo restantes por sus propios valores. Los nodos se crean de forma predeterminada con la misma versión de Kubernetes que el plano de control.

   Reemplace *`instance-type`* con un tipo de instancia disponible en su Outpost.

   Reemplace *`my-key`* con el nombre de su par de claves de Amazon EC2 o la clave pública. Esta clave se utiliza para SSH en sus nodos después de que se lancen. Si aún no tiene un par de claves de Amazon EC2, puede crear uno en la Consola de administración de AWS. Para obtener más información, consulte Pares de claves de Amazon EC2 en la Guía del usuario de Amazon EC2.

   Cree el grupo de nodos con el siguiente comando.

```
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
```

Si su clúster está implementado en la nube de AWS:

- El grupo de nodos que implemente puede asignar direcciones `IPv4` a pods de un bloque de CIDR diferente que el de la instancia. Para obtener más información, consulte the section called “Redes personalizadas”.
- El grupo de nodos que implemente no requiere acceso a Internet saliente. Para obtener más información, consulte the section called “Clústeres privados”.

Para obtener una lista completa de todas las opciones y valores predeterminados disponibles, consulte Soporte de AWS Outposts en la documentación de `eksctl`.

- Si los nodos no se unen al clúster, consulte the section called “Los nodos no pueden unirse al clúster” en Solución de problemas con los clústeres y nodos de Amazon EKS y the section called “No puede asociar nodos a un clúster” en Solución de problemas de los clústeres locales de Amazon EKS en AWS Outposts.
- Un ejemplo de salida sería el siguiente. Se generan varias líneas mientras se crean los nodos. Una de las últimas líneas de salida es la siguiente línea de ejemplo.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

4. (Opcional) Implemente una aplicación de muestra para probar el clúster y los nodos de Linux.

Consola de administración de AWS

Paso 1: lanzamiento de nodos de Linux autoadministrados mediante la Consola de administración de AWS

1. Descargue la versión más reciente de la plantilla de AWS CloudFormation.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2025-11-24/
amazon-eks-outpost-nodegroup.yaml
```

2. Abra la Consola de AWS CloudFormation.
3. Seleccione Create stack (Crear pila) y, a continuación, seleccione With new resources (standard) (Con nuevos recursos [estándar]).

4. Para Especificar plantilla, seleccione Cargar un archivo de plantilla y, a continuación, elija Elegir archivo. Seleccione el archivo `amazon-eks-nodegroup.yaml` que ha descargado en el paso anterior y, a continuación, seleccione Siguiente.
5. En la página Especificar detalles de la pila, ingrese los siguientes parámetros según corresponda y luego seleccione Siguiente:
   - Nombre de pila: elija un nombre para su pila de AWS CloudFormation. Por ejemplo, puede llamarla *`al-nodes`*. El nombre solo puede contener caracteres alfanuméricos (con distinción de mayúsculas y minúsculas) y guiones. Debe comenzar con un carácter alfanumérico y no puede tener más de 100 caracteres. El nombre debe ser único dentro de la región de AWS y la cuenta de AWS en las que va a crear el clúster.
   - ApiServerEndpoint: ingrese el punto de conexión del servidor de la API de Kubernetes, visible en la consola de EKS o con la API DescribeCluster.
   - ClusterName: ingrese el nombre del clúster. Si este nombre no coincide con el nombre del clúster, los nodos no pueden unirse al clúster.
   - ClusterId: ingrese el ID que el servicio de EKS asignó al clúster. Visible con la API DescribeCluster. Si este ID no coincide con el ID del clúster, los nodos no pueden unirse al clúster.
   - CertificateAuthority: ingrese la cadena codificada en base64 de la autoridad de certificación de Kubernetes. Visible en la consola de EKS o con la API DescribeCluster.
   - ServiceCidr: ingrese el CIDR de los servicios de Kubernetes. Visible en la consola de EKS o con la API DescribeCluster.
   - ClusterControlPlaneSecurityGroup: elija el valor de SecurityGroups en la salida de AWS CloudFormation que generó cuando creó la VPC.

     En los siguientes pasos, se muestra una operación para recuperar el grupo aplicable.

     i. Abra la consola de Amazon EKS.

     ii. Elija el nombre del clúster.

     iii. Elija la pestaña Redes.

     iv. Use el valor de Grupo de seguridad adicional como referencia al realizar una selección en la lista desplegable ClusterControlPlaneSecurityGroup.
   - NodeGroupName: escriba un nombre para el grupo de nodos. Este nombre se puede utilizar más adelante para identificar el grupo de nodos de escalado automático que se crea para los nodos.

- NodeAutoScalingGroupMinSize: ingrese el número mínimo de nodos al que se puede reducir horizontalmente el grupo de escalado automático de nodos.
- NodeAutoScalingGroupDesiredCapacity: escriba el número deseado de nodos que desea escalar cuando se crea la pila.
- NodeAutoScalingGroupMaxSize: ingrese el número máximo de nodos que pueda alcanzar el grupo de Auto Scaling de nodos.
- NodeInstanceType: elija un tipo de instancia para los nodos. Si su clúster se ejecuta en la nube de AWS y desea obtener más información, consulte the section called “Tipos de instancias de Amazon EC2”. Si su clúster se ejecuta en un Outpost, solo puede seleccionar un tipo de instancia disponible en su Outpost.
- NodeImageIdSSMParam: rellenado previamente con el parámetro de Amazon EC2 Systems Manager de una AMI optimizada recientemente para Amazon EKS para una versión de Kubernetes variable. Para utilizar otra versión secundaria de Kubernetes compatible con Amazon EKS, reemplace *1.XX* por una versión admitida diferente. Recomendamos especificar la misma versión de Kubernetes que el clúster.

  Para usar una AMI acelerada optimizada para Amazon EKS, actualice el valor de *NodeImageIdSSMParam* al parámetro de SSM deseado. Consulte cómo recuperar los ID de AMI de EKS desde SSM aquí.

  > **Note**
  >
  > Las AMI de los nodos de Amazon EKS se basan en Amazon Linux. Puede realizar un seguimiento de los eventos de seguridad o privacidad de Amazon Linux en el Centro de seguridad de Amazon Linux seleccionando la pestaña de la versión que desee. También puede suscribirse a la fuente RSS correspondiente. Los eventos de seguridad y privacidad incluyen información general del problema, qué paquetes están afectados y cómo actualizar las instancias para corregir el problema.

- NodeImageId: (opcional) si utiliza una AMI personalizada propia (en lugar de una AMI optimizada para Amazon EKS), escriba un ID de AMI de nodo para la región de AWS. Si especifica un valor aquí, anula cualquier valor del campo NodeImageIdSSMParam.
- NodeVolumeSize: especifique un tamaño de volumen raíz para los nodos en GiB.
- NodeVolumeType: especifique un tipo de volumen raíz para sus nodos.
- KeyName: ingrese el nombre de un par de claves SSH de Amazon EC2 que pueda utilizar para conectar mediante SSH con los nodos después de haberlos lanzado. Si aún no tiene un par

de claves de Amazon EC2, puede crear uno en la Consola de administración de AWS. Para obtener más información, consulte Pares de claves de Amazon EC2 en la Guía del usuario de Amazon EC2.

> **Note**
>
> Si no proporciona un par de claves aquí, se produce un error al crear la pila de AWS CloudFormation.

- DisableIMDSv1: cada nodo admite de forma predeterminada la versión 1 (IMDSv1) e IMDSv2 del servicio de metadatos de la instancia. Puede desactivar IMDSv1. Para evitar que los nodos y pods futuros del grupo de nodos usen IMDSv1, establezca DisableIMDSv1 en Verdadero. Para obtener más información, consulte Configuración del servicio de metadatos de instancia. Para obtener más información sobre cómo restringir el acceso en sus nodos, consulte Restringir el acceso al perfil de instancias asignado al nodo de trabajo.
- VpcId: ingrese el ID de la VPC que ha creado. Antes de elegir una VPC, revise los Requisitos y consideraciones de la VPC.
- Subredes: si su clúster está en un Outpost, elija al menos una subred privada en la VPC. Antes de elegir las subredes, revise Requisitos y consideraciones de las subredes. Puede ver qué subredes son privadas abriendo cada enlace de subred desde la pestaña Redes de su clúster.

6. Seleccione las opciones que desee en la página Configurar las opciones de pila y, a continuación, elija Siguiente.
7. Seleccione la casilla de verificación a la izquierda de Reconozco que AWS podría crear recursos de IAM y, luego, seleccione Crear pila.
8. Una vez completada la creación de la pila, selecciónela en la consola y elija Salidas.
9. Anote el valor de NodeInstanceRoles correspondiente al grupo de nodos creado. Lo necesitará al configurar los nodos de Amazon EKS.

Paso 2: habilitación para que los nodos se unan al clúster

1. Verifique si ya tiene el `ConfigMap` de `aws-auth`.

```
kubectl describe configmap -n kube-system aws-auth
```

2. Si se le muestra un `ConfigMap` de `aws-auth`, actualícelo según sea necesario.

   a. Abra el icono `ConfigMap` para editar.

```
kubectl edit -n kube-system configmap/aws-auth
```

b. Añada una nueva entrada de `mapRoles` según sea necesario. Establezca el valor de `rolearn` en el valor de NodeInstanceRole que registró en el procedimiento anterior.

```
[...]
data:
  mapRoles: |
    - rolearn: <ARN of instance role (not instance profile)>
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
[...]
```

c. Guarde el archivo y salga del editor de texto.

3. Si recibe un error que indica "`Error from server (NotFound): configmaps "aws-auth" not found`, aplique el `ConfigMap` bursátil.

a. Descargue el mapa de configuración.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/
aws-auth-cm.yaml
```

b. En el archivo `aws-auth-cm.yaml`, establezca el `rolearn` al valor de NodeInstanceRole que ha registrado en el procedimiento anterior. Puede hacerlo con un editor de texto o puede reemplazar *`my-node-instance-role`* y ejecutar el siguiente comando:

```
sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-instance-
role|' aws-auth-cm.yaml
```

c. Aplique la configuración. Este comando puede tardar varios minutos en finalizar.

```
kubectl apply -f aws-auth-cm.yaml
```

4. Observe el estado de los nodos y espere a que aparezca el estado `Ready`.

```
kubectl get nodes --watch
```

Ingrese `Ctrl+C` para obtener un símbolo del intérprete de comandos.

> **Note**
>
> Si recibe cualquier error de tipo de recurso o autorización, consulte the section called "Acceso denegado o no autorizado (`kubectl`)" en el tema de solución de problemas.

Si los nodos no se unen al clúster, consulte the section called "Los nodos no pueden unirse al clúster" en Solución de problemas con los clústeres y nodos de Amazon EKS y the section called "No puede asociar nodos a un clúster" en Solución de problemas de los clústeres locales de Amazon EKS en AWS Outposts.

5. Instale el controlador de CSI de Amazon EBS. Para obtener más información, consulte Installation (Instalación) en GitHub. En la sección Set up driver permission (Configurar permiso de controlador), asegúrese de seguir las instrucciones de la opción Using IAM instance profile (Uso del perfil de instancia de IAM). Debe usar la clase de almacenamiento `gp2`. No se admite la clase de almacenamiento `gp3`.

   Para crear una clase de almacenamiento `gp2` en el clúster, realice los siguientes pasos.

   a. Ejecute el siguiente comando para crear un archivo `gp2-storage-class.yaml`.

```
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. Aplique el manifiesto al clúster.

```
kubectl apply -f gp2-storage-class.yaml
```

6. (Solo para nodos de GPU) Si ha elegido un tipo de instancia de GPU y una AMI acelerada optimizada para Amazon EKS, debe aplicar el complemento de dispositivo NVIDIA para Kubernetes como un DaemonSet en el clúster. Reemplace *vX.X.X* con la versión NVIDIA/k8s-device-plugin deseada antes de ejecutar el siguiente comando.

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/
deployments/static/nvidia-device-plugin.yml
```

Paso 3: acciones adicionales

1. (Opcional) Implemente una aplicación de muestra para probar el clúster y los nodos de Linux.
2. Si el clúster se implementa en un Outpost, omita este paso. Si el clúster se implementa en la nube de AWS, la siguiente información es opcional. Si la política de IAM administrada AmazonEKS_CNI_Policy se adjunta a su rol de IAM de nodos de Amazon EKS, recomendamos asignarla a un rol de IAM asociado a la cuenta de servicio de `aws-node` de Kubernetes en su lugar. Para obtener más información, consulte the section called “Configuración para IRSA”.

Descripción general de la inteligencia artificial (IA) y el machine learning (ML) en Amazon EKS

Amazon Elastic Kubernetes Service (EKS) es una plataforma de Kubernetes administrada que permite a las organizaciones implementar, administrar y escalar cargas de trabajo de IA y machine learning (ML) con una flexibilidad y un control incomparables. Basado en el ecosistema de Kubernetes de código abierto, EKS le permite aprovechar su experiencia actual en Kubernetes y, al mismo tiempo, se integra sin problemas con las herramientas y los servicios de AWS de código abierto.

Ya sea que esté entrenando modelos a gran escala, realizando inferencias en línea en tiempo real o implementando aplicaciones de IA generativa, EKS ofrece el rendimiento, la escalabilidad y la rentabilidad que exigen sus proyectos de IA y ML.

¿Por qué elegir EKS para IA/ML?

EKS es una plataforma de Kubernetes gestionada que le ayuda a implementar y gestionar cargas de trabajo complejas de IA/ML. Basada en el ecosistema de código abierto de Kubernetes, se integra con los servicios de AWS y proporciona el control y la escalabilidad necesarios para proyectos avanzados. Para los equipos que se inician en las implementaciones de IA/ML, las habilidades existentes en Kubernetes se transfieren directamente, lo que permite una orquestación eficiente de múltiples cargas de trabajo.

EKS es compatible con todo, desde la personalización del sistema operativo hasta el escalado de la computación, y su base de código abierto promueve la flexibilidad tecnológica, preservando la capacidad de elección para las futuras decisiones de infraestructura. La plataforma proporciona el rendimiento y las opciones de ajuste que requieren las cargas de trabajo de IA/ML, y admite funciones como:

- Control total del clúster para afinar los costos y las configuraciones sin abstracciones ocultas.
- Latencia de menos de un segundo para cargas de trabajo de inferencia en tiempo real en producción.
- Personalizaciones avanzadas, como GPU de varias instancias, estrategias multinube y afinación a nivel de sistema operativo.
- Capacidad para centralizar las cargas de trabajo mediante EKS como un orquestador unificado en todos los procesos de IA/ML.

Casos de uso clave

Amazon EKS proporciona una plataforma sólida para una amplia gama de cargas de trabajo de IA/ML, que admite diversas tecnologías y patrones de implementación:

- Inferencia en tiempo real (en línea): EKS permite realizar predicciones inmediatas sobre los datos entrantes, como la detección de fraudes, con una latencia inferior a un segundo mediante herramientas como TorchServe, Triton Inference Server y KServe en instancias Inf1 e Inf2 de Amazon EC2. Estas cargas de trabajo se benefician del escalado dinámico con Karpenter y KEDA, a la vez que utilizan Amazon EFS para partir los modelos en todos los pods. Amazon ECR Pull Through Cache (PTC) acelera las actualizaciones de los modelos, y los volúmenes de datos de Bottlerocket con volúmenes optimizados para Amazon EBS garantizan un acceso rápido a los datos.
- Entrenamiento del modelo general: las organizaciones utilizan EKS para entrenar modelos complejos en grandes conjuntos de datos durante periodos prolongados mediante el uso de Kubeflow Training Operator, Ray Serve y Torch Distributed Elastic en instancias P4d de Amazon EC2 e instancias Trn1 de Amazon EC2. Estas cargas de trabajo son compatibles con la programación por lotes con herramientas como Volcano, Yunikorn y Kueue. Amazon EFS permite compartir los puntos de control de los modelos, y Amazon S3 gestiona la importación y exportación de modelos con políticas de ciclo de vida para la administración de versiones.
- Procesos de generación aumentada por recuperación (RAG): EKS administra los chatbots de atención al cliente y aplicaciones similares mediante la integración de los procesos de recuperación y generación. Estas cargas de trabajo suelen utilizar herramientas como Argo Workflows y Kubeflow para la orquestación, bases de datos vectoriales como Pinecone, Weaviate o Amazon OpenSearch, y exponen las aplicaciones a los usuarios a través del controlador del equilibrador de carga de aplicación (LBC). NVIDIA NIM optimiza el uso de la GPU, mientras que Prometheus y Grafana supervisan el uso de los recursos.
- Implementación de modelos de IA generativa: las empresas implementan servicios de creación de contenido en tiempo real en EKS, como la generación de texto o imágenes, mediante Ray Serve, vLLM y Triton Inference Server en aceleradores de Amazon EC2 G5 y de Inferentia. Estas implementaciones optimizan el rendimiento y el uso de la memoria para modelos a gran escala. JupyterHub permite el desarrollo iterativo, Gradio proporciona interfaces web sencillas y el controlador CSI S3 Mountpoint permite montar buckets S3 como sistemas de archivos para acceder a archivos de modelos de gran tamaño.
- Inferencia por lotes (fuera de línea): las organizaciones procesan grandes conjuntos de datos de manera eficiente mediante trabajos programados con AWS Batch o Volcano. Estas cargas de

trabajo suelen utilizar instancias Inf1 e Inf2 EC2 para los chips Inferentia de AWS, instancias G4dn de Amazon EC2 para las GPU de NVIDIA T4 o instancias de CPU c5 yd c6i, lo que maximiza la utilización de los recursos durante las horas de menor actividad para las tareas de análisis. El SDK de AWS Neuron y los controladores de GPU de NVIDIA optimizan el rendimiento, mientras que el MIG/TS permite compartir la GPU. Las soluciones de almacenamiento incluyen Amazon S3, Amazon EFS y FSx para Lustre, con controladores de CSI para diversas clases de almacenamiento. La administración de modelos aprovecha herramientas como Kubeflow Pipelines, Argo Workflows y Ray Cluster, mientras que Prometheus, Grafana y herramientas de monitoreo de modelos personalizadas se encargan de la supervisión.

Casos prácticos

Los clientes eligen Amazon EKS por varios motivos, como optimizar el uso de la GPU o ejecutar cargas de trabajo de inferencia en tiempo real con una latencia inferior a un segundo, como se demuestra en los siguientes casos prácticos. Para ver una lista de todos los casos prácticos de Amazon EKS, consulte Historias de éxito de clientes de AWS.

- Unitary procesa 26 millones de videos al día con IA para moderar el contenido, lo que requiere inferencias de alto rendimiento y baja latencia, y ha conseguido reducir en un 80 % los tiempos de arranque de los contenedores, lo que garantiza una respuesta rápida a los eventos de escalado a medida que el tráfico fluctúa.
- Miro, la plataforma de colaboración visual que da soporte a 70 millones de usuarios en todo el mundo, informó de una reducción del 80 % en los costos informáticos en comparación con sus anteriores clústeres de Kubernetes autoadministrados.
- Synthesia, que ofrece la creación de videos mediante IA generativa como un servicio para que los clientes puedan crear videos realistas a partir de indicaciones, ha conseguido multiplicar por 30 el rendimiento del entrenamiento del modelo de ML.
- Harri, que proporciona tecnología de Recursos Humanos para el sector de la hostelería, logró escalar un 90 % más rápido en respuesta a los picos de demanda y redujo sus costos informáticos en un 30 % al migrar a los procesadores Graviton de AWS.
- Ada Support, una empresa de automatización del servicio al cliente impulsada por IA, logró una reducción del 15 % en los costos de cómputo junto con un aumento del 30 % en la eficiencia informática.

- Snorkel AI, que permite a las empresas crear y adaptar modelos básicos y modelos extensos de lenguaje, logró ahorrar más del 40 % al implementar mecanismos de escalado inteligentes para sus recursos de GPU.

Comience a utilizar el machine learning en EKS

Para comenzar a planificar y utilizar plataformas y cargas de trabajo de machine learning en EKS en la nube de AWS, continúe con la sección the section called “Recursos”.

Ejecución de cargas de trabajo de inferencia en línea en tiempo real en Amazon EKS

Esta sección está diseñada para ayudarlo a implementar y operar cargas de trabajo de inferencia en línea en tiempo real en Amazon Elastic Kubernetes Service (EKS). Encontrará instrucciones sobre cómo crear clústeres optimizados con nodos acelerados por GPU, cómo integrar servicios de AWS para almacenamiento y escalado automático, cómo implementar modelos de muestra para su validación y consideraciones clave de arquitectura, como desacoplar las tareas de CPU y GPU, seleccionar las AMI y los tipos de instancia adecuados, y garantizar una exposición de baja latencia de los puntos de conexión de inferencia.

Temas

- Guía de prácticas recomendadas para la configuración de clústeres para la inferencia en tiempo real en Amazon EKS
- Inicio rápido: inferencia de LLM de alto rendimiento con vLLM en Amazon EKS

Guía de prácticas recomendadas para la configuración de clústeres para la inferencia en tiempo real en Amazon EKS

Introducción

Esta guía ofrece un tutorial práctico para configurar un clúster de Amazon Elastic Kubernetes Service (EKS) optimizado para cargas de trabajo de inferencia en línea en tiempo real, e incorpora las prácticas recomendadas seleccionadas por expertos de todo el equipo de AWS. Utiliza una sofisticada arquitectura Quickstart de EKS: un conjunto seleccionado de controladores, tipos de instancias y configuraciones alineados con las prácticas recomendadas de AWS en materia

de modelos, aceleradores y escalado. Este enfoque le ayuda a evitar la tarea de seleccionar la configuración del clúster, lo que le permite poner en marcha rápidamente un clúster funcional y preconfigurado. A lo largo del camino, implementaremos ejemplos de cargas de trabajo para validar su configuración, explicaremos los conceptos clave sobre arquitectura (como desvincular las tareas vinculadas a la CPU de los cálculos con uso intensivo de la GPU), abordaremos preguntas comunes (por ejemplo, ¿por qué elegir la AMI de Bottlerocket en lugar de AL2023?) y describiremos los próximos pasos para ampliar las capacidades de su clúster.

Esta guía fue diseñada específicamente para ingenieros de machine learning (ML) e inteligencia artificial (IA), administradores de plataformas, operadores y especialistas en datos/IA que son nuevos en el ecosistema de AWS y EKS, por lo que se asume que están familiarizados con Kubernetes, pero que no tienen experiencia previa en EKS. Está diseñada para ayudarlo a comprender los pasos y procesos necesarios para poner en marcha las cargas de trabajo de inferencia en línea en tiempo real. La guía muestra los aspectos básicos para crear un clúster de inferencia de un solo nodo, como el aprovisionamiento de los recursos de la GPU, la integración del almacenamiento para los artefactos del modelo, la habilitación del acceso seguro a los servicios de AWS y la exposición de los puntos de conexión de inferencia. En todo momento, se hace hincapié en el diseño resiliente y de baja latencia para aplicaciones orientadas al usuario, como la detección de fraudes, los chatbots en tiempo real y el análisis de opiniones de los clientes.

En esta guía, nos centramos exclusivamente en establecer un punto de partida fundamental y prescriptivo mediante instancias G5 de EC2. Si busca configuraciones de clústeres específicas de AWS Inferentia o flujos de trabajo integrales, consulte the section called “Configuración de clústeres de inferencia con Inferentia” o nuestros talleres en the section called “Recursos”.

Antes de empezar

Antes de comenzar, asegúrese de haber realizado las siguientes tareas:

- Configuración de su entorno para Amazon EKS
- Instalación de la versión más reciente de eksctl
- Instale Helm
- (Opcional) Instalación de Docker
- (Opcional) Instalación de la CLI de NGC

Arquitectura

La inferencia en línea en tiempo real se refiere al proceso de utilizar un modelo de machine learning entrenado para generar predicciones o resultados sobre los flujos de datos entrantes con una latencia mínima. Por ejemplo, permite la detección de fraude en tiempo real, la clasificación de imágenes o la generación de gráficos de conocimiento en respuesta a las entradas de los usuarios. La arquitectura de un sistema de inferencia en línea en tiempo real ofrece predicciones de machine learning de baja latencia en aplicaciones orientadas al usuario al desvincular la gestión del tráfico web vinculado a la CPU de los cálculos de IA que utilizan mucha GPU. Por lo general, este proceso se encuentra dentro de un ecosistema de aplicaciones más amplio y, a menudo, incluye servicios de backend, frontend, vectores y modelos, con un enfoque en componentes especializados para permitir el escalado independiente, el desarrollo paralelo y la resiliencia ante los fallos. El aislamiento de las tareas de inferencia en un hardware de GPU dedicado y el aprovechamiento de interfaces como las API y los WebSockets garantizan una alta simultaneidad, un procesamiento rápido de modelos como los transformadores y las interacciones de los usuarios a través del frontend. Tenga en cuenta que, si bien las bases de datos vectoriales y los canales de generación aumentada por recuperación (RAG) suelen desempeñar un papel importante en los sistemas de inferencia en tiempo real, esta guía no contiene información sobre ellos. Como mínimo, una arquitectura típica suele incluir lo siguiente:

- Servicio de frontend: sirve como interfaz orientada al usuario, gestiona la lógica del lado del cliente, reproduce contenido dinámico y facilita las interacciones en tiempo real. Se comunica con el servicio de backend para iniciar solicitudes de inferencia y mostrar los resultados, a menudo iniciando solicitudes al servicio de backend, que utiliza WebSockets para transmitir actualizaciones o API para el intercambio de datos estructurados. Por lo general, este servicio no requiere un equilibrador de carga dedicado, ya que puede alojarse en redes de entrega de contenido (CDN), como AWS CloudFront, para activos estáticos o servirse directamente desde servidores web, y el escalado se gestiona mediante grupos de escalado automático si es necesario para el contenido dinámico.
- Servicio de backend: actúa como el orquestador de la aplicación y administra la lógica empresarial, como la autenticación de usuarios, la validación de datos y la coordinación de servicios (por ejemplo, mediante API para puntos de conexión RESTful o WebSockets para conexiones persistentes). Se comunica con el servicio de inferencia, escala de forma independiente en CPU multinúcleo y RAM para gestionar un tráfico web elevado sin depender de las GPU y, a menudo, requiere un equilibrador de carga (como el equilibrador de carga de aplicación o el equilibrador de carga de red de AWS) para distribuir las solicitudes entrantes entre varias instancias, especialmente en situaciones de alta concurrencia. Un controlador de ingreso puede

administrar aún más las reglas de acceso y enrutamiento externas para mejorar la seguridad y la configuración del tráfico.

- Servicio de inferencia: sirve como núcleo para los cálculos de IA y se ejecuta en GPU con suficiente VRAM (por ejemplo, de 8 a 12 GB para modelos como DistilBERT) para realizar incrustaciones de vectores, extracción de conocimientos e inferencia del modelo (por ejemplo, expuestos a través de API para solicitudes por lotes o WebSockets para transmisión en tiempo real) mediante modelos personalizados o de código abierto. Este aislamiento evita los conflictos de dependencias, permite actualizar el modelo sin tiempo de inactividad y permite el escalado horizontal con equilibrio de carga para múltiples solicitudes simultáneas. Para dar a conocer el servicio del modelo de forma eficaz, normalmente se basa en un equilibrador de carga para distribuir las cargas de trabajo vinculadas a la GPU entre las instancias replicadas, mientras que un recurso o controlador de entrada (como el controlador de Ingress del ALB en AWS) gestiona el enrutamiento externo, la terminación de SSL y el reenvío basado en rutas para garantizar un acceso seguro y eficiente sin sobrecargar las GPU individuales.

Descripción de la solución

Los sistemas de inferencia en línea en tiempo real requieren una arquitectura resiliente y de alto rendimiento que pueda ofrecer una latencia ultrabaja y, al mismo tiempo, gestionar ráfagas de tráfico impredecibles y de gran volumen. Esta descripción general de la solución explica cómo funcionan juntos los siguientes componentes de AWS en el clúster de Amazon EKS que crearemos para garantizar que nuestro clúster pueda alojar y administrar modelos de machine learning que proporcionen predicciones inmediatas sobre datos en tiempo real con un retraso mínimo para los usuarios finales.

- Instancias G5 de Amazon EC2: para las tareas de inferencia con uso intensivo de la GPU, utilizamos los tipos de instancias G5 de EC2 g5.xlarge y g5.2xlarge, que cuentan con una (1) sola GPU NVIDIA A10G con 24 GB de memoria (por ejemplo, 8 mil millones de parámetros en FP16). Basadas en la arquitectura Ampere de NVIDIA, estas GPU funcionan con GPU NVIDIA A10G Tensor Core y procesadores AMD EPYC de segunda generación, admiten de 4 a 8 vCPU, un ancho de banda de la red de hasta 10 Gbps y entre 250 y 450 GB de almacenamiento SSD NVMe local, lo que garantiza un movimiento de datos y una potencia de computación rápidos para modelos complejos, lo que las hace ideales para tareas de inferencia de baja latencia y alto rendimiento. La elección de un tipo de instancia de EC2 depende de la aplicación y del modelo (por ejemplo, modelo de imagen, video o texto) y de sus requisitos de latencia y rendimiento. Por ejemplo, si utiliza un modelo de imagen o video, es posible que desee utilizar instancias P5 de EC2 para obtener una latencia óptima en tiempo real. Recomendamos empezar con las instancias G5

de EC2, ya que son un buen punto de partida para ponerlas en marcha rápidamente y, después, evaluar si son las más adecuadas para sus cargas de trabajo mediante pruebas comparativas de rendimiento. Para casos más avanzados, considere las instancias G6 de EC2.

- Instancias M7g de Amazon EC2: para tareas que requieren un uso intensivo de la CPU, como el preprocesamiento de datos, la gestión de solicitudes de API, el alojamiento del controlador de Karpenter, los complementos y otros componentes del sistema, utilizamos el tipo de instancia M7g m5.xlarge de EC2. Las instancias M7g son instancias basadas en ARM que cuentan con 4 vCPU, 16 GB de memoria y un ancho de banda de la red de hasta 12,5 Gbps, y funcionan con procesadores Graviton3 de AWS. La elección de un tipo de instancia de EC2 depende de la aplicación y de los requisitos de computación, memoria y escalabilidad de la carga de trabajo. Para las cargas de trabajo optimizadas para cálculos, puede considerar las instancias C7 de EC2g, que también utilizan procesadores Graviton3, pero están optimizadas para ofrecer un rendimiento de computación superior al de las instancias M7g en determinados casos de uso. Como alternativa, las instancias C8g de EC2 más recientes (cuando estén disponibles) ofrecen un rendimiento de computación hasta un 30 % mejor que las instancias C7g. Recomendamos empezar con las instancias M7g de EC2 por su rentabilidad y compatibilidad con una amplia gama de cargas de trabajo (p. ej., servidores de aplicaciones, microservicios, servidores de juegos o almacenes de datos de tamaño medio) y, después, evaluar si son las adecuadas para sus cargas de trabajo mediante pruebas comparativas de rendimiento.
- Controlador de CSI de Mountpoint para Amazon S3: para las cargas de trabajo en instancias de una sola GPU en las que varios pods comparten una GPU (por ejemplo, varios pods programados en el mismo nodo para utilizar los recursos de la GPU), utilizamos el controlador de CSI de Mountpoint de S3 para optimizar el uso de la memoria, algo esencial para tareas como la inferencia de modelos grandes en configuraciones de baja complejidad y con poco margen de costos. Expone los buckets de Amazon S3 como un sistema de archivos tipo POSIX disponible para el clúster de Kubernetes, que permite a los pods de inferencia leer los artefactos del modelo (por ejemplo, los pesos del modelo) directamente en la memoria sin tener que descargarlos primero e introducir conjuntos de datos mediante operaciones de archivo estándar. Además, S3 tiene una capacidad de almacenamiento prácticamente ilimitada y acelera las cargas de trabajo de inferencia con un uso intensivo de datos. La elección de un controlador de CSI de almacenamiento depende de la aplicación y de los requisitos de rendimiento y latencia de la carga de trabajo. Si bien el controlador de CSI de FSx para OpenZFS ofrece una latencia inferior a un milisegundo para E/S aleatorias o volúmenes persistentes compartidos totalmente compatibles con POSIX en todos los nodos, recomendamos empezar con el controlador de CSI de Mountpoint de S3 debido a su escalabilidad, menores costos para conjuntos de datos de gran tamaño e integración incluida con el almacenamiento de objetos administrado por S3 para patrones de inferencia de lectura

intensa (por ejemplo, entradas de modelos de transmisión) y, a continuación, evaluar si es el adecuado para sus cargas de trabajo mediante pruebas comparativas de rendimiento.

- Agente de Pod Identity de EKS: para permitir el acceso a los servicios de AWS, utilizamos el agente de Pod Identity de EKS, que utiliza una única entidad principal de servicio y facilita las asociaciones de roles de IAM en el pod dentro del clúster de Amazon EKS. Pod Identity de EKS ofrece una alternativa simplificada al enfoque tradicional de roles de IAM para cuentas de servicio (IRSA), ya que utiliza una única entidad principal de servicio (pods.eks.amazonaws.com) en lugar de depender de proveedores de OIDC individuales para cada clúster, lo que facilita la asignación de permisos. Además, permite reutilizar los roles en varios clústeres y es compatible con características avanzadas, como las etiquetas de sesión de los roles de IAM y los roles de IAM de destino.
- Agente de supervisión de nodos de EKS: para garantizar la disponibilidad y fiabilidad continuas de los servicios de inferencia, utilizamos el agente de supervisión de nodos de EKS con reparación automática, que detecta y reemplaza automáticamente los nodos en mal estado, lo que minimiza el tiempo de inactividad. Supervisa continuamente los nodos para detectar problemas de hardware, kernel, redes y almacenamiento mediante comprobaciones de estado mejoradas (por ejemplo, KernelReady o NetworkingReady). En el caso de los nodos de GPU, detecta los fallos específicos del acelerador e inicia una reparación adecuada acordonando los nodos en mal estado, esperando 10 minutos a que se resuelvan los problemas transitorios de la GPU y sustituyendo los nodos después de 30 minutos en caso de fallos persistentes.
- AMI de Bottlerocket: para proporcionar una base de seguridad reforzada para nuestro clúster de EKS, utilizamos la AMI de Bottlerocket, que incluye solo los componentes esenciales necesarios para ejecutar los contenedores y ofrece tiempos de arranque mínimos para un escalado rápido. La elección de una AMI de nodo es específica de la aplicación y depende de los requisitos de personalización, seguridad y escalabilidad de la carga de trabajo. Si bien la AMI de AL2023 ofrece una mayor flexibilidad para las instalaciones y personalizaciones a nivel del host (por ejemplo, especificar un directorio de caché dedicado en un PV/PVC sin configuraciones de nodos adicionales), recomendamos empezar con la AMI de Bottlerocket, por su tamaño más reducido y su optimización integrada para cargas de trabajo en contenedores (por ejemplo, microservicios, servidores de inferencia, API escalables), y luego evaluar si es la adecuada para sus cargas de trabajo mediante pruebas comparativas de rendimiento.
- Controlador del equilibrador de carga (LBC) de AWS: para exponer los puntos de conexión de inferencia en tiempo real, utilizamos el controlador del equilibrador de carga de AWS, que aprovisiona y administra automáticamente los equilibradores de carga de aplicación (ALB) para el tráfico HTTP/HTTPS y los equilibradores de carga de red (NLB) para el tráfico TCP/UDP en función de los recursos de Ingress y de servicio de Kubernetes, lo que permite la

integración de modelos de inferencia con clientes externos. Además, admite características como el enrutamiento basado en rutas para distribuir las solicitudes de inferencia entre varios pods o nodos, lo que garantiza la escalabilidad durante los picos de tráfico y minimiza la latencia mediante optimizaciones nativas de AWS, como la multiplexación de conexiones y las comprobaciones de estado.

1. Creación de su clúster de EKS

En este paso, creamos un clúster con nodos de CPU y un grupo de nodos administrados mediante una plantilla eksctl ClusterConfig con tecnología de CloudFormation de AWS. Al iniciar el clúster solo con nodos de CPU, podemos usar Karpenter exclusivamente para administrar los nodos de GPU y con un uso intensivo de la CPU para optimizar la asignación de recursos mediante Karpenter NodePools, que crearemos en pasos posteriores. Para respaldar nuestras cargas de trabajo de inferencia en tiempo real, aprovisionamos el clúster con la AMI de Bottlerocket de EKS, el agente de supervisión de nodos de EKS, el agente de Pod Identity de EKS, el controlador de CSI de Mountpoint de S3, el controlador del equilibrador de carga (LBC) de AWS y los controladores kube-proxy, vpc-cni y coredns. Se utilizarán las instancias m7g.xlarge para tareas del sistema de la CPU, incluido el alojamiento del controlador de Karpenter, los complementos y otros componentes del sistema.

De forma predeterminada, `eksctl` creará una VPC dedicada para el clúster con un bloque de CIDR de `192.168.0.0/16`. La VPC incluye tres subredes públicas y tres subredes privadas, cada una distribuida en tres zonas de disponibilidad diferentes (o dos zonas de disponibilidad en la región `us-east-1`), lo que constituye el método ideal para implementar cargas de trabajo de Kubernetes. La plantilla también implementa una puerta de enlace de Internet, que proporciona acceso a Internet a las subredes públicas a través de las rutas predeterminadas en sus tablas de enrutamiento y una sola puerta de enlace NAT en una de las subredes públicas, con rutas predeterminadas en las tablas de enrutamiento de las subredes privadas que dirigen el tráfico saliente a través de la puerta de enlace NAT para el acceso a Internet. Para obtener más información sobre esta configuración, consulte Deploy Nodes to Private Subnets.

Comprobación de sus credenciales

Compruebe si sus credenciales de la CLI de AWS son válidas y pueden autenticarse con los servicios de AWS:

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

Si se ejecuta correctamente, la CLI devolverá los detalles sobre su identidad de AWS (ID de usuario, cuenta y ARN).

Comprobación de la disponibilidad de la instancia

Los tipos de instancia G5 no están disponibles en todas las regiones. Compruebe la región más cercana. Por ejemplo:

```
aws ec2 describe-instance-types --instance-types g5.xlarge g5.2xlarge --region us-
east-1
```

Si se ejecuta correctamente, el tipo de instancia G5 estará disponible en la región que especificó.

La AMI de Bottlerocket no está disponible en todas las regiones. Compruébelo recuperando un ID de la AMI de Bottlerocket para su región más cercana. Por ejemplo:

```
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
```

Si se ejecuta correctamente, la AMI de Bottlerocket estará disponible en la región que especificó.

Preparación del entorno

Primero, configure las siguientes variables de entorno en una nueva ventana de terminal. Nota: Asegúrese de sustituir los marcadores de posición de ejemplo por sus valores únicos, como el nombre del clúster, la región que desee, la versión de lanzamiento de Karpenter y la versión de Kubernetes.

> **Tip**
>
> Algunas variables (como `${AWS_REGION}` y `${K8S_VERSION}`) se definen al principio del bloque y luego se hace referencia a ellas en comandos posteriores para mantener la coherencia y evitar la repetición. Asegúrese de ejecutar los comandos en secuencia para que estos valores se exporten correctamente y estén disponibles para su uso en definiciones posteriores.

```
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/')"
```

Creación de los roles y las políticas necesarios

Karpenter necesita roles y políticas de IAM específicos (por ejemplo, el rol de IAM del controlador de Karpenter, el perfil de instancia y las políticas) para administrar las instancias de EC2 como nodos de trabajo de Kubernetes. Utiliza estos roles para realizar acciones como lanzar y terminar instancias de EC2, etiquetar recursos e interactuar con otros servicios de AWS. Cree los roles y las políticas de Karpenter utilizando el archivo cloudformation.yaml de 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}"
```

El LBC de AWS necesita permiso para aprovisionar y administrar equilibradores de carga de AWS, por ejemplo, la creación de ALB para los recursos de Ingress o NLB para servicios del tipo `LoadBalancer`. Especificaremos esta política de permisos durante la creación del clúster. Durante la creación del clúster, crearemos la cuenta de servicio con eksctl en ClusterConfig. Creación de la política de IAM de LBC:

```
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)"
```

Cuando se instala el controlador de CSI de Mountpoint de S3, se configuran sus pods DaemOnset para usar una cuenta de servicio para la ejecución. El Mountpoint para el controlador de CSI de Mountpoint de S3 necesita permiso para interactuar con el bucket de Amazon S3 que creemos más adelante en esta guía. Especificaremos esta política de permisos durante la creación del clúster. Durante la creación del clúster, crearemos la cuenta de servicio con eksctl en ClusterConfig. Creación de la política de IAM de S3:

```
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}/*\"]}]}"
```

Nota: Si ya existe un rol con este nombre, asígnele uno diferente. El rol que creamos en este paso es específico para su clúster y su bucket de S3.

Creación del clúster

En esta plantilla, eksctl crea automáticamente una cuenta de servicio de Kubernetes para Pod Identity de EKS, el agente de supervisión de nodos, CoreDNS, Kubeproxy y el complemento CNI de la VPC. A día de hoy, el controlador de CSI de Mountpoint de S3 no está disponible para Pod Identity de EKS, por lo que hemos creado roles de IAM para cuenta de servicios (IRSA) y un punto de conexión de OIDC. Además, creamos una cuenta de servicio para el controlador del equilibrador de carga (LBC) de AWS. Para acceder a los nodos de Bottlerocket, eksctl adjunta automáticamente AmazonSSMManagedInstanceCore a Bottlerocket para permitir sesiones seguras del intérprete de comandos a través de SSM.

En la misma terminal en la que configuró las variables de entorno, ejecute el siguiente bloque de comandos para crear el clúster:

```
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
```

Este proceso tarda varios minutos en completarse. Si desea supervisar el estado, consulte la consola de AWS CloudFormation.

2. Comprobación del estado del nodo del clúster y del pod

Realicemos algunas comprobaciones de estado para asegurarnos de que el clúster esté listo. Cuando se complete el comando anterior, observe los tipos de instancia y verifique, con el siguiente comando, que los nodos del sistema de CPU tengan el estado `Ready`:

```
kubectl get nodes -L node.kubernetes.io/instance-type
```

El resultado esperado debe tener el siguiente aspecto:

```
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
```

Compruebe, con el siguiente comando, todas las asociaciones de Pod Identity y de qué manera se asigna un rol a una cuenta de servicio en un espacio de nombres del clúster:

```
eksctl get podidentityassociation --cluster ${EKS_CLUSTER_NAME} --region ${AWS_REGION}
```

El resultado debería mostrar los roles de IAM para Karpenter (“karpenter”) y el LBC de AWS (“aws-load-balancer-controller”).

Compruebe que los DaemonSets estén disponibles:

```
kubectl get daemonsets -n kube-system
```

El resultado esperado debe tener el siguiente aspecto:

```
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
```

Compruebe que todos los complementos estén instalados en el clúster:

```
eksctl get addons --cluster ${EKS_CLUSTER_NAME} --region ${AWS_REGION}
```

El resultado esperado debe tener el siguiente aspecto:

```
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. Instalación de Karpenter

Instale el controlador de Karpenter en los nodos de trabajo de la CPU (`cpu-worker`) para optimizar los costos y conservar los recursos de la GPU. Lo instalaremos en el espacio de nombres “kube-system” y especificaremos la cuenta de servicio “karpenter” que definimos durante la creación del clúster. Además, este comando configura el nombre del clúster y una cola de interrupción de instancias de spot para los nodos de la CPU. Karpenter utilizará IRSA para asumir este rol de IAM.

```
# 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
```

El resultado esperado debe tener el siguiente aspecto:

```
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
```

Compruebe que Karpenter se esté ejecutando:

```
kubectl get pods -n kube-system -l app.kubernetes.io/name=karpenter
```

El resultado esperado debe tener el siguiente aspecto:

```
NAME                          READY   STATUS    RESTARTS   AGE
karpenter-555895dc-865bc      1/1     Running   0          5m58s
karpenter-555895dc-j7tk9      1/1     Running   0          5m58s
```

4. Configuración de NodePools de Karpenter

En este paso, configuramos los NodePools de Karpenter de CPU y GPU que se excluyen mutuamente. El campo `limits` de la especificación de NodePool restringe los recursos totales máximos (p. ej., CPU, memoria o GPU) que cada NodePool puede consumir en todos los nodos aprovisionados, lo que impide el aprovisionamiento de nodos adicionales si se superan estos límites. Si bien los NodePools admiten categorías de instancias amplias (por ejemplo, `c`, `g`), especificar tipos de instancias, tipos de capacidad y límites de recursos específicos le permite calcular más fácilmente los costos de sus cargas de trabajo bajo demanda. En estos NodePools, utilizamos un conjunto diverso de tipos de instancias de la familia de instancias G5. Esto le permite a Karpenter seleccionar automáticamente el tipo de instancia más adecuado en función de las solicitudes de recursos del pod, lo que optimiza la utilización de los recursos y, al mismo tiempo, respeta los límites totales de NodePool. Para obtener más información, consulte Creating NodePools.

Configuración del NodePool de GPU

En este NodePool, establecemos límites de recursos para administrar el aprovisionamiento de nodos con capacidades de GPU. Estos límites están diseñados para restringir los recursos totales en todos los nodos del grupo, por lo que se permite un máximo de 10 instancias en total. Cada instancia puede ser g5.xlarge (4 vCPU, 16 GiB de memoria, 1 GPU) o g5.2xlarge (8 vCPU, 32 GiB de memoria, 1 GPU), siempre que el total de vCPU no supere las 80, la memoria total no supere los 320 GiB y el total de GPU no supere las 10. Por ejemplo, el grupo puede aprovisionar 10 instancias g5.2xlarge (80 vCPU, 320 GiB, 10 GPU) o 10 instancias g5.xlarge (40 vCPU, 160 GiB, 10 GPU), o una combinación como, por ejemplo, 5 g5.xlarge y 5 g5.2xlarge (60 vCPU, 240 GiB, 10 GPU). Así, se garantiza la flexibilidad en función de las demandas de carga de trabajo y se respetan las limitaciones de recursos.

Además, especificamos el ID de la variante Nvidia de la AMI de Bottlerocket. Por último, establecemos una política de interrupciones para eliminar los nodos vacíos después de 30 minutos (`consolidateAfter: 30m`) y establecemos una vida útil máxima de los nodos de 30 días (`expireAfter: 720h`) para optimizar los costos y mantener el estado de los nodos para las tareas que requieren un uso intensivo de la GPU. Para obtener más información, consulte Disable Karpenter Consolidation for interruption sensitive workloads y Use ttlSecondsAfterFinished 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
```

El resultado esperado debe tener el siguiente aspecto:

```
nodepool.karpenter.sh/gpu-a10g-inference-g5 created
ec2nodeclass.karpenter.k8s.aws/gpu-a10g-inference-ec2 created
```

Compruebe que el NodePool esté creado y en buen estado:

```
kubectl get nodepool gpu-a10g-inference-g5 -o yaml
```

Busque `status.conditions` como `ValidationSucceeded: True`, `NodeClassReady: True`, y `Ready: True` para confirmar que el NodePool esté en buen estado.

Configuración del NodePool de la CPU

En este NodePool, establecemos límites para admitir aproximadamente 50 instancias, ajustándonos a una carga de trabajo de CPU moderada (por ejemplo, de 100 a 200 pods) y a las cuotas de vCPU de AWS típicas (por ejemplo, de 128 a 1152). Los límites se calculan suponiendo que NodePool debe escalar verticalmente hasta 50 instancias m7.xlarge: CPU (4 vCPU por instancia × 50 instancias = 200 vCPU) y memoria (16 GiB por instancia × 50 instancias = 800 GiB). Estos límites están diseñados para restringir los recursos totales en todos los nodos del grupo, por lo que se permiten hasta 50 instancias m7g.xlarge (cada una con 4 vCPU y 16 GiB de memoria), siempre que el total de vCPU no supere las 200 y la memoria total no supere los 800 GiB.

Además, especificamos el ID de la variante estándar de la AMI de Bottlerocket. Por último, establecemos una política de interrupciones para eliminar los nodos vacíos después de 60 minutos (`consolidateAfter: 60m`) y establecemos una vida útil máxima de los nodos de 30 días (`expireAfter: 720h`) para optimizar los costos y mantener el estado de los nodos para las

tareas que requieren un uso intensivo de la GPU. Para obtener más información, consulte Disable Karpenter Consolidation for interruption sensitive workloads y Use ttlSecondsAfterFinished to Auto Clean-Up Kubernetes Jobs.

```
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
```

El resultado esperado debe tener el siguiente aspecto:

```
nodepool.karpenter.sh/cpu-inference-m7gxlarge created
ec2nodeclass.karpenter.k8s.aws/cpu-inference-m7gxlarge-ec2 created
```

Compruebe que el NodePool esté creado y en buen estado:

```
kubectl get nodepool cpu-inference-m7gxlarge -o yaml
```

Busque `status.conditions` como `ValidationSucceeded: True`, `NodeClassReady: True`, y `Ready: True` para confirmar que el NodePool esté en buen estado.

5. Implementación de un pod de GPU para exponer una GPU

Necesita el complemento del dispositivo de Nvidia para que Kubernetes pueda exponer los dispositivos de GPU al clúster de Kubernetes. Normalmente, tendría que implementar el complemento como un DaemOnset; sin embargo, la AMI de Bottlerocket preinstala el complemento como parte de la AMI. Esto significa que, cuando se utilizan las AMI de Bottlerocket, no es necesario implementar el complemento DaemonSet del dispositivo Nvidia. Para obtener más información, consulte Kubernetes Device Plugin to expose GPUs.

Implementación de un pod de ejemplo

Karpenter actúa de forma dinámica: aprovisiona los nodos de la GPU cuando una carga de trabajo (pod) solicita recursos de la GPU. Para comprobar que los pods pueden solicitar y usar GPU, implemente un pod que solicite el recurso `nvidia.com/gpu` dentro de sus límites (por ejemplo, `nvidia.com/gpu: 1`). Para obtener más información sobre estas etiquetas, consulte Schedule workloads with GPU requirements using Well-Known labels.

```
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
```

El resultado esperado debe tener el siguiente aspecto:

```
pod/gpu-ndivia-smi created
```

Espere un minuto y compruebe si el pod tiene los estados “Pendiente”, “Creando un contenedor”, “En ejecución” y, luego, “Completado”.

```
kubectl get pod gpu-nvidia-smi -w
```

Verifique que el nodo del pod pertenezca al NodePool de la GPU:

```
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"
```

El resultado esperado debe tener el siguiente aspecto:

```
Name                                Nodepool
```

```
ip-192-168-83-245.ec2.internal   gpu-a10g-inference-g5
```

Compruebe los registros del pod:

```
kubectl logs gpu-nvidia-smi
```

El resultado esperado debe tener el siguiente aspecto:

```
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. (Opcional) Preparación y carga de los artefactos del modelo para su implementación

En este paso, implementará un servicio de modelo para la clasificación de imágenes en tiempo real, empezando por cargar los pesos de los modelos en un bucket de Amazon S3. A modo de demostración, utilizamos el modelo de visión GPUnet-0 de código abierto que forma parte de la GPUnet de NVIDIA y admite la inferencia de baja latencia en imágenes mediante GPU de NVIDIA y TensorRT. Este modelo está preentrenado en ImageNet, nos permite clasificar objetos en fotos o secuencias de video sobre la marcha y se considera un modelo pequeño con 11,9 millones de parámetros.

Configure su entorno

Para descargar los pesos del modelo GPUnet-0, en este paso, necesitará acceder al catálogo NGC de NVIDIA y al Docker instalado en su máquina local. Siga estos pasos para crear una cuenta gratuita y configurar la CLI de NGC:

- Cree una cuenta gratuita de NGC y genere una clave de API desde el panel de control de NGC (Ícono de usuario > Configuración > Generar clave de API > Generar clave personal > Catálogo de NGC).
- Descargue e instale la CLI de NGC (Linux/macOS/Windows) y configure la CLI mediante `ngc config set`. Introduzca su clave de API cuando se le pida; defina org como `nvidia` y pulse Aceptar para aceptar los valores predeterminados de los demás. Si se ejecuta con éxito, debería verse similar a `Successfully saved NGC configuration to /Users/your-username/.ngc/config`.

Verificación de los permisos de cuentas de servicio

Antes de empezar, compruebe los permisos de la cuenta de servicio de Kubernetes:

```
kubectl get serviceaccount s3-csi-driver-sa -n kube-system -o yaml
```

Durante la creación del clúster, adjuntamos la S3csiDriverPolicy a un rol de IAM y anotamos la cuenta de servicio (“s3-csi-driver-sa”). Los pods del controlador de CSI de Mountpoint de S3 heredan

los permisos del rol de IAM al interactuar con S3. El resultado esperado debe tener el siguiente aspecto:

```
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
```

Adición de una tolerancia

El controlador de CSI de S3 se ejecuta como un DaemOnset en todos los nodos. Los pods utilizan el controlador de CSI en esos nodos para montar los volúmenes de S3. Para permitir que se programe en los nodos de nuestra GPU que tengan taints, añada una tolerancia al 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"}}]'
```

El resultado esperado debe tener el siguiente aspecto:

```
daemonset.apps/s3-csi-node patched
```

Carga de los pesos del modelo en S3

En este paso, creará un bucket de Amazon S3, descargará los pesos del modelo de GPUnet-0 de NVIDIA GPU Cloud (NGC) y los subirá al bucket. Nuestra aplicación accederá a estos pesos durante el tiempo de ejecución para realizar inferencias.

Cree su bucket de Amazon S3:

```
aws s3 mb s3://${S3_BUCKET_NAME} --region ${AWS_REGION}
```

Habilite el control de versiones en S3 para el bucket, a fin de evitar que las eliminaciones y sobrescrituras accidentales provoquen una pérdida de datos inmediata y permanente:

```
aws s3api put-bucket-versioning --bucket ${S3_BUCKET_NAME} --versioning-configuration
 Status=Enabled
```

Aplique una regla de ciclo de vida al bucket para eliminar las versiones de objetos sobrescritas o eliminadas 14 días después de que dejen de estar actualizadas, para eliminar los marcadores de eliminación caducados y para eliminar las cargas incompletas de varias partes transcurridos 7 días. Para obtener más información, consulte Ejemplos de configuraciones de S3 Lifecycle.

```
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}}]}'
```

Descargue los pesos del modelo GPUnet-0 de NGC. Por ejemplo, en macOS:

```
ngc registry model download-version nvidia/dle/gpunet_0_pyt_ckpt:21.12.0_amp --dest ~/
downloads
```

> **Note**
>
> Es posible que tenga que ajustar este comando de descarga para su sistema operativo. Para que este comando funcione en un sistema Linux, es probable que tenga que crear el directorio como parte del comando (por ejemplo, `mkdir ~/downloads`).

El resultado esperado debe tener el siguiente aspecto:

```
{
  "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]"
}
```

Cambie el nombre del archivo de puntos de control para que coincida con el nombre esperado en el código de nuestra aplicación en pasos posteriores (no es necesario extraerlo, ya que es un punto de control estándar de PyTorch *.pth.tar que contiene el diccionario de estados del modelo):

```
mv ~/downloads/gpunet_0_pyt_ckpt_v21.12.0_amp/0.65ms.pth.tar gpunet-0.pth
```

Habilite el tiempo de ejecución de AWS común en la CLI de AWS para optimizar el rendimiento de S3:

```
aws configure set s3.preferred_transfer_client crt
```

Cargue los pesos del modelo en el bucket de S3:

```
aws s3 cp gpunet-0.pth s3://${S3_BUCKET_NAME}/gpunet-0.pth
```

El resultado esperado debe tener el siguiente aspecto:

```
upload: ./gpunet-0.pth to s3://eks-rt-inference-models-us-east-1-1752722786/
gpunet-0.pth
```

Creación del servicio de modelo

En este paso, configurará una aplicación web FastAPI para la clasificación de imágenes acelerada por la GPU mediante el modelo de visión GPUnet-0. La aplicación descarga los pesos del modelo de Amazon S3 en tiempo de ejecución, obtiene la arquitectura del modelo del repositorio de NVIDIA para almacenarla en caché y descarga las etiquetas de clase de ImageNet a través de HTTP. La aplicación incluye el preprocesamiento de imágenes, transforma y expone dos puntos de conexión: un punto de conexión GET raíz para comprobar el estado y un POST `/predict` que acepta la URL de una imagen.

Servimos el modelo mediante FastAPI con PyTorch, cargando pesos desde Amazon S3 en tiempo de ejecución en una configuración en contenedores para la creación rápida de prototipos y la implementación de Kubernetes. Para ver otros métodos, como el procesamiento por lotes optimizado o los motores de alto rendimiento, consulte Serving ML Models.

Creación de la aplicación

Cree un directorio para los archivos de su aplicación, como `model-testing`, cámbielo de directorio y añada el siguiente código a un nuevo archivo denominado `app.py`:

```
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))
```

Cree el Dockerfile.

El siguiente Dockerfile crea una imagen de contenedor para nuestra aplicación utilizando el modelo GPUnet del repositorio de GitHub NVIDIA Deep Learning Examples for Tensor Cores.

Reducimos el tamaño de la imagen del contenedor utilizando una base PyTorch solo en tiempo de ejecución, instalando solo los paquetes esenciales con limpieza de caché, almacenando previamente en caché el código del modelo y evitando "sobrecargar" los pesos en la imagen del contenedor para permitir extracciones y actualizaciones más rápidas. Para obtener más información, consulte Reducing Container Image Sizes.

En el mismo directorio que `app.py`, cree el `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"]
```

Pruebe la aplicación

Desde el mismo directorio que su `app.py` y `Dockerfile`, cree la imagen del contenedor para la aplicación de inferencia, orientada a la arquitectura de AMD64:

```
docker build --platform linux/amd64 -t gpunet-inference-app .
```

Defina variables de entorno para sus credenciales de AWS y, si lo desea, un token de sesión de AWS. Por ejemplo:

```
export AWS_REGION="us-east-1"
export AWS_ACCESS_KEY_ID=ABCEXAMPLESCUJFEIELSMUHHAZ
export AWS_SECRET_ACCESS_KEY=123EXAMPLEMZREoQXr8XkiicsOgWDQ5TpUsq0/Z
```

Ejecute el contenedor de forma local e inserte credenciales de AWS como variables de entorno para el acceso a S3. Por ejemplo:

```
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
```

El resultado esperado debe tener el siguiente aspecto:

```
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)
```

En una nueva ventana de terminal, pruebe el punto de conexión de inferencia enviando un ejemplo de solicitud POST con una URL de imagen pública como parámetro de consulta:

```
curl -X POST "http://localhost:8080/predict?image_url=http://images.cocodataset.org/
test-stuff2017/000000024309.jpg"
```

El resultado esperado debería ser una respuesta en JSON con las 5 mejores predicciones, similar a esta (las etiquetas y probabilidades reales pueden variar ligeramente según la imagen y la precisión del modelo):

```
{"predictions":[{"label":"desk","probability":0.28885871171951294},
{"label":"laptop","probability":0.24679335951805115},
{"label":"notebook","probability":0.08539070934057236},
{"label":"library","probability":0.030645888298749924},
{"label":"monitor","probability":0.029896063730120 66}]}
```

Cierre la aplicación presionando “Ctrl + C”.

Inserción del contenedor en Amazon ECR

En este paso, subimos la imagen del contenedor del servicio de modelo GPUnet-0 a Amazon Elastic Container Registry (ECR) para que esté disponible para su implementación en Amazon EKS. Este proceso implica crear un nuevo repositorio de ECR para almacenar la imagen, autenticarla con ECR y, a continuación, etiquetar y enviar la imagen del contenedor a nuestro registro.

En primer lugar, vuelva al directorio en el que configuró las variables de entorno al principio de esta guía. Por ejemplo:

```
cd ..
```

Cree un repositorio en Amazon ECR:

```
aws ecr create-repository --repository-name gpunet-inference-app --region ${AWS_REGION}
```

Inicie sesión en Amazon ECR:

```
aws ecr get-login-password --region ${AWS_REGION} | docker login --username AWS --
password-stdin ${AWS_ACCOUNT_ID}.dkr.ecr.${AWS_REGION}.amazonaws.com
```

El resultado esperado debe tener el siguiente aspecto:

```
Login Succeeded
```

Etiquete la imagen:

```
docker tag gpunet-inference-app:latest ${AWS_ACCOUNT_ID}.dkr.ecr.
${AWS_REGION}.amazonaws.com/gpunet-inference-app:latest
```

Envíe la imagen a su repositorio de Amazon ECR:

```
docker push ${AWS_ACCOUNT_ID}.dkr.ecr.${AWS_REGION}.amazonaws.com/gpunet-inference-
app:latest
```

Este último paso demora varios minutos en completarse.

7. (Opcional) Exposición del servicio de modelo

En este paso, expondrá su servicio de modelo de inferencia en tiempo real de forma externa en Amazon EKS mediante el controlador del equilibrador de carga (LBC) de AWS. Esto implica configurar el LBC, montar los pesos del modelo de Amazon S3 como un volumen persistente mediante el controlador de CSI de Mountpoint de S3, implementar un pod de aplicaciones acelerado por GPU, crear un servicio y una entrada para aprovisionar un equilibrador de carga de aplicación (ALB) y probar el punto de conexión.

En primer lugar, compruebe la asociación de Pod Identity con el LBC de AWS y confirme que la cuenta de servicio esté correctamente vinculada al rol de IAM requerido:

```
eksctl get podidentityassociation --cluster ${EKS_CLUSTER_NAME} --namespace kube-system
 --service-account-name aws-load-balancer-controller
```

El resultado esperado debe tener el siguiente aspecto:

```
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
```

Etiquetado del grupo de seguridad del clúster

El controlador del equilibrador de carga de AWS solo admite un grupo de seguridad único con la clave de etiqueta `karpenter.sh/discovery: "${EKS_CLUSTER_NAME}"` para la selección del grupo de seguridad de Karpenter. Al crear un clúster con eksctl, el grupo de seguridad del clúster predeterminado (que tiene la etiqueta `"kubernetes.io/cluster/<cluster-name>: owned"`) no se etiqueta automáticamente con etiquetas `karpenter.sh/discovery`. Esta etiqueta es esencial para que Karpenter detecte y adjunte este grupo de seguridad a los nodos que aprovisiona. La conexión de este grupo de seguridad garantiza la compatibilidad con el controlador

del equilibrador de carga (LBC) de AWS, lo que le permite administrar automáticamente las reglas de tráfico entrante para los servicios expuestos a través de Ingress, como el servicio del modelo en estos pasos.

Exporte el ID de la VPC del clúster:

```
CLUSTER_VPC_ID="$(aws eks describe-cluster --name ${EKS_CLUSTER_NAME} --query
 cluster.resourcesVpcConfig.vpcId --output text)"
```

Exporte el grupo de seguridad predeterminado para el clúster:

```
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)"
```

Agregue la etiqueta `karpenter.sh/discovery` al grupo de seguridad del clúster predeterminado. Esto permitirá que nuestros selectores EC2NodeClass de CPU y GPU lo utilicen:

```
aws ec2 create-tags --resources ${CLUSTER_SG_ID} --tags Key=karpenter.sh/
discovery,Value=${EKS_CLUSTER_NAME}
```

Verifique que se haya agregado la etiqueta:

```
aws ec2 describe-security-groups --group-ids ${CLUSTER_SG_ID} --query
 "SecurityGroups[].Tags"
```

Entre los resultados, debería ver lo siguiente con la etiqueta y el nombre del clúster. Por ejemplo:

```
{
  "Key": "karpenter.sh/discovery",
  "Value": "eks-rt-inference-us-east-1"
}
```

Configuración del controlador del equilibrador de carga de AWS (LBC)

El LBC de AWS es esencial para administrar el tráfico de entrada a las cargas de trabajo de IA/ML en Amazon EKS, lo que garantiza el acceso a los puntos de conexión de inferencia o a los canales de procesamiento de datos. Al integrarse con los equilibradores de carga de aplicación

(ALB) y los equilibradores de carga de red (NLB) de AWS, el LBC dirige dinámicamente el tráfico a aplicaciones en contenedores, como las que ejecutan modelos de lenguaje de gran tamaño, modelos de visión artificial o servicios de inferencia en tiempo real. Como ya creamos la cuenta de servicio y la asociación con Pod Identity durante la creación del clúster, configuramos el `serviceAccount.name` para que coincidan con lo definido en la configuración de nuestro clúster (`aws-load-balancer-controller`).

Añada el repositorio de gráficos de Helm eks-charts, propiedad de AWS:

```
helm repo add eks https://aws.github.io/eks-charts
```

Actualice sus repositorios locales de Helm con los gráficos más recientes:

```
helm repo update eks
```

Implemente el LBC de AWS con Helm, especificando el nombre del clúster de EKS y haciendo referencia a la cuenta de servicio creada previamente:

```
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
```

El resultado esperado debe tener el siguiente aspecto:

```
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!
```

Montaje del modelo en un volumen persistente

En este paso, montará los pesos del modelo desde su bucket de Amazon S3 mediante un PersistentVolume (PV) respaldado por el controlador de CSI Mountpoint para Amazon S3. Esto

permite que los pods de Kubernetes accedan a los objetos de S3 como archivos locales, lo que elimina las descargas que consumen muchos recursos a contenedores de almacenamiento de pods efímeros o de inicio, lo que resulta ideal para modelos grandes de varios gigabytes de peso.

El PV monta toda la raíz del bucket (no se ha especificado ninguna ruta en `volumeAttributes`), admite el acceso simultáneo de solo lectura desde varios pods y expone archivos como los pesos del modelo (`/models/gpunet-0.pth`) dentro del contenedor para la inferencia. Esto garantiza que no se active la "descarga" alternativa en nuestra aplicación (`app.py`) porque el archivo existe a través del montaje. Al desacoplar el modelo de la imagen del contenedor, se permite el acceso compartido y las actualizaciones independientes de las versiones del modelo sin necesidad de recompilar la imagen.

Creación del PersistentVolume (PV)

Cree un recurso PersistentVolume (PV) para montar el bucket de S3 que contenga los pesos del modelo, lo que permitirá el acceso de solo lectura a varios pods sin descargar archivos en tiempo de ejecución:

```
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
```

Creación del PersistentVolumeClaim (PVC)

Cree un PersistentVolumeClaim (PVC) para enlazarlo al PV y solicite acceso de solo lectura a los datos del modelo de S3 montado:

```
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
```

Implemente de la aplicación

Implemente la aplicación de inferencia como una implementación de Kubernetes, montando el volumen persistente respaldado por S3 para acceder al modelo, aplicando selectores y tolerancias de nodos de la GPU y configurando variables de entorno para la ruta del modelo. Esta implementación establece la ruta del modelo (var. del ent. de "`/models/gpunet-0.pth`"), por lo que nuestra aplicación (en `app.py`) la utilizará de forma predeterminada. Con el volumen de la implementación en `/models` (solo lectura), la descarga del modelo no se activará si el archivo ya está presente en el PVC.

```
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
```

Karpenter tardará unos minutos en aprovisionar un nodo de GPU si aún no hay uno disponible. Compruebe que el pod de inferencia tenga el estado “En ejecución”:

```
kubectl get pods -l app=gpunet-inference-app
```

El resultado esperado debe tener el siguiente aspecto:

```
NAME                                READY   STATUS    RESTARTS   AGE
```

```
gpunet-inference-app-5d4b6c7f8-abcde        1/1     Running   0          2m
```

Exposición del servicio con Ingress y el equilibrador de carga

Cree un servicio ClusterIP para exponer la implementación de inferencia internamente dentro del clúster de EKS, dirigido al puerto de la aplicación:

```
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
```

Cree un recurso de Ingress para aprovisionar un equilibrador de carga de aplicación (ALB) con acceso a Internet a través del LBC de AWS y enrute el tráfico externo al servicio de inferencia:

```
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
```

Espere unos minutos para que el equilibrador de carga de aplicación (ALB) termine de aprovisionar. Supervise el estado de los recursos de Ingress para confirmar que el ALB se ha aprovisionado:

```
kubectl get ingress gpunet-model-ingress
```

El resultado esperado debería tener este aspecto (con el campo ADDRESS lleno):

```
NAME                   CLASS   HOSTS   ADDRESS
 PORTS   AGE
gpunet-model-ingress   alb     *       k8s-default-gpunetmo-183de3f819-516310036.us-
east-1.elb.amazonaws.com   80      6m58s
```

Extraiga y exporte el nombre de host del ALB del estado de Ingress para usarlo en pruebas posteriores:

```
export ALB_HOSTNAME=$(kubectl get ingress gpunet-model-ingress -o
 jsonpath='{.status.loadBalancer.ingress[0].hostname}')
```

Prueba del servicio de modelo

Valide el punto de conexión de inferencia expuesto enviando una solicitud POST con una URL de imagen de muestra (por ejemplo, del conjunto de datos COCO), simulando una predicción en tiempo real:

```
curl -X POST "http://${ALB_HOSTNAME}/predict?image_url=http://images.cocodataset.org/
test-stuff2017/000000024309.jpg"
```

El resultado esperado debería ser una respuesta en JSON con las 5 mejores predicciones, similar a esta (las etiquetas y probabilidades reales pueden variar ligeramente según la imagen y la precisión del modelo):

```
{"predictions":[{"label":"desk","probability":0.2888975441455841},
{"label":"laptop","probability":0.2464350312948227},
{"label":"notebook","probability":0.08554483205080032},
```

```
{"label":"library","probability":0.030612602829933167},
{"label":"monitor","probability":0.029896672815084457}]}
```

Si lo desea, puede continuar probando otras imágenes en una nueva solicitud POST. Por ejemplo:

```
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
```

Conclusión

En esta guía, configurará un clúster de Amazon EKS optimizado para cargas de trabajo de inferencia en tiempo real aceleradas por GPU. Usted ha aprovisionado un clúster con instancias G5 de EC2, ha instalado el controlador de CSI de Mountpoint de S3, el agente de Pod Identity de EKS, el agente de supervisión de nodos de EKS, la AMI de Bottlerocket, el controlador del equilibrador de carga (LBC) de AWS y Karpenter para administrar los NodePools de CPU y GPU. Ha utilizado el complemento de dispositivo de NVIDIA para habilitar la programación de la GPU y ha configurado S3 con PersistentVolume y PersistentVolumeClaim para acceder a los modelos. Ha validado la configuración implementando un pod de GPU de muestra, configurando el acceso al modelo GPUNet-0 de NVIDIA en Amazon S3, habilitando la inicialización del pod y exponiendo el servicio de inferencia mediante el equilibrador de carga de aplicación. Para aprovechar al máximo su clúster, configure el agente de supervisión de nodos de EKS con reparación automática. Asegúrese de realizar pruebas comparativas, incluidas las evaluaciones del desempeño, la latencia y el rendimiento de la GPU para optimizar los tiempos de respuesta. Para obtener más información, consulte Using Monitoring and Observability Tools for your AI/ML Workloads.

Limpieza

Para evitar incurrir en cargos futuros, debe eliminar manualmente la pila de CloudFormation asociada para borrar todos los recursos creados durante esta guía, incluida la red de la VPC.

Elimine la pila de CloudFormation usando el indicador `--wait` con eksctl:

```
eksctl delete cluster --region ${AWS_REGION} --name ${EKS_CLUSTER_NAME} --wait
```

Cuando se haya completado, debería ver la siguiente respuesta de salida:

```
2025-07-29 13:03:55 [#]  all cluster resources were deleted
```

Elimine el bucket de Amazon S3 creado durante esta guía desde la consola de Amazon S3.

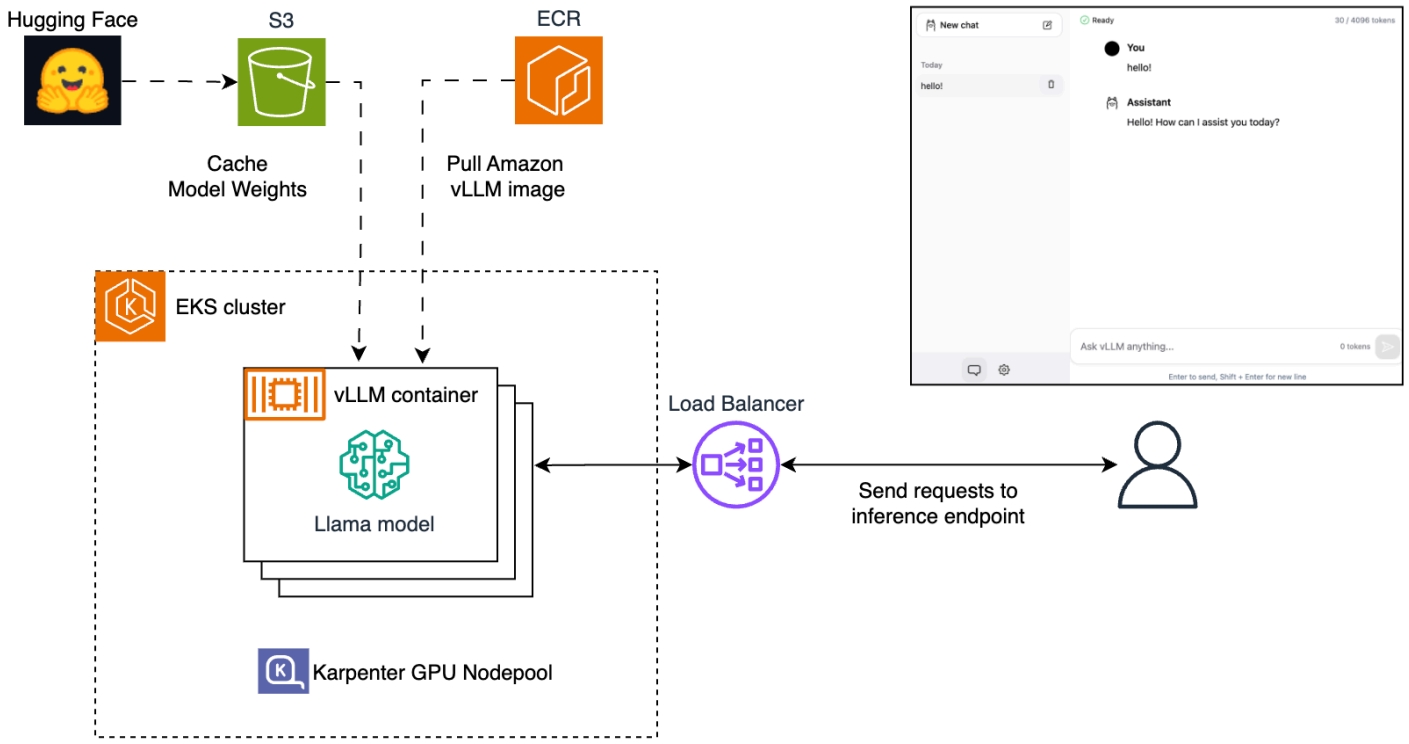
Inicio rápido: inferencia de LLM de alto rendimiento con vLLM en Amazon EKS

Introducción

Esta guía de inicio rápido proporciona un recorrido por la implementación de modelos de lenguaje de gran tamaño (LLM) en Amazon EKS mediante vLLM y GPU para aplicaciones de inferencia en tiempo real basadas en texto.

La solución usa Amazon EKS para la orquestación de contenedores y vLLM para un servicio de modelos eficiente, lo que le permite crear aplicaciones de IA escalables con aceleración de GPU y servidores de inferencias de alto rendimiento. El modelo Llama 3.1 8B Instruct se utiliza para la demostración, pero puede implementar cualquier otro LLM compatible con vLLM (consulte la documentación de vLLM para obtener una lista de los modelos compatibles). Para probar la inferencia del LLM, utilizamos una aplicación de chatbot de muestra basada en el proyecto nextjs-vllm-ui. Por último, utilizamos GuideLLM para comparar y ajustar los parámetros de configuración de vLLM a fin de optimizar el rendimiento de la inferencia.

Arquitectura de vLLM en EKS



Cuando complete este procedimiento, dispondrá de un punto de conexión de inferencia de vLLM optimizado para ofrecer un rendimiento y una baja latencia, y podrá interactuar con un modelo de Llama a través de una aplicación de frontend de chat, lo que demuestra un caso de uso típico para asistentes de chatbot y otras aplicaciones basadas en LLM.

Para obtener orientación adicional y recursos de implementación avanzados, consulte Guía de prácticas recomendadas de EKS sobre cargas de trabajo de IA y ML y los gráficos de inferencia de IA en EKS listos para la producción.

Antes de empezar

Antes de comenzar, asegúrese de que disponga de lo siguiente:

- Un clúster de Amazon EKS con los siguientes componentes principales: grupos de nodos de Karpenter con la familia de instancias de EC2 G5 o G6, el complemento para dispositivos de NVIDIA instalado en los nodos de trabajo con GPU y el controlador CSI de Mountpoint de S3 instalado. Para crear esta configuración de línea de base, siga los pasos de the section called “Creación de un clúster” hasta completar el paso 4.
- Una cuenta de Hugging Face. Para registrarse, consulte https://huggingface.co/login.

Configuración de almacenamiento de modelos con Amazon S3

Almacene archivos de LLM de gran tamaño de forma eficiente en Amazon S3 para separar el almacenamiento de los recursos de computación. Este enfoque agiliza las actualizaciones de los modelos, reduce los costos y simplifica la administración en las configuraciones de producción. S3 administra archivos de gran tamaño de forma fiable, mientras que la integración con Kubernetes mediante el controlador CSI de Mountpoint permite a los pods acceder a modelos como el almacenamiento local sin necesidad de largas descargas durante el arranque. Siga estos pasos para crear un bucket de S3, cargar un LLM y montarlo como un volumen en su contenedor de servidores de inferencias.

También hay otras soluciones de almacenamiento disponibles en EKS para el almacenamiento en caché de modelos, como EFS y FSx para Lustre. Para obtener más información, consulte las prácticas recomendadas de EKS.

Configuración de las variables de entorno

Cree un nombre único para un nuevo bucket de Amazon S3 que crearemos más adelante en esta guía. Una vez creado, utilice el mismo nombre del bucket para todos los pasos. Por ejemplo:

```
MY_BUCKET_NAME=model-store-$(date +%s)
```

Defina las variables de entorno y guárdelas en un archivo:

```
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
```

Cargue las variables de entorno en su entorno de intérprete de comandos. Si cierra el entorno de intérprete de comandos actual y abre uno nuevo, asegúrese de reutilizar las variables de entorno con este mismo comando:

```
source .env-quickstart-vllm
```

Creación de un bucket de S3 para almacenar archivos del modelo

Cree un bucket de S3 para almacenar los archivos del modelo:

```
aws s3 mb s3://${BUCKET_NAME} --region ${AWS_REGION}
```

Descarga del modelo desde Hugging Face

Hugging Face es uno de los principales centros de modelos para acceder a modelos de LLM. Para descargar el modelo Llama, tendrá que aceptar la licencia del modelo y configurar la autenticación mediante token:

1. Acepte la licencia del modelo Llama 3.1 8B Instruct en https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct.
2. Genere un token de acceso (vaya a Perfil > Configuración > Tokens de acceso y, a continuación, cree un nuevo token con el tipo de token de lectura).

Establezca una variable de entorno con el token de Hugging Face:

```
export HF_TOKEN=your_token_here
```

Instale el paquete pip3 si aún no está instalado en su entorno. Comando de ejemplo en Amazon Linux 2023:

```
sudo dnf install -y python3-pip
```

Instale la CLI de Hugging Face:

```
pip install huggingface-hub
```

Descargue el modelo Llama-3.1-8B-Instruct desde Hugging Face (~15 GB) con el indicador `--exclude` para omitir el formato PyTorch heredado y descargar solo los archivos optimizados en formato safetensors, lo que reduce el tamaño de la descarga y mantiene la compatibilidad total con los motores de inferencia más populares:

```
huggingface-cli download meta-llama/Meta-Llama-3.1-8B-Instruct \
  --exclude "original/*" \
  --local-dir ./llama-3.1-8b-instruct \
  --token $HF_TOKEN
```

Compruebe los archivos descargados:

```
$ ls llama-3.1-8b-instruct
```

El resultado esperado debe tener el siguiente aspecto:

```
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
```

Carga de archivos del modelo

Active Common Runtime (CRT) de AWS para mejorar el rendimiento de las transferencias de S3. El cliente de transferencia basado en CRT proporciona un rendimiento y una fiabilidad mejorados para operaciones con archivos de gran tamaño:

```
aws configure set s3.preferred_transfer_client crt
```

Cargue el modelo:

```
aws s3 cp ./llama-3.1-8b-instruct s3://$BUCKET_NAME/llama-3.1-8b-instruct \
  --recursive
```

El resultado esperado debe tener el siguiente aspecto:

```
...
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
```

Configuración los permisos de CSI de Mountpoint de S3

El controlador CSI de Mountpoint de S3 permite la integración nativa entre Kubernetes y S3, lo que permite a los pods acceder directamente a archivos del modelo como si se tratara de un almacenamiento local, lo que elimina la necesidad de llevar a cabo copias locales durante el arranque del contenedor.

Cree una política de IAM para permitir que Mountpoint de S3 lea desde su bucket de S3:

```
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}/*\"]}]}"
```

Para buscar el nombre del rol de IAM que utiliza el controlador CSI de Mountpoint de S3, consulte las anotaciones de la cuenta de servicio del controlador CSI de S3:

```
ROLE_NAME=$(kubectl get serviceaccount s3-csi-driver-sa -n kube-system -o
 jsonpath='{.metadata.annotations.eks\.amazonaws\.com/role-arn}' | cut -d'/' -f2)
```

Adjunte su política de IAM al rol de CSI de Mountpoint de S3:

```
aws iam attach-role-policy \
 --role-name ${ROLE_NAME} \
 --policy-arn arn:aws:iam::${AWS_ACCOUNT_ID}:policy/S3BucketAccess-${BUCKET_NAME}
```

Si CSI de Mountpoint de S3 no está instalado en el clúster, siga los pasos de implementación que se indican en the section called “Creación de un clúster”.

Montaje de un bucket de S3 como un volumen de Kubernetes

Cree un volumen persistente (PV) y una recuperación de volumen persistente (PVC) para proporcionar acceso de solo lectura al bucket de S3 en varios pods de inferencia. El modo de acceso ReadOnlyMany garantiza el acceso simultáneo a los archivos del modelo, mientras que el controlador CSI se encarga del montaje del bucket de S3:

```
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
```

Configuración de la infraestructura de GPU

Nodos del clúster

Estamos utilizando el clúster de EKS creado en the section called “Creación de un clúster”. Este clúster incluye grupos de nodos de Karpenter que pueden aprovisionar nodos habilitados para GPU con suficiente almacenamiento de nodos para descargar la imagen del contenedor de vLLM. Si utiliza un clúster de EKS personalizado, asegúrese de que pueda lanzar nodos habilitados para GPU.

Selección de una instancia

La selección correcta de instancias para la inferencia de LLM requiere garantizar que la memoria de GPU disponible sea suficiente para cargar el peso del modelo. El peso del modelo para Llama 3.1 8B Instruct es de aproximadamente 16 GB (tamaño de los archivos del modelo .safetensor), por lo que debemos proporcionar al menos esta cantidad de memoria al proceso vllm para cargar el modelo.

Tanto las instancias de EC2 G5 de Amazon con GPU A10G como las instancias de EC2 G6 con GPU L4 proporcionan 24 GB de VRAM por GPU, suficiente para cargar pesos de Llama 3.1 8B Instruct. Si va a implementar un modelo con un peso mayor, considere la posibilidad de utilizar un tipo de instancia de varias GPU o una configuración de varios nodos.

Controladores para dispositivos de NVIDIA

Los controladores de NVIDIA proporcionan el entorno de tiempo de ejecución necesario para que los contenedores accedan a los recursos de GPU de forma eficiente. Permite la asignación y administración de recursos de GPU en Kubernetes, lo que permite que las GPU estén disponibles como recursos programables.

Nuestro clúster utiliza AMI de Bottlerocket de EKS, que incluyen todos los complementos y controladores para dispositivos de NVIDIA necesarios en todos los nodos con GPU, lo que garantiza una accesibilidad inmediata a la GPU para las cargas de trabajo en contenedores sin necesidad de configuración adicional. Si utiliza otros tipos de nodos de EKS, debe asegurarse de que estén instalados todos los controladores y complementos necesarios.

Prueba de la infraestructura de GPU

Para probar las capacidades de GPU del clúster, siga los pasos que se indican a continuación a fin de garantizar que los pods puedan acceder a los recursos de GPU de NVIDIA y programarlos correctamente en nodos habilitados para GPU.

Implemente un pod de prueba de SMI de NVIDIA:

```
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
```

Consulte los registros del pod para comprobar que se muestren los detalles de GPU, de forma similar a la salida siguiente (no necesariamente el mismo modelo de GPU):

```
$ 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
   |
+---------------------------------------------------------------------------------------
+
```

Este resultado muestra que los pods pueden acceder correctamente a los recursos de GPU.

IMPORTANTE: Este pod utiliza una configuración de nodeSelector que se alinea con los grupos de nodos de Karpenter en the section called “Creación de un clúster”. Si utiliza grupos de nodos diferentes, asegúrese de que el pod coincida con nodeSelector y las tolerancias en consecuencia.

Implementación de un contenedor de inferencia

La pila de distribución determina las capacidades de rendimiento y escalabilidad de la infraestructura de inferencia. vLLM se ha convertido en una solución líder para las implementaciones de producción. La arquitectura de vLLM proporciona un procesamiento continuo por lotes para el procesamiento dinámico de solicitudes, optimizaciones del kernel para una inferencia más rápida y una administración eficiente de la memoria de GPU a través de PagedAttention. Estas características, combinadas con una API de REST lista para la producción y la compatibilidad con los formatos de modelos más populares, la convierten en una opción óptima para las implementaciones de inferencias de alto rendimiento.

Selección de la imagen de contenedor de aprendizaje profundo de AWS

Los contenedores de aprendizaje profundo de AWS (DLC) proporcionan entornos optimizados previamente con actualizaciones de seguridad, compatibilidad con infraestructuras de AWS y configuraciones de controladores optimizadas. De este modo, se reduce la complejidad de la implementación y los gastos generales de mantenimiento mientras se garantiza la preparación para la producción.

Para esta implementación, utilizaremos el DLC de AWS para vLLM 0.9, que incluye bibliotecas de NVIDIA y configuraciones de rendimiento de GPU optimizadas diseñadas específicamente para la inferencia de modelos de transformadores en instancias de GPU de AWS.

```
image: 763104351884.dkr.ecr.us-east-1.amazonaws.com/vllm:0.9-gpu-py312-ec2
```

Aplicación de los manifiestos de Kubernetes de vLLM

Se puede implementar vLLM de varias formas en EKS. En esta guía, se muestra la implementación de vLLM mediante una implementación de Kubernetes, que es una forma sencilla y nativa de Kubernetes para comenzar. Para ver las opciones de implementación avanzadas, consulte los documentos de vLLM y los esquemas de IA en EKS.

Defina los parámetros de implementación mediante los manifiestos de Kubernetes para controlar la asignación de recursos, la ubicación de los nodos, los sondeos de estado, la exposición del servicio, etc. Configure su implementación para ejecutar un pod habilitado para GPU mediante la imagen del contenedor de aprendizaje profundo de AWS para vLLM. Establezca parámetros optimizados para la inferencia de LLM y exponga el punto de conexión compatible con OpenAPI de vLLM mediante el servicio del equilibrador de carga de 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
```

Compruebe que el estado del pod de vLLM sea `Ready 1/1`:

```
kubectl get pod -l app=vllm-inference-app -w
```

Resultado previsto:

```
NAME                                 READY   UP-TO-DATE   AVAILABLE   AGE
vllm-inference-app-65df5fddc8-5kmjm   1/1     1            1           5m
```

Es posible que pasen varios minutos hasta que se extraiga la imagen del contenedor y vLLM cargue los archivos del modelo en la memoria de GPU. Continúe solo cuando el pod esté listo y disponible.

Exposición del servicio

Exponga el punto de conexión de inferencia de forma local mediante el reenvío de puertos de Kubernetes para su desarrollo y pruebas locales. Deje este comando en ejecución en una ventana de terminal independiente:

```
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
```

El controlador del equilibrador de carga de AWS crea automáticamente un equilibrador de carga de red que expone externamente el punto de conexión del servicio de vLLM. Para obtener el punto de conexión del NLB, ejecute:

```
NLB=$(kubectl get service vllm-inference-svc -o
 jsonpath='{.status.loadBalancer.ingress[0].hostname}')
```

¿Debe instalar el controlador del equilibrador de carga de AWS? Siga los pasos de implementación de the section called “ AWS Controlador del equilibrador de carga de ”.

Ejecutar una inferencia

Validación del pod de inferencia

Valide la funcionalidad del contenedor de inferencias de forma local a través del puerto reenviado. Envíe una solicitud de conexión y asegúrese de que la respuesta incluya el código HTTP 200:

```
$ 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
```

Para probar las capacidades de inferencia y validar la conectividad externa, envíe una solicitud de finalización al LLM a través del punto de conexión del NLB:

```
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
  }'
```

Este punto de conexión sigue el formato de la API de OpenAI, lo que lo hace compatible con las aplicaciones existentes y, al mismo tiempo, proporciona parámetros de generación configurables, como la longitud de respuesta y la temperatura, para controlar la diversidad de la salida.

Ejecución de la aplicación de chatbot

A modo de demostración, en esta guía se ejecuta un ejemplo de aplicación de chatbot con el proyecto nextjs-vllm-ui para mostrar las interacciones de los usuarios con el modelo.

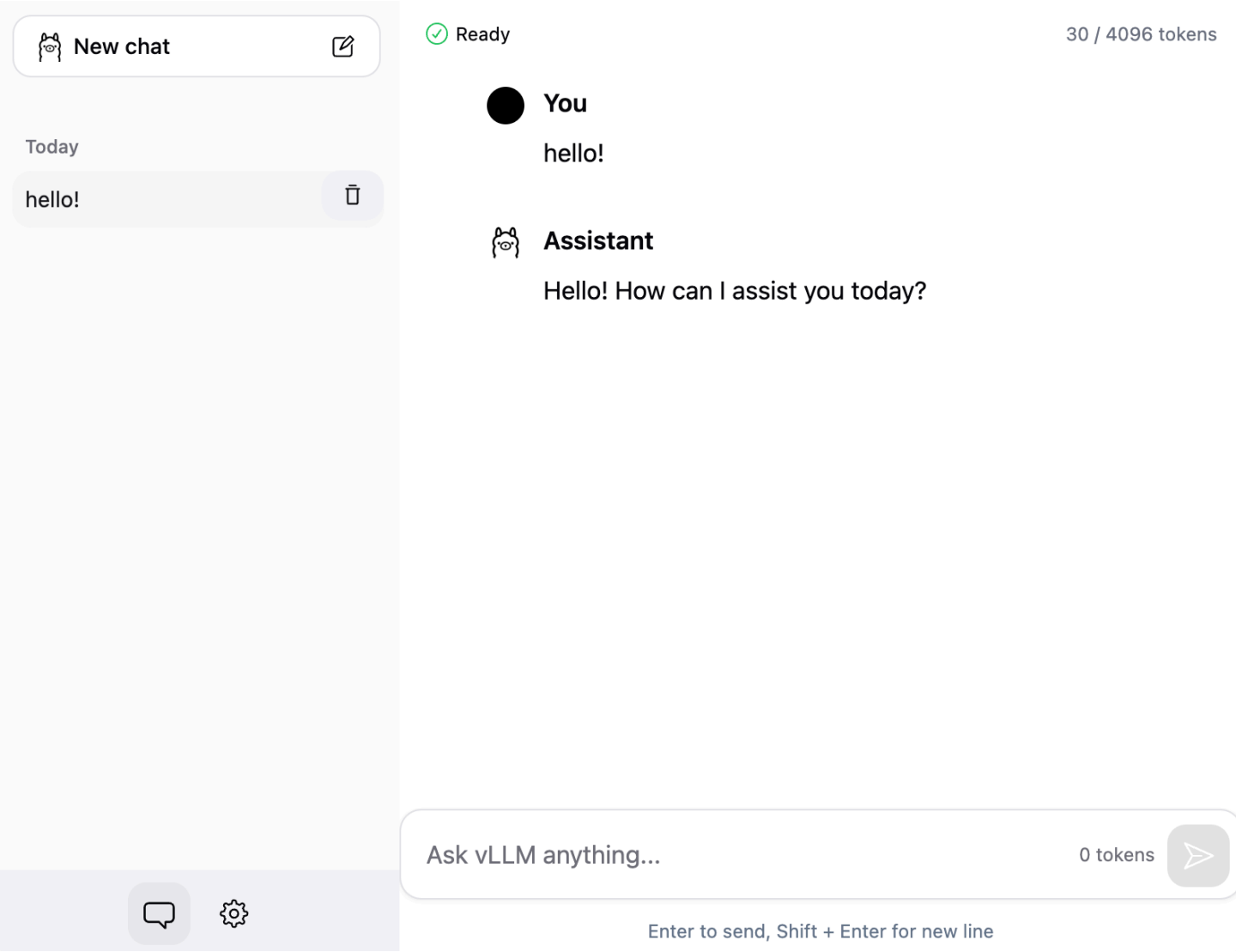
Ejecute la interfaz de usuario del chatbot como un contenedor de Docker que asigne el puerto 3000 a localhost y se conecte al punto de conexión del NLB de vLLM:

```
docker run --rm \
  -p 3000:3000 \
  -e VLLM_URL="http://${NLB}:80" \
  --name nextjs-vllm-ui-demo \
  ghcr.io/yoziru/nextjs-vllm-ui:latest
```

Abra el navegador web y vaya a http://localhost:3000/

Debería ver la interfaz de chat en la que puede interactuar con el modelo Llama.

Interfaz de usuario de chat



Optimización del rendimiento de inferencia

Los motores de inferencia especializados, como vLLM, proporcionan características avanzadas que aumentan considerablemente el rendimiento de inferencia, como el procesamiento continuo por lotes, el almacenamiento en caché de KV eficiente y los mecanismos optimizados de atención de la memoria. Puede ajustar los parámetros de configuración de vLLM para mejorar el rendimiento de inferencia y, al mismo tiempo, cumplir con los requisitos específicos de sus casos de uso y sus patrones de carga de trabajo. Una configuración adecuada es esencial para lograr la saturación de la GPU, ya que permite aprovechar al máximo los costosos recursos de GPU y, al mismo tiempo, ofrecer un alto rendimiento, baja latencia y operaciones rentables. Las siguientes optimizaciones lo ayudarán a maximizar el rendimiento de su implementación de vLLM en EKS.

Configuraciones de VLLM de referencia

Para ajustar los parámetros de configuración de vLLM para su caso de uso, compare diferentes configuraciones mediante una herramienta integral de comparación de inferencias como GuideLLM. De este modo, se recopilarán métricas clave, como el rendimiento de solicitudes por segundo (RPS), la latencia de extremo a extremo (E2E), el tiempo transcurrido hasta el primer token (TTFT) y la latencia de cola (TPOT) para comparar diferentes configuraciones.

Configuración de vLLM de línea de base

Esta es la configuración de línea de base que se utilizó para ejecutar vLLM:

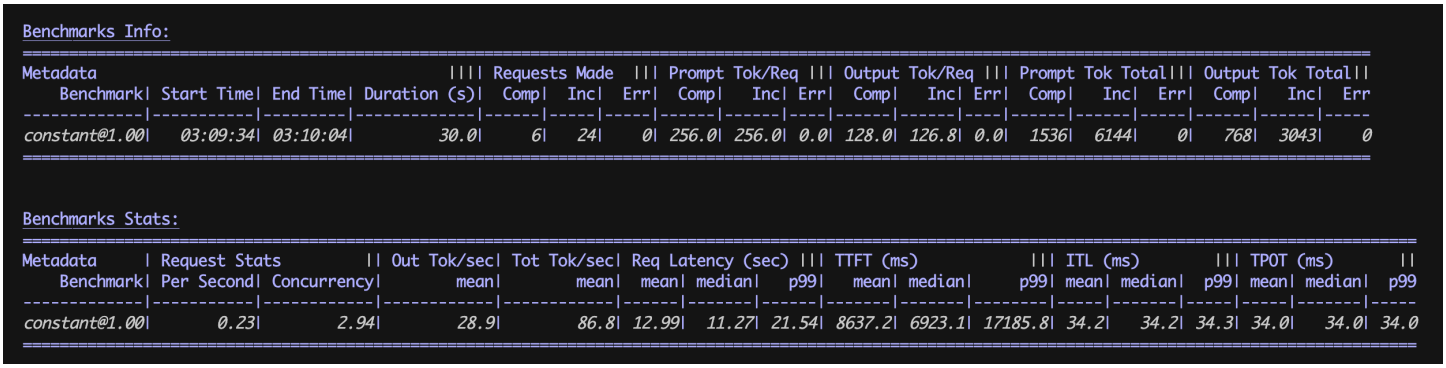
Parámetro de vLLM	Descripción
tensor_parallel_size: 1	Distribución del modelo en 1 GPU
gpu_memory_utilization: 0,90	Reserva de un 10 % de memoria de GPU para cubrir la sobrecarga del sistema
max_sequence_length: 8192	Longitud máxima total de la secuencia (entrada + salida)
max_num_seqs: 1	Número máximo de solicitudes simultáneas por GPU (procesamiento por lotes)

Ejecute GuideLLM con esta configuración básica para establecer una línea de base de rendimiento. Para esta prueba, GuideLLM está configurado para generar 1 solicitud por segundo, con solicitudes de 256 tokens y respuestas de 128 tokens.

```
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"
```

Resultado previsto:

Resultados de la referencia de línea de base



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

Configuración de VLLM ajustada

Ajuste los parámetros de vLLM para utilizar mejor los recursos de GPU y la paralelización:

Parámetro de vLLM	Descripción
tensor_parallel_size: 1	Manténgalo en 1 GPU. La paralelización del tensor debe coincidir con la cantidad de GPU que utilizará vLLM.
gpu_memory_utilization: 0,92	Reduzca la sobrecarga de memoria de GPU si es posible y, al mismo tiempo, asegúrese de que vLLM continúe ejecutándose sin errores.
max_sequence_length: 4096	Ajuste la secuencia máxima según los requisitos de su caso de uso: si se reduce la secuencia máxima, se liberan recursos que se pueden utilizar para aumentar la paralelización.
max_num_seqs: 8	El aumento del máximo de secuencias aumenta el rendimien to, pero también aumenta la latencia. Aumente este valor para maximizar el rendimiento y, al mismo tiempo, garantizar que la latencia se mantenga dentro de los requisitos de su caso de uso.

Aplique estos cambios a la implementación en ejecución mediante el comando patch de kubectl:

```
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"}
]'
```

Compruebe que el estado del pod de vLLM sea `Ready 1/1`:

```
kubectl get pod -l app=vllm-inference-app -w
```

Resultado previsto:

```
NAME                                    READY   UP-TO-DATE   AVAILABLE   AGE
vllm-inference-app-65df5fddc8-5kmjm   1/1     1            1           5m
```

A continuación, ejecute GuideLLM de nuevo con los mismos valores de referencia que antes:

```
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"
```

Resultado previsto:

Resultados de la referencia optimizados

```
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|   03:20:01| 03:20:31|         30.0|    25|    5|    0| 256.1| 256.0| 0.0| 128.0| 73.6| 0.0|  6403|  1280|    0|   3200|   368|     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|       0.86|        4.45|      109.6|       328.9|  5.19|    5.20|  5.23| 147.9|  148.4| 169.4| 39.7|   39.8| 39.8| 39.4|   39.5| 39.5
═══════════════════════════════════════════════════════════════════════════════════════════════════════════════════════════════
```

Resultados de la referencia

Calcule los resultados de la referencia en una tabla para la configuración de vLLM optimizada y de línea de base:

Valores medios	Configuración de línea de base	Configuración optimizada
RPS	0,23 solicitudes/s	0,86 solicitudes/s
E2E	12,99 s	5,19 s
TTFT	8637,2 ms	147,9 ms
TPOT	34,0 ms	39,5 ms

Las configuraciones de vLLM optimizadas mejoraron significativamente el rendimiento de inferencia (RPS) y redujeron la latencia (E2E, TTFT) con solo un pequeño aumento de milisegundos en la latencia de cola (TPOT). Estos resultados demuestran cómo vLLM mejora considerablemente el rendimiento de inferencia, ya que permite que cada contenedor procese más solicitudes en menos tiempo, lo que supone un funcionamiento rentable.

Configuración del clúster de Amazon EKS para cargas de trabajo de IA/ML

Esta sección está diseñada para ayudarlo a configurar los clústeres de Amazon EKS optimizados para cargas de trabajo de IA/ML. Encontrará instrucciones sobre cómo ejecutar contenedores acelerados por GPU utilizando AMI optimizadas para Linux y Windows, cómo configurar clústeres de entrenamiento con Elastic Fabric Adapter (EFA) para redes de alto rendimiento y cómo crear clústeres de inferencia con instancias de AWS Inferentia, incluidos los requisitos previos, los procedimientos paso a paso y las consideraciones de implementación.

Temas

- Uso de AMI aceleradas optimizadas para EKS para instancias de GPU
- Instalación del complemento para dispositivos de Kubernetes
- Ejecución de contenedores acelerados por GPU (Windows en EC2 G-Series)
- Impartición de formación en machine learning en Amazon EKS con Elastic Fabric Adapter
- Uso de instancias de AWS Inferentia con Amazon EKS para machine learning

Uso de AMI aceleradas optimizadas para EKS para instancias de GPU

Amazon EKS admite las AMI de Amazon Linux y Bottlerocket optimizadas para EKS para instancias de GPU. Las AMI aceleradas optimizadas para EKS simplifican la ejecución de cargas de trabajo de IA y ML en los clústeres de EKS al proporcionar imágenes del sistema operativo validadas y prediseñadas para la pila de Kubernetes acelerada. Además de los componentes principales de Kubernetes que se incluyen en las AMI estándar optimizadas para EKS, las AMI aceleradas optimizadas para EKS incluyen los módulos de kernel y los controladores necesarios para ejecutar las instancias de EC2 G y P de GPU de NVIDIA y las instancias de EC2 de GPU de AWS Inferentia y Trainium en los clústeres de EKS.

En la siguiente tabla, se muestran los tipos de instancias de GPU compatibles para cada variante de AMI acelerada optimizada para EKS. Consulte las versiones de AL2023 y las versiones de Bottlerocket optimizadas para EKS en GitHub para ver las actualizaciones más recientes de las variantes de AMI.

Variante de AMI de EKS	Tipos de instancias de EC2
AL2023 x86_64 de NVIDIA	p6-b300, p6-b200, p5, p5e, p5en, p4d, p4de, p3, p3dn, gr6, g6, g6e, g6f, gr6f, g5, g4dn
ARM de AL2023 de NVIDIA	p6e-gb200, g5g
AL2023 x86_64 de Neuron	inf1, inf2, trn1, trn2
Bottlerocket x86_64 aws-k8s-nvidia	p6-b300, p6-b200, p5, p5e, p5en, p4d, p4de, p3, p3dn, gr6, g6, g6e, g6f, gr6f, g5, g4dn
Bottlerocket aarch64/arm64 aws-k8s-nvidia	g5g
Bottlerocket x86_64 aws-k8s	inf1, inf2, trn1, trn2

AMI de NVIDIA optimizadas para EKS

Al utilizar las AMI de NVIDIA optimizadas para EKS, acepta el Contrato de licencia de usuario final (EULA) de NVIDIA para la nube.

Para encontrar las AMI de NVIDIA optimizadas para EKS más recientes, consulte the section called “Obtención de los ID más recientes” y the section called “Obtención de los ID más recientes”.

Si utiliza Amazon Elastic Fabric Adaptor (EFA) con las AMI de NVIDIA de AL2023 o Bottlerocket optimizadas para EKS, debe instalar el complemento para dispositivos de EFA por separado. Para obtener más información, consulte the section called “Configuración del entrenamiento de clústeres con EFA”.

AMI de NVIDIA de AL2023 de EKS

Cuando utilice el operador de GPU de NVIDIA con las AMI de NVIDIA de AL2023 optimizadas para EKS, debe desactivar la instalación del controlador y el kit de herramientas por parte del operador, ya que se incluyen en las AMI de EKS. Las AMI de NVIDIA de AL2023 optimizadas para EKS no incluyen el complemento para dispositivos de Kubernetes de NVIDIA ni el controlador NVIDIA de DRA, que deben instalarse por separado. Para obtener más información, consulte the section called “Instalación del complemento para dispositivos de Kubernetes de NVIDIA”.

Además de los componentes estándar de AMI de EKS, las AMI de NVIDIA de AL2023 optimizadas para EKS incluyen los siguientes componentes.

- Controlador de NVIDIA
- Controlador de modo de usuario CUDA de NVIDIA
- Kit de herramientas de contenedores de NVIDIA
- NVIDIA Fabric Manager
- NVIDIA con persistencia
- Controlador de IMEX de NVIDIA
- Administrador de subredes NVLink de NVIDIA
- EFA mínimo (módulo de kernel y rdma-core)

Para obtener más información sobre el controlador de modo de usuario CUDA de NVIDIA y el tiempo de ejecución y las bibliotecas de CUDA que se utilizan en los contenedores de aplicaciones, consulte la documentación de NVIDIA. La versión de CUDA que se muestra en `nvidia-smi` es la versión del controlador de modo de usuario CUDA de NVIDIA instalada en el host, que debe ser compatible con el tiempo de ejecución y las bibliotecas de CUDA que se utilizan en los contenedores de aplicaciones.

Las AMI de NVIDIA de AL2023 optimizadas para EKS admiten el kernel 6.12 para las versiones 1.33 y posteriores de Kubernetes, y la versión 580 del controlador de NVIDIA para todas las versiones de Kubernetes. Se requiere el controlador de NVIDIA 580 para utilizar CUDA 13+.

Consulte las versiones de AL2023 optimizadas para EKS en GitHub para obtener información sobre las versiones de los componentes incluidas en las AMI. Consulte el script de instalación de la AMI de NVIDIA de AL2023 de EKS y el script de carga del kernel para obtener información sobre cómo las AMI de EKS configuran las dependencias de NVIDIA. Puede encontrar la lista de paquetes instalados y sus versiones en una instancia de EC2 en ejecución con el comando `dnf list installed`.

Al crear AMI personalizadas con las AMI optimizadas para EKS como base, no se recomienda ni se admite ejecutar una actualización del sistema operativo (por ejemplo, `dnf upgrade`) ni actualizar cualquiera de los paquetes de Kubernetes o GPU que se incluyen en las AMI optimizadas para EKS, ya que se corre el riesgo de interrumpir la compatibilidad de los componentes. Si actualiza el sistema operativo o los paquetes que se incluyen en las AMI optimizadas para EKS, se recomienda llevar a cabo pruebas exhaustivas en un entorno de desarrollo o ensayo antes de implementarlas en producción.

Al crear AMI personalizadas para instancias de GPU, se recomienda crear AMI personalizadas independientes para cada familia y generación del tipo de instancia que vaya a ejecutar. Las AMI aceleradas optimizadas para EKS instalan de forma selectiva los controladores y paquetes en tiempo de ejecución en función de la familia y generación del tipo de instancia subyacentes. Para obtener más información, consulte los scripts de AMI de EKS para la instalación y el tiempo de ejecución.

AMI de NVIDIA de Bottlerocket de EKS

Cuando utilice el operador de GPU de NVIDIA con las AMI de NVIDIA de Bottlerocket optimizadas para EKS, debe desactivar la instalación del controlador, el kit de herramientas y el complemento para dispositivos por parte del operador, ya que se incluyen en las AMI de EKS.

Además de los componentes estándar de AMI de EKS, las AMI de NVIDIA de Bottlerocket optimizadas para EKS incluyen los siguientes componentes. Las dependencias mínimas de EFA (módulo del kernel y rdma-core) están instaladas en todas las variantes de Bottlerocket.

- Complemento para dispositivos de Kubernetes de NVIDIA
- Controlador de NVIDIA
- Controlador de modo de usuario CUDA de NVIDIA

- Kit de herramientas de contenedores de NVIDIA
- NVIDIA Fabric Manager
- NVIDIA con persistencia
- Controlador de IMEX de NVIDIA
- Administrador de subredes NVLink de NVIDIA
- MIG Manager de NVIDIA

Para obtener más información sobre el controlador de modo de usuario CUDA de NVIDIA y el tiempo de ejecución y las bibliotecas de CUDA que se utilizan en los contenedores de aplicaciones, consulte la documentación de NVIDIA. La versión de CUDA que se muestra en `nvidia-smi` es la versión del controlador de modo de usuario CUDA de NVIDIA instalada en el host, que debe ser compatible con el tiempo de ejecución y las bibliotecas de CUDA que se utilizan en los contenedores de aplicaciones.

Consulte la información sobre la versión de Bottlerocket en la documentación de Bottlerocket para obtener detalles sobre los paquetes instalados y sus versiones. Las AMI de NVIDIA de Bottlerocket optimizadas para EKS admiten el kernel 6.12 para las versiones 1.33 y posteriores de Kubernetes, y la versión 580 del controlador de NVIDIA para las versiones 1.34 y posteriores de Kubernetes. Se requiere el controlador de NVIDIA 580 para utilizar CUDA 13+.

AMI de Neuron optimizadas para EKS

Para obtener información sobre cómo ejecutar cargas de trabajo de entrenamiento e inferencia que utilizan Neuron con Amazon EKS, consulte las siguientes referencias:

- Contenedores - Kubernetes - Introducción en la documentación de AWS Neuron
- Training example en las muestras de EKS de AWS Neuron en GitHub
- Implementación de cargas de trabajo de inferencia de ML con Inferentia en Amazon EKS

Para encontrar las AMI de Neuron optimizadas para EKS más recientes, consulte the section called “Obtención de los ID más recientes” y the section called “Obtención de los ID más recientes”.

Si utiliza Amazon Elastic Fabric Adaptor (EFA) con las AMI de Neuron de AL2023 o Bottlerocket optimizadas para EKS, debe instalar el complemento para dispositivos de EFA por separado. Para obtener más información, consulte the section called “Configuración del entrenamiento de clústeres con EFA”.

AMI de Neuron de AL2023 de EKS

Las AMI de Neuron de AL2023 optimizadas para EKS no incluyen el complemento para dispositivos de Kubernetes de Neuron ni la extensión del programador de Kubernetes de Neuron, que deben instalarse por separado. Para obtener más información, consulte the section called “Instalación del complemento para dispositivos de Kubernetes de Neuron”.

Además de los componentes estándar de AMI de EKS, las AMI de Neuron de AL2023 optimizadas para EKS incluyen los siguientes componentes.

- Controlador de Neuron (aws-neuronx-dkms)
- Herramientas de Neuron (aws-neuronx-tools)
- EFA mínimo (módulo de kernel y rdma-core)

Consulte el script de instalación de la AMI de Neuron de AL2023 de EKS para obtener información sobre cómo las AMI de EKS configuran las dependencias de Neuron. Consulte las versiones de AL2023 optimizadas para EKS en GitHub para ver las versiones de los componentes incluidas en las AMI. Puede encontrar la lista de paquetes instalados y sus versiones en una instancia de EC2 en ejecución con el comando `dnf list installed`.

AMI de Neuron de Bottlerocket de EKS

Las variantes estándar de Bottlerocket (aws-k8s) incluyen las dependencias de Neuron, que se detectan y cargan automáticamente cuando se ejecutan en instancias de EC2 de AWS Inferentia o Trainium.

Las AMI de Bottlerocket optimizadas para EKS no incluyen el complemento para dispositivos de Kubernetes de Neuron ni la extensión del programador de Kubernetes de Neuron, que deben instalarse por separado. Para obtener más información, consulte the section called “Instalación del complemento para dispositivos de Kubernetes de Neuron”.

Además de los componentes estándar de AMI de EKS, las AMI de Neuron de Bottlerocket optimizadas para EKS incluyen los siguientes componentes.

- Controlador de Neuron (aws-neuronx-dkms)
- EFA mínimo (módulo de kernel y rdma-core)

Al utilizar las AMI de Bottlerocket optimizadas para EKS con instancias de Neuron, se debe configurar lo siguiente en los datos del usuario de Bottlerocket. Esta configuración permite que el contenedor obtenga la propiedad del dispositivo de Neuron montado en función de los valores `runAsUser` y `runAsGroup` proporcionados en la especificación de la carga de trabajo. Para obtener más información sobre la compatibilidad de Neuron con Bottlerocket, consulte Quickstart on EKS readme en GitHub.

```
[settings]
[settings.kubernetes]
device-ownership-from-security-context = true
```

Consulte el registro de cambios del kit de kernel de Bottlerocket para obtener información sobre la versión del controlador de Neuron incluida en las AMI de Bottlerocket optimizadas para EKS.

Instalación del complemento para dispositivos de Kubernetes

Los complementos para dispositivos de Kubernetes han sido el principal mecanismo para anunciar infraestructuras especializadas (como GPU, interfaces de red y adaptadores de red) como recursos consumibles para las cargas de trabajo de Kubernetes. Si bien la asignación dinámica de recursos (DRA) se posiciona como el futuro de la administración de dispositivos en Kubernetes, la mayoría de los proveedores de infraestructura especializados dan soporte a los controladores de DRA desde una fase temprana. Los complementos para dispositivos de Kubernetes siguen siendo un enfoque ampliamente disponible para usar las GPU en los clústeres de Kubernetes actualmente.

Consideraciones

- Al utilizar las AMI de AL2023 optimizadas para EKS con las GPU de NVIDIA, debe instalar el complemento para dispositivos de Kubernetes de NVIDIA. Puede instalar y administrar el complemento para dispositivos de Kubernetes de NVIDIA con Helm, con las herramientas de Kubernetes que prefiera o con el operador de GPU de NVIDIA.
- Al utilizar las AMI de Bottlerocket optimizadas para EKS con las GPU de NVIDIA, no es necesario instalar el complemento para dispositivos de Kubernetes de NVIDIA, ya que se incluye en las AMI de Bottlerocket optimizadas para EKS. Esto incluye los casos en los que utiliza instancias de GPU con el modo automático de EKS.
- Si utiliza las AMI de AL2023 o Bottlerocket optimizadas para EKS con las GPU de AWS Inferentia o Trainium, debe instalar el complemento para dispositivos de Kubernetes de Neuron y, opcionalmente, instalar la extensión del programador de Kubernetes de Neuron. Para obtener más información, consulte la documentación de Neuron para ejecuciones en EKS.

Instalación del complemento para dispositivos de Kubernetes de NVIDIA

En el siguiente procedimiento, se describe cómo instalar el complemento para dispositivos de Kubernetes de NVIDIA y cómo ejecutar una prueba de muestra en instancias de GPU de NVIDIA.

Requisitos previos

- Clúster de EKS ya creado
- Nodos de GPU de NVIDIA en ejecución en el clúster con la AMI de NVIDIA de AL2023 optimizada para EKS
- Si tiene Helm instalado en su entorno de línea de comandos, consulte las instrucciones de configuración de Helm.

Procedimiento

1. Agregue el repositorio de gráficos de Helm `nvdp`.

```
helm repo add nvdp https://nvidia.github.io/k8s-device-plugin
```

2. Actualice el repositorio de Helm local para asegurarse de que cuenta con los gráficos más recientes.

```
helm repo update
```

3. Obtenga la versión más reciente del complemento para dispositivos de Kubernetes de NVIDIA.

```
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. Instale el complemento para dispositivos de Kubernetes de NVIDIA en el clúster y sustituya `0.17.4` por la versión más reciente del comando anterior.

```
helm install nvdp nvdp/nvidia-device-plugin \
  --namespace nvidia \
  --create-namespace \
  --version 0.17.4 \
```

```
  --set gfd.enabled=true
```

5. Compruebe que el complemento para dispositivos de Kubernetes de NVIDIA se esté ejecutando en el clúster. A continuación, se muestra el resultado con dos nodos en el clúster.

```
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. Compruebe que los nodos tengan GPU asignables con el siguiente comando.

```
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. Cree un archivo denominado `nvidia-smi.yaml` con el siguiente contenido. Este manifiesto lanza una imagen de contenedor de AL2023 mínima que ejecuta `nvidia-smi` en un nodo.

```
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. Aplique el manifiesto con el siguiente comando.

```
kubectl apply -f nvidia-smi.yaml
```

9. Una vez que el pod termine de ejecutarse, consulte sus registros con el siguiente comando.

```
kubectl logs nvidia-smi
```

Un ejemplo de salida sería el siguiente.

```
+-----------------------------------------------------------------------------------------
+
| 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
     |
+---------------------------------------------------------------------------------------------
+
```

Instalación del complemento para dispositivos de Kubernetes de Neuron

En el siguiente procedimiento, se describe cómo instalar el complemento para dispositivos de Kubernetes de Neuron y cómo ejecutar una prueba de muestra en una instancia de Inferentia.

Requisitos previos

- Clúster de EKS ya creado
- Nodos de GPU de Neuron que se ejecutan en el clúster mediante la AMI de Neuron de AL2023 optimizada para EKS o la AMI de Bottlerocket
- Si tiene Helm instalado en su entorno de línea de comandos, consulte las instrucciones de configuración de Helm.

Procedimiento

1. Instale el complemento para dispositivos de Kubernetes de Neuron en el clúster.

```
helm upgrade --install neuron-helm-chart oci://public.ecr.aws/neuron/neuron-helm-
chart \
    --set "npd.enabled=false"
```

2. Compruebe que el complemento para dispositivos de Kubernetes de Neuron se esté ejecutando en el clúster. A continuación, se muestra el resultado con un único nodo de Neuron en el clúster.

```
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. Compruebe que los nodos tengan NeuronCores asignables con el siguiente comando.

```
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. Compruebe que los nodos tengan NeuronDevices asignables con el siguiente comando.

```
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. Cree un archivo denominado `neuron-ls.yaml` con el siguiente contenido. Este manifiesto lanza un contenedor de Neuron Monitor que tiene la herramienta `neuron-ls` instalada.

```
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. Aplique el manifiesto con el siguiente comando.

```
kubectl apply -f neuron-ls.yaml
```

7. Una vez que el pod termine de ejecutarse, consulte sus registros con el siguiente comando.

```
kubectl logs neuron-ls
```

A continuación se muestra un ejemplo de salida.

```
instance-type: inf2.xlarge
instance-id: ...
+--------+--------+--------+---------+
| NEURON | NEURON | NEURON |   PCI   |
| DEVICE | CORES  | MEMORY |   BDF   |
+--------+--------+--------+---------+
| 0      | 2      | 32 GB  | 00:1f.0 |
+--------+--------+--------+---------+
```

Ejecución de contenedores acelerados por GPU (Windows en EC2 G-Series)

> **Important**
>
> El complemento de dispositivo de Kubernetes para DirectX, desarrollado por TensorWorks, es una herramienta de terceros que no cuenta con el respaldo, soporte ni mantenimiento de AWS. AWS no asume ninguna responsabilidad por la seguridad, la confiabilidad o el rendimiento de este complemento.

Aprenda a ejecutar cargas de trabajo de contenedores de Windows aceleradas por GPU en Amazon EKS (Elastic Kubernetes Service) mediante GPU NVIDIA con el complemento de dispositivo de Kubernetes para DirectX desarrollado por TensorWorks. Para obtener más información, consulte Complemento de dispositivo de Kubernetes para DirectX.

Existen dos enfoques principales para configurar la aceleración de la GPU para los contenedores de Windows:

- Opción 1: cree una AMI optimizada para Windows EKS personalizada con los controladores de GPU necesarios preinstalados.
  - Utilice este enfoque cuando necesite un entorno consistente y preconfigurado listo para ejecutar contenedores de Windows acelerados por GPU, y esté dispuesto a invertir el esfuerzo adicional necesario para crear y mantener la AMI personalizada.
- Opción 2: instale los controladores de GPU necesarios en los nodos de trabajo de EKS después de inicializar la instancia.
  - Utilice este enfoque cuando prefiera un proceso de configuración más sencillo y no tenga inconveniente en instalar los controladores de GPU en cada nuevo nodo de trabajo. Más adecuado para un entorno de desarrollo cuando se evalúan o crean prototipos de cargas de trabajo aceleradas por la GPU.

Puede aprovechar ambos enfoques según los pasos que se detallan en esta guía.

Consideraciones

Esta guía proporciona los pasos para instalar y configurar la aceleración por GPU para los contenedores de Windows mediante GPU NVIDIA, controladores NVIDIA GRID y el complemento de dispositivo de Kubernetes para DirectX desarrollado por TensorWorks. Los pasos han sido probados y verificados para proporcionar aceleración por GPU a las cargas de trabajo de contenedores de Windows en Amazon EKS. Consulte the section called “Limitaciones conocidas” para obtener más información sobre los controladores y complementos de dispositivo compatibles. Antes de continuar, tenga en cuenta lo siguiente:

- Únicamente se han probado y verificado los tipos de instancias de la familia G con controladores NVIDIA GRID para garantizar su funcionamiento con esta guía. Si bien otros tipos de instancias y combinaciones de controladores también pueden ejecutar contenedores de Windows acelerados por la GPU, es posible que requieran pasos de configuración adicionales que no se describen en esta guía.
- Únicamente se han probado y verificado las cargas de trabajo basadas en DirectX para garantizar su funcionamiento con esta guía. Aunque otras API de GPU como OpenGL, Vulkan y OpenCL podrían ser compatibles para ejecutar contenedores de Windows acelerados por GPU, es posible que requieran pasos de configuración adicionales que no se abordan en esta guía.
- Existen algunas limitaciones conocidas que se deben tener en cuenta antes de ejecutar contenedores de Windows acelerados por la GPU. Consulte la sección the section called “Limitaciones conocidas” para obtener más información.

Requisitos previos

Para habilitar la aceleración por GPU para los contenedores de Windows en Amazon EKS, debe cumplir los siguientes requisitos antes de continuar:

- Inicie un clúster de Amazon EKS con la versión 1.27 o posterior de Kubernetes.
- Aprovisione nodos Windows con Windows Server 2022 o posterior.
- Aprovisione nodos de Windows en la familia G de tipos de instancias, como G4 o G5.
- Aprovisione los nodos de Windows con un tiempo de ejecución de contenedor con containerd `1.7.x` o `2.x.x`. (Consulte the section called “Obtención de información de la versión” para verificar la versión de containerd en la AMI optimizada de Amazon EKS).

Instale el controlador de la GPU en cada nodo de Windows

Para instalar los controladores NVIDIA GRID en los nodos de trabajo de EKS, siga los pasos que se indican en Controladores NVIDIA para la instancia de Amazon EC2. Vaya a Opciones de instalación - Opción 3: Controladores GRID y siga los pasos de instalación.

Realización de la instalación para Windows Server Core

En el caso de Windows Server Core, que no cuenta con una experiencia de escritorio, instale los controladores NVIDIA GRID de forma silenciosa mediante los siguientes comandos:

```
$nvidiaInstallerFilePath = nvidia-driver-installer.exe # Replace with path to installer
$installerArguments = "-s -clean -noreboot -noeula"
Start-Process -FilePath $nvidiaInstallerFilePath -ArgumentList $installerArguments -
Wait -NoNewWindow -PassThru
```

Comprobación de la instalación

Ejecute el siguiente comando de PowerShell para mostrar información de diagnóstico sobre las GPU de la instancia:

```
nvidia-smi
```

Este comando muestra la versión del controlador NVIDIA, así como información sobre el hardware de la GPU. Asegúrese de que la salida de este comando coincide con la versión del controlador NVIDIA GRID que esperaba instalar.

Implementación del complemento de dispositivo GPU en cada nodo

Para habilitar la detección y exposición de los recursos de GPU a los contenedores en los nodos de Windows, necesitará un complemento de dispositivo. Implemente el complemento de dispositivo DirectX de TensorWorks en cada nodo de trabajo. Para ello, ejecútelo como un DaemonSet en el clúster de EKS. Siga la guía de instalación especificada en el archivo README.md, que incluirá los siguientes pasos. Se recomienda:

- Implementar el complemento de dispositivo en el espacio de nombres `kube-system`.
- Establecer límites de recursos apropiados para el DaemonSet para asegurar que no consume recursos excesivos en los nodos.

> **Note**
>
> El complemento de dispositivo DaemonSet se ejecutará en cada nodo como un contenedor de procesos host con privilegios elevados. Se recomienda implementar controles de RBAC para restringir el acceso a este DaemonSet, de manera que solo los usuarios autorizados puedan ejecutar comandos privilegiados.

Cuando se ejecutan contenedores acelerados por GPU, el complemento de dispositivo admite dos modos:

- Modo de tenencia única: este modo dedica todos los recursos de GPU a un único contenedor en la instancia. Instale los complementos de dispositivo con compatibilidad con tenencia única mediante el siguiente comando. Para obtener más información, consulte el archivo README.md.

```
kubectl apply -f "https://raw.githubusercontent.com/TensorWorks/directx-device-plugins/main/deployments/default-daemonsets.yml"
```

- Modo de tenencia múltiple: este modo permite compartir los recursos de GPU entre varios contenedores en la instancia. Instale los complementos de dispositivo con compatibilidad con tenencia múltiple mediante el siguiente comando. Para obtener más información, consulte el archivo README.md.

```
kubectl apply -f "https://raw.githubusercontent.com/TensorWorks/directx-device-plugins/main/deployments/multitenancy-inline.yml"
```

Como alternativa, utilice un ConfigMap para especificar la tenencia múltiple.

```
kubectl apply -f "https://raw.githubusercontent.com/TensorWorks/directx-device-plugins/main/deployments/multitenancy-configmap.yml"
```

Verificación de la implementación del complemento del dispositivo

Tras implementar el complemento de dispositivo, sustituya `<namespace>` y ejecute el siguiente comando para comprobar que el complemento de dispositivo DirectX se ejecuta correctamente en todos los nodos de Windows.

```
kubectl get ds device-plugin-wddm -n <namespace>
```

Verificación de que los contenedores están listos para su implementación

Una vez que el complemento de dispositivo DaemonSet esté en ejecución en los nodos de trabajo de Windows con GPU, utilice el siguiente comando para verificar que cada nodo disponga de GPU asignables. El número correspondiente debe coincidir con el número de dispositivos DirectX de cada nodo.

```
kubectl get nodes "-o=custom-columns=NAME:.metadata.name,DirectX:.status.allocatable.directx\.microsoft\.com/display"
```

Ejecución de contenedores de Windows con aceleración de GPU

Antes de lanzar los pods, especifique el nombre del recurso `directx.microsoft.com/display` en `.spec.containers[].resources`. Esto indicará que los contenedores requieren capacidades habilitadas para GPU y el `kube-scheduler` intentará colocar los pods en el nodo Windows preconfigurado con recursos de GPU disponibles.

Como ejemplo, consulte el comando de muestra a continuación, que lanza un `Job` para ejecutar una simulación de Monte Carlo y estimar el valor de pi. Este ejemplo proviene del repositorio de GitHub Complementos de dispositivo de Kubernetes para DirectX, el cual incluye múltiples ejemplos que puede elegir y ejecutar para probar las capacidades de GPU de los nodos de Windows.

```
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
```

Limitaciones conocidas

Se pueden utilizar todas las GPU

Todas las GPU de la instancia estarán disponibles para cada contenedor en ejecución en el host, incluso si solicita un número específico de GPU para un contenedor determinado. Además, el comportamiento predeterminado es que todos los contenedores que se ejecutan en el host utilizarán la GPU con el índice 0, incluso si hay varias GPU disponibles en el nodo. Por lo tanto, para que las tareas que utilizan múltiples GPU funcionen correctamente, debe designar explícitamente el dispositivo GPU específico que se utilizará dentro del código de la aplicación.

La implementación exacta para asignar un dispositivo a utilizar en la aplicación dependerá del lenguaje de programación o del marco de trabajo que utilice. Por ejemplo, si utiliza la programación CUDA, para seleccionar una GPU específica, puede especificar explícitamente el dispositivo a utilizar en el código de la aplicación mediante la función cudaSetDevice().

La necesidad de especificar explícitamente el dispositivo se debe a un problema conocido que afecta a los contenedores de Windows. Puede seguir el progreso en la resolución de este problema en el tema #333 de microsoft/Windows-Containers. La siguiente tabla es una representación visual y un ejemplo práctico de este comportamiento de asignación de GPU.

Considere un escenario en el que hay un único nodo de Windows del tipo de instancia `g4dn.12xlarge` de EC2, que viene con cuatro GPU. Considere un escenario en el que se

inicializan tres pods en esta instancia. La tabla muestra que, independientemente de la cantidad de GPU solicitadas por cada contenedor, los tres pods tienen acceso a las cuatro GPU de la instancia y, de forma predeterminada, utilizarán la GPU con el índice de dispositivo 0.

Pod	GPU solicitadas	Acceso real a la GPU	Uso de GPU predeterminado	Índices de GPU disponibles	Total de GPU de la instancia
Pod 1	1 GPU	Las 4 GPU	GPU con índice 0	0, 1, 2, 3	4
Pod 2	2 GPU	Las 4 GPU	GPU con índice 0	0, 1, 2, 3	4
Pod 3	1 GPU	Las 4 GPU	GPU con índice 0	0, 1, 2, 3	4

Compatibilidad con el complemento para dispositivos de Kubernetes

La implementación oficial de NVIDIA del complemento para dispositivos de Kubernetes no es compatible con Windows. Puede seguir el avance de la incorporación de la compatibilidad oficial con Windows en el tema nº 419 de NVIDIA/k8s-device-plugin.

Limitaciones de las instancias de computación de GPU

Según la configuración de la cuenta de AWS, es posible que tenga límites de servicio en cuanto a la cantidad y los tipos de instancias de computación de GPU de Amazon EC2 que puede inicializar. Si necesita capacidad adicional, puede Solicitar un aumento de cuota.

Debe crear una AMI optimizada para la GPU de Windows

No existe una AMI optimizada para GPU de Windows en EKS ni un componente administrado del Generador de imágenes de EC2 proporcionado por Amazon EKS. Deberá seguir los pasos de esta guía para crear una AMI personalizada optimizada para Windows en EKS con los controladores de GPU necesarios preinstalados, o instalar los controladores de GPU requeridos en los nodos de trabajo de EKS después de inicializar las instancias.

No se admiten Inferentia ni Trainium

Las cargas de trabajo basadas en AWS Inferentia y AWS Trainium no son compatibles con Windows.

Impartición de formación en machine learning en Amazon EKS con Elastic Fabric Adapter

En este tema se describe cómo integrar Elastic Fabric Adapter (EFA) con los pods implementados en el clúster de Amazon EKS. Elastic Fabric Adapter (EFA) es una interfaz de red para instancias de Amazon EC2 que le permite ejecutar aplicaciones que requieren altos niveles de comunicaciones entre nodos a escala en AWS. Su interfaz de hardware de bypass del sistema operativo diseñada a medida mejora el rendimiento de las comunicaciones entre instancias, lo que es fundamental para ajustar la escala de estas aplicaciones. Con EFA, las aplicaciones de High Performance Computing (HPC, informática de alto rendimiento) que utilizan la Message Passing Interface (MPI, interfaz de paso de mensajes) y las aplicaciones de Machine Learning (ML) que utilizan NVIDIA Collective Communications Library (NCCL, Biblioteca de comunicación colectiva de NVIDIA) pueden aumentar su escala a miles de CPU o GPU. Como resultado, obtiene el rendimiento de las aplicaciones de los clústeres HPC en las instalaciones con la elasticidad y flexibilidad bajo demanda de la nube de AWS. La integración de Elastic Fabric Adapter (EFA) con aplicaciones que se ejecutan en clústeres de Amazon EKS puede reducir el tiempo necesario para completar cargas de trabajo de formación distribuidas a gran escala sin tener que agregar instancias adicionales al clúster. Para obtener más información sobre EFA, consulte Elastic Fabric Adapter.

Tipos de instancias con EFA

El complemento de dispositivos de Kubernetes para AWS EFA es compatible con todos los tipos de instancias de Amazon EC2 que tienen EFA. Para obtener una lista de los tipos de instancia que tienen EFA, consulte Tipos de instancias admitidos en la Guía del usuario de Amazon EC2. Sin embargo, para ejecutar rápidamente aplicaciones de ML, recomendamos que una instancia tenga chips de aceleración de hardware, como GPU de Nvidia, chips de AWS Inferentia o AWS Trainium, además de EFA. Para ver una lista de los tipos de instancias que tienen chips de aceleración de hardware y EFA, consulte Computación acelerada en la Guía del usuario de Amazon EC2.

Al comparar los tipos de instancias con el fin elegir entre ellos, tenga en cuenta la cantidad de tarjetas de red de EFA disponibles para cada uno, así como la cantidad de tarjetas de acelerador, de CPU y de memoria. Puede asignar hasta un EFA por tarjeta de red. Un EFA cuenta como una interfaz de red. Para ver cuántos EFA están disponibles para cada tipo de instancia que tiene EFA, consulte la lista de tarjetas de red en la Guía del usuario de Amazon EC2.

Interfaces EFA y solo de EFA

Un Elastic Fabric Adapter (EFA) es una interfaz de red que combina las capacidades de un Elastic Network Adapter (ENA) y una interfaz OS-Bypass, con tecnología del protocolo AWS Scalable Reliable Datagram (SRD). Las funcionalidades de EFA permiten que las aplicaciones se comuniquen directamente con el hardware para un transporte de baja latencia. Puede optar por acceder únicamente a las capacidades de EFA mediante interfaces solo de EFA, lo que limita la comunicación a las interfaces que se encuentran dentro de la misma zona de disponibilidad.

Para crear nodos que puedan tener interfaces solo de EFA, debe utilizar una plantilla de lanzamiento de EC2 personalizada y establecer el `InterfaceType` en `efa-only`. En su plantilla de lanzamiento personalizada, no puede configurar la tarjeta de red `0` en una interfaz solo de EFA, ya que es la tarjeta de red principal y la interfaz de red de la instancia de EC2. Debe disponer de la versión `1.18.5` o posterior de VPC CNI para las interfaces exclusivas de EFA. Si utiliza Amazon Linux 2, la versión de la AMI debe ser `v20240928` o posterior para las interfaces exclusivas de EFA.

El siguiente procedimiento lo guiará para crear un clúster de EKS con `eksctl` con nodos que tengan GPU de Nvidia e interfaces de EFA. No se puede utilizar `eksctl` para crear nodos y grupos de nodos que utilicen interfaces solo de EFA.

Requisitos previos

- Un clúster existente de Amazon EKS. Si no tiene ningún clúster existente, cree uno con Introducción. El clúster debe implementarse en una VPC que tenga al menos una subred privada con suficientes direcciones IP disponibles para implementar nodos en ella. La subred privada debe tener acceso a Internet saliente proporcionado por un dispositivo externo, como una puerta de enlace NAT.

  Si planea usar `eksctl` para crear su grupo de nodos, `eksctl` también puede crear un clúster.
- La versión `2.12.3` o posterior, o bien, la versión `1.27.160` o posterior de la AWS interfaz de la línea de comandos (AWS CLI) instalada y configurada en su dispositivo o AWS CloudShell. Para comprobar su versión actual, utilice `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Los administradores de paquetes, como `yum`, `apt-get` o Homebrew para macOS, suelen estar atrasados varias versiones respecto de la versión de la AWS CLI más reciente. Para instalar la versión más reciente, consulte Instalación y Configuración rápida con aws configure en la Guía del usuario de la interfaz de la línea de comandos de AWS. La versión de AWS CLI instalada en AWS CloudShell también puede estar atrasada varias versiones respecto de la versión más reciente.

Para actualizarla, consulte Instalación de la CLI de AWS en su directorio principal en la Guía del usuario de AWS CloudShell.

- La herramienta de línea de comandos de `kubectl` está instalada en su dispositivo o AWS CloudShell. La versión puede ser la misma o hasta una versión secundaria anterior o posterior a la versión de Kubernetes de su clúster. Por ejemplo, si la versión del clúster es `1.29`, puede usar la versión `1.28`, `1.29` o `1.30` de `kubectl` con él. Para instalar o actualizar `kubectl`, consulte the section called “Configure kubectl y eksctl”.
- Debe tener instalada la versión `1.7.10` o una posterior del complemento CNI de Amazon VPC para Kubernetes antes de lanzar nodos de trabajo que admitan varios Elastic Fabric Adapters, como `p4d` o `p5`. Para obtener más información sobre cómo actualizar la versión del complemento CNI de Amazon VPC para Kubernetes, consulte the section called “CNI de Amazon VPC”.
- Para las instancias p6-b200, debe utilizar la versión v0.5.6 o posterior del complemento del dispositivo de EFA.

> **Important**
>
> Una consideración importante que se requiere para adoptar EFA con Kubernetes es configurar y administrar Huge Pages como un recurso en el clúster. Para obtener más información, consulte Administración de Huge Pages en la documentación de Kubernetes. Las instancias de Amazon EC2 con el controlador de EFA instalado preasignan 5128 páginas de gran tamaño de 2 MiB, que puede solicitar como recursos para utilizar en sus especificaciones de trabajo.

Crear un grupo de nodos

El procedimiento siguiente ayuda a crear un grupo de nodos con un grupo de nodos con respaldo de `p4d.24xlarge` con interfaces EFA y RDMA GPUDirect. También lo ayuda a ejecutar una prueba de ejemplo de NVIDIA Collective Communications Library (NCCL) para descubrir el rendimiento NCCL de varios nodos mediante el uso de EFA. El ejemplo se puede utilizar en una plantilla para la formación de aprendizaje profundo distribuida en Amazon EKS mediante EFA.

1. Determine qué tipos de instancias de Amazon EC2 compatibles con EFA están disponibles en la región de AWS en la que desea implementar los nodos. Sustituya *`region-code`* por la región de AWS en la que desea implementar el grupo de nodos.

```
aws ec2 describe-instance-types --region region-code \
    --filters Name=network-info.efa-supported,Values=true \
    --query "InstanceTypes[*].[InstanceType]" --output text
```

Al implementar nodos, el tipo de instancia que desea implementar debe estar disponible en la región de AWS en la que está su clúster.

2. Determine en qué zonas de disponibilidad está disponible el tipo de instancia que desea implementar. En este tutorial, el tipo de instancia `p5.48xlarge` se utiliza y se debe devolver en la salida de la región de AWS que especificó en el paso anterior. Cuando implemente nodos en un clúster de producción, sustituya *`p5.48xlarge`* con cualquier tipo de instancia que se devolvió en el paso anterior.

```
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
```

Un ejemplo de salida sería el siguiente.

```
us-west-2a    us-west-2c    us-west-2b
```

Tenga en cuenta las zonas de disponibilidad devueltas para su uso en pasos posteriores. Al implementar nodos en un clúster, la VPC debe tener subredes con direcciones IP disponibles en una de las zonas de disponibilidad que se muestran en el resultado.

3. Cree un grupo de nodos mediante `eksctl`. Necesita tener la versión `0.215.0` o posterior de la herramienta de línea de comandos `eksctl` instalada en su dispositivo o AWS CloudShell. Para instalar o actualizar `eksctl`, consulte la sección de Instalación en la documentación de `eksctl`.

   a. Copie el siguiente contenido en un archivo denominado *`efa-cluster.yaml`*. Sustituya los valores de ejemplo por sus propios valores. Puede reemplazar `p5.48xlarge` por una instancia diferente, pero si lo hace, asegúrese de que los valores de `availabilityZones` sean zonas de disponibilidad que se devolvieron para el tipo de instancia en el paso uno.

```
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. Cree un grupo de nodos administrados en un clúster existente.

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

Si no dispone de un clúster existente, puede ejecutar el siguiente comando para crear un clúster y el grupo de nodos.

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

> **Note**
>
> Como el tipo de instancia que se usa en este ejemplo tiene GPU, `eksctl` instala automáticamente el complemento del dispositivo de NVIDIA de Kubernetes en cada instancia cuando se usa Amazon Linux 2. Esto no es necesario para Bottlerocket, ya que el complemento del dispositivo NVIDIA está integrado en la variante de NVIDIA de EKS de Bottlerocket. Cuando `efaEnabled` se establece como `true` en la configuración del grupo de nodos, `eksctl` también implementará automáticamente el complemento del dispositivo de EFA en los nodos.

Uso de Bottlerocket con EFA

La versión 1.28.0 y posteriores de la AMI de Bottlerocket incluyen soporte oficial para EFA. Para usar Bottlerocket en nodos compatibles con EFA, especifique `amiFamily: Bottlerocket` en su configuración. Si necesita usar un ID de AMI personalizado, debe usar `nodeGroups` estándar en lugar de `managedNodeGroups`.

A continuación se muestra un ejemplo de configuración:

```
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
```

La configuración de sysctl `vm.nr_hugepages` anterior configura el número de 2 Mi de páginas de gran tamaño. En este ejemplo, 3000 significa 3000 * 2 Mi = 6000 Mi de páginas de gran tamaño.

Verificación de la instalación del complemento del dispositivo de EFA

Al crear un grupo de nodos con `efaEnabled: true`, `eksctl` implementa automáticamente el complemento del dispositivo de EFA de Kubernetes. Puede verificar que el complemento del dispositivo está instalado y funciona correctamente:

1. Compruebe el estado del DaemonSet:

   ```
   kubectl get daemonsets -n kube-system
   ```

   Código de salida de ejemplo:

   ```
   NAME                                  DESIRED   CURRENT   READY   UP-TO-DATE
    AVAILABLE   NODE SELECTOR   AGE
   aws-efa-k8s-device-plugin-daemonset   2         2         2       2            2
         <none>          6m16s
   ...
   ```

   Aquí, el DaemonSet del complemento del dispositivo de EFA se ejecuta en dos nodos. El estado de ambos es LISTO y DISPONIBLE.

2. A continuación, verifique los pods creados por el DaemonSet:

   ```
   kubectl get pods -n kube-system -l name=aws-efa-k8s-device-plugin
   ```

   Código de salida de ejemplo:

   ```
   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
   ```

   El estado de los pods del complemento del dispositivo de EFA es En ejecución, lo que confirma que el complemento se ha implementado y está funcionando correctamente.

3. Verifique el registro de recursos:

   Para confirmar que el recurso `vpc.amazonaws.com/efa` está registrado en el kubelet, puede describir los nodos:

   ```
   kubectl describe nodes
   ```

Si el recurso de EFA está registrado correctamente, lo verá enumerado bajo los recursos Capacidad y Asignable del nodo. Por ejemplo:

```
Capacity:
  ...
  vpc.amazonaws.com/efa:  4
Allocatable:
  ...
  vpc.amazonaws.com/efa:  4
```

Esta salida confirma que el nodo reconoce el recurso de EFA, que se pone a disposición de los pods que lo soliciten.

(Opcional) Prueba del rendimiento de EFA

Es recomendable que pruebe la configuración de EFA. Puede usar las pruebas de NCCL en el repositorio `aws-samples/awsome-distributed-training` de GitHub. Las pruebas de NCCL evalúan el rendimiento de la red mediante Nvidia Collective Communication Library. Los siguientes pasos envían las pruebas de NCCL en Amazon EKS.

1. Implementar el operador MPI de Kubeflow:

   Para las pruebas NCCL puede aplicar el operador MPI de Kubeflow. El operador Message Passing Interface (MPI, interfaz de paso de mensajes) facilita la ejecución de la formación distribuida de estilo AllReduce en Kubernetes. Para obtener más información, consulte Operador de MPI en GitHub.
2. Ejecutar la prueba de rendimiento NCCL de varios nodos para verificar GPUDirectRDMA/EFA:

   Para verificar el rendimiento de NCCL con GPUDirectRDMA sobre EFA, ejecute la prueba de rendimiento de NCCL estándar. Para obtener más información, consulte el repositorio oficial Pruebas de NCCL en GitHub.

   Complete los pasos que se indican a continuación para ejecutar una prueba de rendimiento de NCCL de dos nodos. En el trabajo de prueba de NCCL de ejemplo, cada trabajo solicita ocho GPU, 5210 Mi de `hugepages-2Mi`, cuatro EFA y 8000 Mi de memoria, lo que significa que cada trabajo consume todos los recursos de una instancia `p5.48xlarge`.

   a. Cree el manifiesto MPIJob:

Copie lo siguiente en un archivo denominado `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. Aplique las pruebas de NCCCL MPIJob:

   Aplique el manifiesto para enviar `MPIJob`. Esto creará dos instancias `p5.48xlarge` de Amazon EC2.

```
kubectl apply -f nccl-tests.yaml
```

   Un ejemplo de salida sería el siguiente.

```
mpijob.kubeflow.org/nccl-tests created
```

c. Comprobación de que el trabajo haya iniciado los pods:

   Consulte sus pods en ejecución.

```
kubectl get pods
```

   Un ejemplo de salida sería el siguiente.

```
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
```

El operador MPI crea un pod de lanzamiento y dos pods de trabajo (uno en cada nodo).

d. Comprobación de que el trabajo se ejecute correctamente con los registros:

Consulte el registro del pod `nccl-tests-launcher`. Sustituya *nbql9* con el valor de su salida.

```
kubectl logs -f nccl-tests-launcher-nbql9
```

Si la prueba se completó correctamente, puede implementar las aplicaciones que utilizan Nvidia Collective Communication Library.

Uso de instancias de AWS Inferentia con Amazon EKS para machine learning

En este tema se describe cómo crear un clúster de Amazon EKS con nodos que ejecutan instancias Inf1 de Amazon EC2 e implementar (opcionalmente) una aplicación de ejemplo. Las instancias Inf1 de Amazon EC2 están equipadas con chips de AWS Inferentia creados a medida por AWS para proporcionar un alto rendimiento y una inferencia de menor costo en la nube. Los modelos de machine learning se implementan en contenedores utilizando AWS Neuron, un kit de desarrollo de software (SDK) especializado que consta de un compilador, tiempo de ejecución y herramientas de perfilado que optimizan el rendimiento de inferencia de machine learning de los chips Inferentia. AWS Neuron admite marcos de machine learning populares como TensorFlow, PyTorch y MXNet.

> **Note**
>
> Los identificadores lógicos de los dispositivos Neuron deben ser contiguos. Si un pod que solicita varios dispositivos Neuron está programado en un tipo de instancia `inf1.6xlarge` o `inf1.24xlarge` (que tiene más de un dispositivo Neuron), ese pod no se iniciará si el programador de Kubernetes selecciona identificadores de dispositivo no contiguos. Para obtener más información, consulte Device logical IDs must be contiguous en GitHub.

Requisitos previos

- Debe tener `eksctl` instalado en el equipo. Si no lo tiene instalado, consulte la sección de Instalación en la documentación de `eksctl`.
- Debe tener `kubectl` instalado en el equipo. Para obtener más información, consulte the section called “Configure kubectl y eksctl”.
- (Opcional) Debe tener instalado `python3` en su equipo. Si no lo tiene instalado, consulte Descargas de Python para obtener instrucciones de instalación.

Creación de un clúster

1. Cree un clúster con nodos Inf1 de instancias de Amazon EC2. Puede reemplazar *`inf1.2xlarge`* con cualquier tipo de instancia Inf1. La utilidad `eksctl` detecta que va a lanzar un grupo de nodos con un tipo de instancia `Inf1` e iniciará sus nodos utilizando uno de las AMI optimizadas de Amazon Linux acelerado de Amazon EKS.

   > **Note**
   >
   > No puede usar roles de IAM para cuentas de servicio con TensorFlow Serving.

   ```
   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**
   >
   > Tenga en cuenta el valor de la siguiente línea de la salida. Se usa en un paso posterior (opcional).

```
[9]  adding identity "arn:aws:iam::111122223333:role/eksctl-inferentia-nodegroup-ng-
in-NodeInstanceRole-FI7HIYS3BS09" to auth ConfigMap
```

Al lanzar un grupo de nodos con instancias `Inf1`, `eksctl` instala automáticamente el complemento de dispositivo de Kubernetes de AWS Neuron. Este complemento anuncia dispositivos Neuron como un recurso del sistema al programador de Kubernetes, que puede solicitar un contenedor. Además de las políticas de IAM de nodos de Amazon EKS predeterminadas, se agrega la política de acceso de solo lectura de Amazon S3 para que la aplicación de ejemplo, que se trató en un paso posterior, pueda cargar un modelo entrenado desde Amazon S3.

2. Asegúrese de que todos los pods se hayan iniciado correctamente.

```
kubectl get pods -n kube-system
```

Salida abreviada:

```
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
```

(Opcional) Implementar una imagen de aplicación TensorFlow Serving

Un modelo entrenado debe compilarse en un destino de Inferentia antes de poder implementarlo en instancias Inferentia. Para continuar, necesitará un modelo TensorFlow optimizado para Neuron guardado en Amazon S3. Si aún no tiene un SavedModel, siga el tutorial para crear un modelo ResNet50 compatible con Neuron y cargue el SavedModel resultante a S3. ResNet-50 es un modelo de machine learning popular utilizado para tareas de reconocimiento de imágenes. Para obtener más información sobre cómo compilar modelos de Neuron, consulte El chip AWS Inferentia con DLAMI en la Guía para desarrolladores de aprendizaje profundo de AWS.

El manifiesto de implementación de ejemplo administra un contenedor de servicio de inferencia preconstruido para TensorFlow proporcionado por AWS Deep Learning Containers. Dentro del contenedor está el Runtime (Tiempo de ejecución) de AWS Neuron y la aplicación TensorFlow Serving. Una lista completa de Deep Learning Containers preconstruidos optimizados para Neuron

se mantiene en GitHub en Available Images (Imágenes disponibles). Al iniciarse, el DLC obtendrá su modelo de Amazon S3, lanzará Neuron TensorFlow Serving con el modelo guardado y esperará las solicitudes de predicción.

El número de dispositivos de Neuron asignados a su aplicación de servicio se puede ajustar cambiando el recurso `aws.amazon.com/neuron` en el yaml de implementación. Tenga en cuenta que la comunicación entre TensorFlow Serving y el tiempo de ejecución de Neuron ocurre a través de GRPC, lo que requiere pasar la capacidad de `IPC_LOCK` al contenedor.

1. Agregue la política de IAM de `AmazonS3ReadOnlyAccess` al rol de instancia de nodo que se creó en el paso 1 de Crear un clúster. Esto es necesario para que la aplicación de muestra pueda cargar un modelo entrenado desde Amazon S3.

```
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess \
    --role-name eksctl-inferentia-nodegroup-ng-in-NodeInstanceRole-FI7HIYS3BS09
```

2. Cree un archivo denominado `rn50_deployment.yaml` con el siguiente contenido. Actualice el código de región y la ruta del modelo para que coincida con la configuración deseada. El nombre del modelo es para fines de identificación cuando un cliente realiza una solicitud al servidor TensorFlow. En este ejemplo se utiliza un nombre de modelo para que coincida con un script de cliente ResNet50 de ejemplo que se utilizará en un paso posterior para enviar solicitudes de predicción.

```
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. Implemente el modelo.

```
kubectl apply -f rn50_deployment.yaml
```

4. Cree un archivo denominado `rn50_service.yaml` con el siguiente contenido. Los puertos HTTP y gRPC están abiertos para aceptar solicitudes de predicción.

```
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. Cree un servicio de Kubernetes para su aplicación de distribución de modelos de TensorFlow.

```
kubectl apply -f rn50_service.yaml
```

(Opcional) Haga predicciones contra su servicio de distribución de TensorFlow

1. Para realizar pruebas localmente, reenvíe el puerto gRPC al servicio `eks-neuron-test`.

```
kubectl port-forward service/eks-neuron-test 8500:8500 &
```

2. Cree un script de Python denominado `tensorflow-model-server-infer.py` con el siguiente contenido. Este script ejecuta la inferencia a través de gRPC, que es el marco de trabajo de servicio.

```
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. Ejecute el script para enviar predicciones a su servicio.

```
python3 tensorflow-model-server-infer.py
```

Un ejemplo de salida sería el siguiente.

```
[[(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)]]
```

Administración de los recursos computacionales para las cargas de trabajo de IA/ML en Amazon EKS

Esta sección está diseñada para ayudarlo a administrar los recursos computacionales para las cargas de trabajo de machine learning en Amazon Elastic Kubernetes Service (EKS). Encontrará información detallada sobre la reserva de GPU mediante bloques de capacidad para grupos de nodos administrados y nodos autoadministrados, incluidos los requisitos previos, la configuración de la plantilla de lanzamiento, las configuraciones de escalado, la preparación de la carga de trabajo y las consideraciones clave para gestionar los ciclos de vida de las reservas y la finalización correcta de los nodos.

Temas

- Creación de un grupo de nodos administrados con bloques de capacidad para ML
- Creación de nodos autoadministrados con bloques de capacidad para ML
- Uso de UltraServers P6e-GB200 con Amazon EKS

Creación de un grupo de nodos administrados con bloques de capacidad para ML

Los bloques de capacidad para machine learning (ML) permiten reservar instancias de GPU en una fecha futura para respaldar cargas de trabajo de ML de corta duración. Para obtener más información, consulte Bloques de capacidad para ML en la Guía del usuario de Amazon EC2 para instancias de Linux.

Consideraciones

Important

- Los bloques de capacidad solo están disponibles para determinados tipos de instancias de Amazon EC2 y regiones de AWS. Para obtener información sobre compatibilidad, consulte los requisitos previos para trabajar con bloques de capacidad en la Guía del usuario de Amazon EC2 para instancias de Linux.
- Para obtener más información, consulte Use Capacity Blocks for machine learning workloads en la Guía del usuario de Amazon EC2 Auto Scaling.

- Los grupos de nodos administrados con bloques de capacidad solo se pueden crear con plantillas de lanzamiento personalizadas.
- Al actualizar grupos de nodos administrados con bloques de capacidad, asegúrese de que el tamaño deseado del grupo de nodos esté establecido en `0`.

Creación de un grupo de nodos administrados con bloques de capacidad de Amazon EC2

Puede utilizar bloques de capacidad con grupos de nodos administrados de Amazon EKS para aprovisionar y escalar nodos de trabajo acelerados por GPU. Los ejemplos de plantillas de AWS CloudFormation que aparecen a continuación no cubren todos los aspectos necesarios en un clúster de producción. Por lo general, también querrá un script de arranque para unir el nodo al clúster y especificar la AMI acelerada de Amazon EKS. Para obtener más información, consulte the section called “Creación”.

1. Cree una plantilla de lanzamiento adecuada para sus cargas de trabajo y que funcione con los grupos de nodos administrados por Amazon EKS. Para obtener más información, consulte the section called “Plantillas de inicialización”.

   Además de los requisitos de los procedimientos anteriores, asegúrese de que `LaunchTemplateData` incluya lo siguiente:

   - `InstanceMarketOptions` con `"capacity-block"` establecido en `MarketType`
   - `CapacityReservationSpecification: CapacityReservationTarget` con `CapacityReservationId` establecido en el bloque de capacidad (por ejemplo: `cr-02168da1478b509e0`)
   - `InstanceType` establecido en un tipo de instancia que admita bloques de capacidad (por ejemplo: *`p5.48xlarge`*)

   A continuación se muestra un extracto de una plantilla de CloudFormation que crea una plantilla de lanzamiento dirigida a un bloque de capacidad. Para crear un grupo de nodos administrado por AMI personalizado, también puede agregar los parámetros `ImageId` y `UserData`.

```
NodeLaunchTemplate:
  Type: "AWS::EC2::LaunchTemplate"
  Properties:
    LaunchTemplateData:
```

```
      InstanceMarketOptions:
        MarketType: "capacity-block"
      CapacityReservationSpecification:
        CapacityReservationTarget:
          CapacityReservationId: "cr-02168da1478b509e0"
      InstanceType: p5.48xlarge
```

2. Utilice la plantilla de lanzamiento para crear un grupo de nodos administrados.

   A continuación, se muestra un ejemplo de comando de creación de grupos de nodos para bloques de capacidad. Reemplace *example-values* por los valores que se apliquen a su clúster.

   Al crear el grupo de nodos administrados por bloques de capacidad, haga lo siguiente:

   - Configure el `capacity-type` en `"CAPACITY_BLOCK"`. Si el tipo de capacidad no está configurado en `"CAPACITY_BLOCK"` o falta alguno de los valores de plantilla de lanzamiento requeridos anteriormente, se rechazará la solicitud de creación.
   - Al especificar `subnets` en la solicitud de creación, asegúrese de especificar solo la subred de la misma zona de disponibilidad que la reserva de capacidad.
   - Si especifica un valor distinto de cero de `desiredSize` en la solicitud de creación, Amazon EKS lo respetará al crear el grupo de escalado automático (ASG). Sin embargo, si la solicitud de creación se efectúa antes de que la reserva de capacidad esté activa, el ASG no podrá lanzar instancias de Amazon EC2 hasta que se active. Como resultado, las actividades de escalado de ASG tendrán errores de inicio. Siempre que la reserva se active, las instancias se lanzarán correctamente y el ASG se ampliará hasta alcanzar el `desiredSize` mencionado en el momento de la creación.

   ```
   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. Asegúrese de que los nodos se unan después del escalado vertical. Los clústeres de Amazon EKS que utilizan grupos de nodos administrados con bloques de capacidad no llevan a cabo

ninguna validación de que las instancias lanzadas realmente se unan al clúster y se registren en él.

4. Si ha configurado `desiredSize` en `0` en el momento de la creación, tiene diferentes opciones para escalar verticalmente el grupo de nodos cuando se active la reserva de capacidad:
   - Cree una política de escalado programado para el ASG que se ajuste a los tiempos de reserva del bloque de capacidad. Para obtener más información, consulte Scheduled scaling for Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.
   - Utilice la consola de Amazon EKS o `eks update-nodegroup-config` para actualizar la configuración de escalado y establezca el tamaño deseado del grupo de nodos.
   - Use el Escalador automático de clústeres de Kubernetes. Para obtener más información, consulte Escalador automático de clústeres en AWS.
5. El grupo de nodos ya está listo para la programación de cargas de trabajo y pods.
6. Para drenar los pods sin problemas antes de que finalice la reserva, Amazon EKS utiliza una política de escalado programado para reducir verticalmente el tamaño del grupo de nodos a `0`. Este escalado programado se establecerá con el nombre `Amazon EKS Node Group Capacity Scaledown Before Reservation End`. Se recomienda no editar ni eliminar esta acción.

   Amazon EC2 comienza a cerrar las instancias 30 minutos antes de la hora de finalización de la reserva. Como resultado, Amazon EKS configurará una reducción vertical programada del grupo de nodos 40 minutos antes de que finalice la reserva para poder expulsar los pods de forma segura y sin problemas.

Creación de nodos autoadministrados con bloques de capacidad para ML

Los bloques de capacidad para machine learning (ML) permiten reservar instancias de GPU en una fecha futura para respaldar cargas de trabajo de ML de corta duración. Para obtener más información, consulte Bloques de capacidad para ML en la Guía del usuario de Amazon EC2 para instancias de Linux.

Consideraciones

Important

- Los bloques de capacidad solo están disponibles para determinados tipos de instancias de Amazon EC2 y regiones de AWS. Para obtener información sobre compatibilidad, consulte

los requisitos previos para trabajar con bloques de capacidad en la Guía del usuario de Amazon EC2 para instancias de Linux.

- Si crea el grupo de nodos autoadministrados antes de que se active la reserva de capacidad, defina la capacidad deseada como `0`.
- Para que los nodos dispongan de tiempo suficiente para vaciarse correctamente, le sugerimos que programe el escalado para que llegue a cero más de 30 minutos antes de la hora de finalización de la reserva del bloque de capacidad.
- Para drenar los pods correctamente, se recomienda configurar AWS Node Termination Handler, tal como se explica en los pasos de ejemplo.

Uso de bloques de capacidad con nodos autoadministrados

Puede utilizar bloques de capacidad con Amazon EKS para aprovisionar y escalar nodos autoadministrados. En los siguientes pasos se ofrece un ejemplo general. Los ejemplos de plantillas de AWS CloudFormation no cubren todos los aspectos necesarios en una carga de trabajo de producción. Por lo general, también es necesario un script de arranque para unir el nodo al clúster, especificar una AMI acelerada de Amazon EKS y un perfil de instancia adecuado para unirse al clúster. Para obtener más información, consulte the section called “Amazon Linux”.

1. Cree una plantilla de lanzamiento que se adapte a su carga de trabajo. Para obtener más información, consulte Use Capacity Blocks for machine learning workloads en la Guía del usuario de Amazon EC2 Auto Scaling.

   Asegúrese de que `LaunchTemplateData` incluya lo siguiente:

   - `InstanceMarketOptions` con `"capacity-block"` establecido en `MarketType`
   - `CapacityReservationSpecification: CapacityReservationTarget` con `CapacityReservationId` establecido en el bloque de capacidad (por ejemplo: `cr-`*`02168da1478b509e0`*)
   - `IamInstanceProfile` con `Arn` establecido en el *`iam-instance-profile-arn`* aplicable
   - `ImageId` establecido en el *`image-id`* aplicable
   - `InstanceType` establecido en un tipo de instancia que admita bloques de capacidad (por ejemplo: *`p5.48xlarge`*)
   - `SecurityGroupIds` establecido en los ID aplicables (por ejemplo:*`sg-05b1d815d1EXAMPLE`*)
   - `UserData` establecido en los *`user-data`* aplicables a su grupo de nodos autoadministrado

A continuación se muestra un extracto de una plantilla de CloudFormation que crea una plantilla de lanzamiento dirigida a un bloque de capacidad.

```
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
```

Debe pasar la subred de la zona de disponibilidad en la que se realiza la reserva, ya que los bloques de capacidad son zonales.

2. Utilice la plantilla de lanzamiento para crear un grupo de nodos autoadministrado. Si lo hace antes de que se active la reserva de capacidad, ajuste la capacidad deseada a `0`. Al crear el grupo de nodos, asegúrese de especificar únicamente la subred correspondiente a la zona de disponibilidad en la que se va a reservar la capacidad.

   A continuación se muestra un ejemplo de plantilla de CloudFormation a la que puede hacer referencia al crear una que se aplique a su carga de trabajo. En este ejemplo se obtienen `LaunchTemplateId` y `Version` del recurso `AWS::Amazon EC2::LaunchTemplate` que se muestra en el paso anterior. También se obtienen los valores de `DesiredCapacity`, `MaxSize`, `MinSize` y `VPCZoneIdentifier` que se declaran en otras partes de la misma plantilla.

```
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. Una vez que el grupo de nodos se ha creado correctamente, asegúrese de registrar el valor de `NodeInstanceRole` correspondiente al grupo de nodos que se ha creado. Esto es necesario para asegurarse de que, al escalar el grupo de nodos, los nuevos nodos se unan al clúster y Kubernetes pueda reconocerlos. Para obtener más información, consulte las instrucciones de Consola de administración de AWS en Creación de nodos autoadministrados de Amazon Linux.
4. Se recomienda crear una política de escalado programado para el grupo de escalado automático que se ajuste a los tiempos de reserva del bloque de capacidad. Para obtener más información, consulte Scheduled scaling for Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

   Puede utilizar todas las instancias que haya reservado hasta 30 minutos antes de la hora de finalización del bloque de capacidad. Las instancias que aún estén en ejecución en ese momento comenzarán a finalizar. Para que los nodos dispongan de tiempo suficiente para vaciarse correctamente, le sugerimos que programe el escalado para que llegue a cero más de 30 minutos antes de la hora de finalización de la reserva del bloque de capacidad.

   Si en lugar de esto desea escalar verticalmente cada vez que el estado de la reserva de capacidad sea `Active`, debe actualizar la capacidad deseada para el grupo de escalado automático a la hora de inicio de la reserva del bloque de capacidad. En ese caso, también tendría que reducir verticalmente y manualmente más de 30 minutos antes de la hora de finalización de la reserva del bloque de capacidad.
5. El grupo de nodos ya está listo para la programación de cargas de trabajo y pods.
6. Para drenar los pods correctamente, se recomienda configurar AWS Node Termination Handler. Este controlador podrá observar eventos del ciclo de vida de “reducción horizontal de ASG” de Amazon EC2 Auto Scaling mediante EventBridge y permitir que el plano de control de Kubernetes tome las medidas necesarias antes de que la instancia deje de estar disponible. De lo contrario,

los objetos de Kubernetes y pods quedarán bloqueados en estado pendiente. Para obtener más información, consulte AWS Node Termination Handler en GitHub.

Si no configura Node Termination Handler, le recomendamos que comience a agotar los pods manualmente antes del plazo de 30 minutos, con el fin de que tengan tiempo suficiente para drenarse correctamente.

Uso de UltraServers P6e-GB200 con Amazon EKS

En este tema, se describe cómo configurar y usar Amazon EKS con UltraServers P6e-GB200. El tipo de instancia `p6e-gb200.36xlarge` con 4 GPU Blackwell de NVIDIA solo está disponible como UltraServers P6e-GB200. Existen dos tipos de UltraServers P6e-GB200. El UltraServer `u-p6e-gb200x36` tiene 9 instancias `p6e-gb200.36xlarge`, mientras que el UltraServer `u-p6e-gb200x72` tiene 18 instancias `p6e-gb200.36xlarge`.

Para obtener más información, consulte la página web de UltraServers P6e-GB200 de Amazon EC2.

Consideraciones

- Amazon EKS es compatible con UltraServers P6e-GB200 en las versión 1.33 y posteriores de Kubernetes. El lanzamiento de esta versión de Kubernetes admite la asignación dinámica de recursos (DRA), activada de forma predeterminada en EKS y en las AMI aceleradas optimizadas para EKS de AL2023. La DRA es un requisito para utilizar UltraServers P6e-GB200 con EKS. La DRA no se admite en el modo automático de EKS o Karpenter, y se recomienda utilizar grupos de nodos autoadministrados de EKS o grupos de nodos administrados de EKS al utilizar UltraServers P6e-GB200 con EKS.
- Los UltraServers P6e-GB200 están disponibles a través de los Bloques de capacidad de EC2 para ML Consulte the section called “Administración de la capacidad” para obtener información sobre cómo lanzar nodos de EKS con Bloques de capacidad.
- Al utilizar grupos de nodos administrados de EKS con Bloques de capacidad, debe usar plantillas de lanzamiento personalizadas. Al actualizar grupos de nodos administrados de EKS con UltraServers P6e-GB200, debe establecer el tamaño deseado del grupo de nodos en `0` antes de la actualización.
- Se recomienda utilizar la variante ARM de NVIDIA de AL2023 de las AMI aceleradas optimizadas para EKS. Esta AMI incluye la configuración y los componentes del nodo necesarios para funcionar con UltraServers P6e-GB200. Si decide crear su propia AMI, es responsable de instalar

y validar la compatibilidad del software del nodo y del sistema, incluidos los controladores. Para obtener más información, consulte the section called “Uso de AMI de GPU de Linux de EKS”.

- Se recomienda utilizar una versión `v20251103` o posterior de la AMI optimizada para EKS, que incluya la versión 580 del controlador NVIDIA. Esta versión del controlador NVIDIA permite que la Administración de memoria coherente basada en controladores (CDMM) aborde una posible sobrecarga de memoria. Cuando la CDMM está activada, no se admiten las siguientes capacidades: GPU multiinstancia (MIG) y vGPU de NVIDIA. Para obtener más información sobre la CDMM, consulte NVIDIA Coherent Driver-based Memory Management (CDMM).
- Cuando utilice el operador de GPU de NVIDIA con la AMI de NVIDIA de AL2023 optimizada para EKS, debe desactivar la instalación del controlador y el kit de herramientas por parte del operador, ya que se incluyen en la AMI. Las AMI de NVIDIA de AL2023 optimizadas para EKS no incluyen el complemento para dispositivos de Kubernetes de NVIDIA ni el controlador NVIDIA de DRA, que deben instalarse por separado.
- Cada instancia `p6e-gb200.36xlarge` se puede configurar con hasta 17 tarjetas de red y puede usar EFA para la comunicación entre UltraServers. El tráfico de la red de carga de trabajo puede atravesar UltraServers, pero, para obtener el máximo rendimiento, se recomienda programar las cargas de trabajo en el mismo UltraServer y utilizar IMEX para la comunicación de la GPU dentro del UltraServer. Para obtener más información, consulte Configuración de EFA para instancias P6e-GB200.
- Cada instancia `p6e-gb200.36xlarge` cuenta con 3 almacenes de instancias de 7,5 TB cada uno. De forma predeterminada, la AMI optimizada para EKS no formatea ni monta los almacenes de instancias. El almacenamiento efímero del nodo se puede compartir entre los pods que solicitan almacenamiento efímero y las imágenes de contenedores que se descargan en el nodo. Si utiliza la AMI optimizada para EKS de AL2023, se puede configurar como parte del arranque de nodos en los datos del usuario mediante la configuración de la política de almacenamiento local de instancias en NodeConfig como RAID0. Si se establece en RAID0, se transfieren los almacenes de instancias y se configura el tiempo de ejecución del contenedor y el kubelet para utilizar este almacenamiento efímero.

Componentes

Se recomiendan los siguientes componentes para ejecutar cargas de trabajo en EKS con los UltraServers P6e-GB200. Si lo desea, puede utilizar el operador de GPU de NVIDIA para instalar los componentes del nodo de NVIDIA. Cuando utilice el operador de GPU de NVIDIA con la AMI de NVIDIA de AL2023 optimizada para EKS, debe desactivar la instalación del controlador y el kit de herramientas por parte del operador, ya que se incluyen en la AMI.

Pila	Componente
AMI acelerada optimizada para EKS	Kernel 6.12
	Controlador de GPU de NVIDIA
	Controlador de modo de usuario CUDA de NVIDIA
	Kit de herramientas de contenedores de NVIDIA
	NVIDIA Fabric Manager
	Controlador de IMEX de NVIDIA
	Administrador de subredes NVLink de NVIDIA
	Controlador de EFA
Componentes que se ejecutan en el nodo	CNI de VPC
	Complemento para dispositivos de EFA
	Complemento para dispositivos de K8s de NVIDIA
	Controlador de DRA de NVIDIA
	Detección de características de nodos (NFD) de NVIDIA
	Detección de características de GPU (GFD) de NVIDIA

Los componentes de los nodos de la tabla anterior llevan a cabo las siguientes funciones:

- CNI de VPC: asigna las IP de VPC como la interfaz de red principal para los pods que se ejecutan en EKS.
- Complemento para dispositivos de EFA: asigna los dispositivos EFA como redes secundarias para los pods que se ejecutan en EKS. Es responsable del tráfico de red en los UltraServers P6e-GB200. En el caso de las cargas de trabajo de varios nodos, las de GPU a GPU dentro de un UltraServer pueden fluir a través de NVLink de varios nodos.

- Complemento para dispositivos de Kubernetes de NVIDIA: asigna las GPU como dispositivos para los pods que se ejecutan en EKS. Se recomienda utilizar el complemento para dispositivos de Kubernetes de NVIDIA hasta que la funcionalidad de asignación de GPU del controlador de DRA de NVIDIA deje de ser experimental. Consulte las versiones de los controladores de DRA de NVIDIA para obtener información actualizada.
- Controlador de DRA de NVIDIA: activa los recursos personalizados de ComputeDomain que facilitan la creación de dominios de IMEX que siguen las cargas de trabajo que se ejecutan en los UltraServers P6e-GB200.
  - El recurso ComputeDomain describe un dominio de intercambio de memoria entre nodos (IMEX). Cuando se implementan en el clúster cargas de trabajo con un ResourceClaim para un ComputeDomain, el controlador de DRA de NVIDIA crea automáticamente un DaemonSet de IMEX que se ejecuta en los nodos coincidentes y establece los canales de IMEX entre los nodos antes de que se inicie la carga de trabajo. Para obtener más información sobre IMEX, consulte la descripción general de IMEX de NVIDIA para sistemas NVLink de varios nodos.
  - El controlador de DRA de NVIDIA utiliza una etiqueta de ID de grupo (`nvidia.com/gpu.clique`) aplicada por la GFD de NVIDIA y que transmite los conocimientos sobre la topología de la red y el dominio de NVLink.
  - Es una práctica recomendada crear un ComputeDomain por cada trabajo de carga de trabajo.
- Detección de características de nodos (NFD) de NVIDIA: dependencia necesaria para que la GFD aplique etiquetas a los nodos en función de los atributos de nodo detectados.
- Detección de características de GPU (GFD) de NVIDIA: aplica una etiqueta de topología estándar de NVIDIA llamada `nvidia.com/gpu.clique` a los nodos. Los nodos dentro de un mismo `nvidia.com/gpu.clique` tienen accesibilidad de NVLink de varios nodos, y puede utilizar las afinidades de los pods en la aplicación para programar los pods en el mismo dominio de NVLink.

Procedimiento

En la siguiente sección, se supone que cuenta con un clúster de EKS que ejecuta la versión 1.33 o posterior de Kubernetes con uno o más grupos de nodos con UltraServers P6e-GB200 que ejecutan la AMI acelerada optimizada para EKS ARM de NVIDIA de AL2023. Consulte los enlaces de the section called “Administración de la capacidad” para obtener los pasos con los requisitos previos para los grupos de nodos administrados y los nodos autoadministrados de EKS.

En el siguiente procedimiento, se utilizan los siguientes componentes.

Nombre	Versión	Descripción
Operador de GPU de NVIDIA	25.3.4+	Para la administración del ciclo de vida de los complementos necesarios, como el complemento para dispositivos de Kubernetes de NVIDIA y la NFD o GFD.
Controladores de DRA de NVIDIA	25.8.0+	Para la administración de dominios de IMEX y CRD de ComputeDomain.
Complemento para dispositivos de EFA	0.5.14+	Para la comunicación entre UltraServers.

Instalación del operador de GPU de NVIDIA

El operador de GPU de NVIDIA simplifica la administración de los componentes necesarios para usar las GPU en los clústeres de Kubernetes. Como el kit de herramientas de contenedores y el controlador de GPU de NVIDIA se instalan como parte de la AMI acelerada optimizada para EKS, deben configurarse como `false` en la configuración de valores de Helm.

1. Cree un archivo de valores de Helm denominado `gpu-operator-values.yaml` con la configuración siguiente.

```
devicePlugin:
  enabled: true
nfd:
  enabled: true
gfd:
  enabled: true
driver:
  enabled: false
toolkit:
  enabled: false
migManager:
  enabled: false
```

2. Instale el operador de GPU de NVIDIA para el clúster con el archivo `gpu-operator-values.yaml` creado en el paso anterior.

```
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
```

Instalación del controlador de DRA de NVIDIA

A partir de la versión `v25.3.4` del operador de GPU de NVIDIA, el controlador de DRA de NVIDIA debe instalarse por separado. Se recomienda seguir las notas de la versión del operador de GPU de NVIDIA, ya que esto podría cambiar en una versión futura.

1. Cree un archivo de valores de Helm denominado `dra-values.yaml` con la configuración siguiente. Tenga en cuenta `nodeAffinity` y `tolerations` que configuran el controlador de DRA para implementar solo en nodos con una GPU de NVIDIA.

```
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. Instale el controlador de DRA de NVIDIA para el clúster con el archivo `dra-values.yaml` creado en el paso anterior.

```
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. Tras la instalación, el controlador de DRA crea recursos `DeviceClass` que permiten a Kubernetes comprender y asignar los recursos `ComputeDomain`, lo que permite que las cargas de trabajo de GPU distribuidas en UltraServers P6e-GB200 administren el IMEX.

   Confirme que los recursos de DRA estén disponibles con los siguientes comandos.

```
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
```

Instalación del complemento para dispositivos de EFA

Para utilizar la comunicación de EFA entre UltraServers, debe instalar el complemento para dispositivos de Kubernetes para EFA. Las instancias P6e-GB200 se pueden configurar con hasta 17 tarjetas de red y el NCI principal (índice 0) debe ser del tipo `interface` y admitir un ancho de banda ENA de hasta 100 Gbps. Configure las interfaces de EFA y ENA según sus requisitos durante el aprovisionamiento de nodos. Consulte la documentación de AWS sobre la configuración de EFA para instancias P6e-GB200 para obtener más detalles sobre la configuración de EFA.

1. Cree un archivo de valores de Helm denominado `efa-values.yaml` con la configuración siguiente.

```
tolerations:
  - key: nvidia.com/gpu
    operator: Exists
    effect: NoSchedule
```

2. Instale el operador de DRA de NVIDIA para el clúster con el archivo `dra-values.yaml` creado en el paso anterior.

```
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
```

   Por ejemplo, si configuró sus instancias con 1 interfaz de solo EFA en cada grupo del NCI, al describir un nodo, se espera que haya 4 dispositivos de EFA asignables por nodo.

```
kubectl describe node/<gb200-node-name>
```

```
Capacity:
  ...
  vpc.amazonaws.com/efa:  4
Allocatable:
  ...
  vpc.amazonaws.com/efa:  4
```

Validación del IMEX mediante NVLink de varios nodos

Para una prueba de NCCL de NVLink de varios nodos y otras microrreferencias, consulte el repositorio de GitHub awesome-distributed-training. En los siguientes pasos se muestra cómo ejecutar una prueba de NVLink de varios nodos con nvbandwidth.

1. Para ejecutar una prueba de ancho de banda de varios nodos en dos nodos del dominio NVL72, instale antes el operador de MPI:

```
kubectl create -f https://github.com/kubeflow/mpi-operator/releases/download/v0.7.0/
mpi-operator.yaml
```

2. Cree un archivo de valores de Helm denominado `nvbandwidth-test-job.yaml` que defina el manifiesto de la prueba. Tenga en cuenta la afinidad del pod `nvidia.com/gpu.clique` para programar los trabajadores en el mismo dominio de NVLink que tiene accesibilidad de NVLink de varios nodos.

   A partir de la versión `v25.8.0` del controlador de DRA de NVIDIA, los ComputeDomains son elásticos y se puede establecer `.spec.numNodes 0` en la definición de ComputeDomain. Consulte las notas de la versión más reciente del controlador de DRA de NVIDIA para ver las actualizaciones.

```
---
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. Cree el ComputeDomain e inicie el trabajo con el siguiente comando.

```
kubectl apply -f nvbandwidth-test-job.yaml
```

4. Al crear el ComputeDomain, puede ver que el ComputeDomain de la carga de trabajo tiene dos nodos:

```
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. Revise los resultados del trabajo con el siguiente comando.

```
kubectl logs --tail=-1 -l job-name=nvbandwidth-test-launcher
```

6. Cuando se haya completado la prueba, elimínela con el siguiente comando.

```
kubectl delete -f nvbandwidth-test-job.yaml
```

Recetas para optimizar su clúster de Amazon EKS para cargas de trabajo de IA/ML

Esta sección está diseñada para proporcionar recetas breves para optimizar su clúster de Amazon EKS, especialmente para las cargas de trabajo de IA/ML que implican hardware especializado. Encontrará instrucciones sobre cómo evitar que los pods se programen en nodos específicos añadiendo taints a los grupos de nodos administrados, incluidos los requisitos previos, los procedimientos paso a paso y las consideraciones de implementación.

Temas

- Receta: Limitación para que los pods no se programen en nodos específicos

Receta: Limitación para que los pods no se programen en nodos específicos

Descripción general

Los nodos que cuentan con procesadores especializados, como las GPU, pueden ser más costosos de ejecutar que aquellos de máquinas estándar. Para proteger estos nodos de las cargas de trabajo que no requieren un hardware especial, se pueden usar las taints de Kubernetes. Las taints marcan los nodos para repeler los pods que no tienen tolerancias coincidentes, lo que garantiza que solo se programen las cargas de trabajo compatibles. Para obtener más información, consulte Taints y toleraciones en la documentación de Kubernetes.

Las taints de nodos de Kubernetes se pueden aplicar a grupos de nodos administrados nuevos y existentes mediante la Consola de administración de AWS o a través de la API de Amazon EKS. Esta receta muestra cómo aplicar taints a los grupos de nodos administrados de Amazon EKS

mediante la CLI de AWS. Para obtener información sobre la creación de un grupo de nodos con una taint mediante la Consola de administración de AWS, consulte the section called “Creación”.

Requisitos previos

- Un clúster existente de Amazon EKS.
- La CLI de AWS instalada y configurada con los permisos adecuados.

Pasos

Paso 1: Creación de un grupo de nodos con taints

Use el comando `aws eks create-nodegroup` para crear un nuevo grupo de nodos administrados con taints. En este ejemplo, se aplica una taint con la clave `dedicated`, el valor `gpuGroup` y el efecto `NO_SCHEDULE`.

```
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"
     }
   ]
}'
```

Para obtener más información y ejemplos, consulte taint en la documentación de referencia de Kubernetes.

Paso 2: Actualización de las taints en un grupo de nodos existente

Utilice el comando aws eks update-nodegroup-config de la CLI de AWS para añadir, eliminar o reemplazar taints en los grupos de nodos administrados.

```
aws eks update-nodegroup-config
  --cluster-name my-cluster
  --nodegroup-name node-taints-example
  --taints 'removeTaints=[{key=dedicated,value=gpuGroup,effect=NO_SCHEDULE}]'
```

Notas

- Las taints se pueden actualizar después de crear el grupo de nodos mediante la API `UpdateNodegroupConfig`.
- La clave de la taint debe comenzar con una letra o un número. Puede contener letras, números, guiones (`-`), puntos (`.`) y guiones bajos (`_`). Puede tener hasta 63 caracteres.
- De manera opcional, la clave de la taint puede comenzar con un prefijo de subdominio DNS y una única `/`. Si comienza con un prefijo de subdominio DNS, puede tener 253 caracteres de longitud.
- El valor es opcional y debe comenzar por una letra o un número. Puede contener letras, números, guiones (`-`), puntos (`.`) y guiones bajos (`_`). Puede tener hasta 63 caracteres.
- Cuando se usa Kubernetes directamente o la Consola de administración de AWS, el efecto de la taint debe ser `NoSchedule`, `PreferNoSchedule` o `NoExecute`. Sin embargo, cuando se usa la AWS CLI o la API, el efecto de la taint debe ser `NO_SCHEDULE`, `PREFER_NO_SCHEDULE` o `NO_EXECUTE`.
- Se permite un máximo de 50 taints por grupo de nodos.
- Si las taint que se crearon mediante un grupo de nodos administrado se eliminan manualmente de un nodo, Amazon EKS no volverá a añadir las taint al nodo. Esto es cierto incluso si las taint se especifican en la configuración del grupo de nodos administrado.

Recursos para empezar a utilizar la IA/ML en Amazon EKS

Para comenzar a utilizar el machine learning en EKS, elija uno de estos patrones prescriptivos para preparar rápidamente un clúster de EKS y el software y hardware de ML para comenzar a ejecutar cargas de trabajo de ML.

Talleres

Taller sobre IA generativa en Amazon EKS

Descubra cómo empezar a utilizar las aplicaciones de modelos de lenguaje de gran tamaño (LLM) y la inferencia en Amazon EKS. Descubra cómo implementar y administrar cargas de trabajo de LLM aptas para producción. Mediante talleres prácticos, explorará cómo aprovechar Amazon EKS junto con los servicios de AWS y las herramientas de código abierto para crear soluciones de LLM sólidas. El entorno del taller proporciona toda la infraestructura y las herramientas necesarias, lo que le permite centrarse en el aprendizaje y la implementación.

IA generativa en Amazon EKS con Neuron

Descubra cómo empezar a utilizar las aplicaciones de modelos de lenguaje de gran tamaño (LLM) y la inferencia en Amazon EKS. Descubra cómo implementar y administrar cargas de trabajo de LLM aptas para producción, cómo implementar patrones RAG avanzados con bases de datos vectoriales y cómo crear aplicaciones de LLM respaldadas por datos utilizando marcos de código abierto. Mediante talleres prácticos, explorará cómo aprovechar Amazon EKS junto con los servicios de AWS y las herramientas de código abierto para crear soluciones de LLM sólidas. El entorno del taller proporciona toda la infraestructura y las herramientas necesarias, lo que le permite centrarse en el aprendizaje y la implementación.

prácticas recomendadas

Los temas centrados en la IA/ML de la guía de prácticas recomendadas de Amazon EKS proporcionan recomendaciones detalladas en las siguientes áreas para optimizar las cargas de trabajo de IA/ML en Amazon EKS.

Cálculo y escalado automático de IA/ML

En esta sección, se describen las prácticas recomendadas para optimizar el cálculo y el escalado automático de IA/ML en Amazon EKS, centrándose en la administración de los recursos de la GPU, la resiliencia de los nodos y el escalado de las aplicaciones. Proporciona estrategias como la programación de las cargas de trabajo con etiquetas conocidas y la afinidad entre nodos, el uso de bloques de capacidad de ML o reservas de capacidad bajo demanda, y la implementación de comprobaciones de estado de los nodos con herramientas como EKS Node Monitoring Agent.

Redes de IA/ML

En esta sección, se describen las prácticas recomendadas para optimizar las redes de IA/ML en Amazon EKS a fin de mejorar el rendimiento y la escalabilidad, incluidas estrategias como seleccionar instancias con un mayor ancho de banda de la red o Elastic Fabric Adapter (EFA) para el entrenamiento distribuido, instalar herramientas como MPI y NCCL, y habilitar la delegación de prefijos para aumentar las direcciones IP y mejorar los tiempos de lanzamiento de los pods.

Seguridad de IA/ML

Esta sección se centra en proteger el almacenamiento de datos y garantizar la conformidad de las cargas de trabajo de IA/ML en Amazon EKS, incluidas prácticas como el uso de Amazon S3 con AWS Key Management Service (KMS) para el cifrado del servidor (SSE-KMS), la configuración de buckets con claves KMS regionales y claves de bucket de S3 para reducir los costos, la concesión de permisos de IAM para acciones de KMS (como el descifrado de pods de EKS) y la auditoría con registros de AWS CloudTrail.

Almacenamiento de IA/ML

En esta sección, se describen las prácticas recomendadas para optimizar el almacenamiento en las cargas de trabajo de IA/ML en Amazon EKS, incluidas prácticas como implementar modelos con controladores de CSI para montar servicios como S3, FSx for Lustre o EFS como volúmenes persistentes, seleccionar el almacenamiento en función de las necesidades de carga de trabajo (p. ej., FSx for Lustre para entrenamiento distribuido con opciones como Scratch-SSD o Persistent-SSD) y habilitar características como la compresión de datos y la división en bandas.

Observabilidad de IA/ML

Esta sección se centra en la supervisión y la optimización del uso de la GPU para las cargas de trabajo de IA/ML en Amazon EKS a fin de mejorar la eficiencia y reducir los costos, e incluye estrategias como centrarse en un uso elevado de la GPU con herramientas como CloudWatch Container Insights y el exportador de DCGM de NVIDIA integrado con Prometheus y Grafana, y métricas que le recomendamos que analice para sus cargas de trabajo de IA/ML.

Rendimiento de IA/ML

Esta sección se centra en mejorar el escalado y el rendimiento de las aplicaciones para las cargas de trabajo de IA/ML en Amazon EKS mediante la administración de imágenes de contenedores y la optimización del inicio, incluidas prácticas como el uso de imágenes base pequeñas y ligeras o AWS

Deep Learning Containers con compilaciones de varias etapas, la precarga de imágenes mediante instantáneas de EBS o la extracción previa a la memoria caché de tiempo de ejecución mediante DaemonSets o implementaciones.

Arquitecturas de referencia

Explore estos repositorios de GitHub para obtener arquitecturas de referencia, código de muestra y utilidades para implementar el entrenamiento distribuido y la inferencia para cargas de trabajo de IA/ML en Amazon EKS y otros servicios de AWS.

AWSome Distributed Training

Este repositorio ofrece una colección de prácticas recomendadas, arquitecturas de referencia, ejemplos de entrenamiento del modelo y utilidades para entrenar modelos de gran tamaño en AWS. Es compatible con el entrenamiento distribuido con Amazon EKS, que incluye plantillas de CloudFormation para clústeres de EKS, compilaciones personalizadas de AMI y contenedores, casos de prueba para marcos como PyTorch (DDP/FSDP, MegatronLM, NeMo) y JAX, y herramientas para la validación, la observabilidad y el monitoreo del rendimiento, como exportadores Prometheus de EFA y sistemas Nvidia Nsight.

AWSome Inference

Este repositorio proporciona arquitecturas de referencia y casos de prueba para optimizar las soluciones de inferencia en AWS, centrándose en Amazon EKS y en las instancias de EC2 aceleradas. Incluye configuraciones de infraestructura para clústeres de VPC y EKS, proyectos para marcos como NIM de NVIDIA, TensorRT-LLM, Triton Inference Server y RayService, con ejemplos de modelos como Llama3-8B y Llama 3.1 405B. Incluye implementaciones de varios nodos mediante el LeaderWorkerSet de K8s, el escalado automático de EKS, las GPU de instancia múltiple (MIG) y casos de uso reales, como un bot de audio para ASR, inferencia y TTS.

Tutoriales

Si le interesa configurar plataformas y marcos de machine learning en EKS, explore los tutoriales que se describen en esta sección. Estos tutoriales abarcan desde patrones para aprovechar al máximo los procesadores de la GPU hasta la elección de herramientas de modelado o la creación de marcos de trabajo para sectores especializados.

Cree plataformas de IA generativa en EKS

- Implemente modelos de IA generativa en Amazon EKS

- Creación de plataformas multiusuario de JupyterHub en Amazon EKS

Ejecute marcos especializados de IA generativa en EKS

- Acelere las cargas de trabajo de entrenamiento distribuido de IA generativa con el marco NVIDIA NeMo en Amazon EKS
- Ejecución de TorchServe en Amazon Elastic Kubernetes Service

Maximice el rendimiento de la GPU NVIDIA para ML en EKS

- Implemente el uso compartido de GPU para utilizar de forma eficiente las GPU NVIDIA en los clústeres de EKS:

  Uso compartido de GPU en Amazon EKS con segmentación de tiempo de NVIDIA e instancias de EC2 aceleradas
- Utilice GPU de múltiples instancias (MIG) y microservicios NIM para ejecutar más pods por GPU en los clústeres de EKS:

  Maximización de la utilización de la GPU con la GPU de múltiples instancias (MIG) de NVIDIA en Amazon EKS: cómo ejecutar más pods por GPU para mejorar el rendimiento
- Cree e implemente un sistema de machine learning escalable en Kubernetes con Kubeflow en AWS

Ejecución de cargas de trabajo de codificación de video en EKS

- Entrega de contenido en video con GPU fraccionadas en contenedores en Amazon EKS

Aceleración de la carga de imágenes para cargas de trabajo de inferencia

- Cómo H2O.ai optimizó y protegió su infraestructura de IA/ML con Karpenter y Bottlerocket

Supervisión de las cargas de trabajo de ML

- Supervisión de cargas de trabajo de GPU en Amazon EKS mediante servicios de código abierto administrados por AWS
- Habilite las métricas de GPU basadas en pods en Amazon CloudWatch

Herramientas de Amazon EKS

Amazon EKS proporciona un conjunto de herramientas de AWS que lo ayudan a administrar, solucionar problemas e interactuar con los clústeres de EKS de manera más eficiente. Estas herramientas aprovechan las capacidades basadas en IA y las interfaces estandarizadas para simplificar las operaciones de los clústeres, acelerar la resolución de problemas y permitir interacciones en lenguaje natural con recursos de Kubernetes. Entre las herramientas disponibles, se incluyen las siguientes:

- Servidor Protocolo de contexto para modelos (MCP) de Amazon EKS: un servicio completamente administrado que permite experiencias basadas en IA para la administración de clústeres de EKS. El servidor MCP de EKS se integra con asistentes de codificación de IA compatibles con el MCP para proporcionar un conocimiento contextual y en tiempo real de clústeres y recursos de Kubernetes. Úselo para administrar clústeres, implementar aplicaciones, solucionar problemas y consultar la documentación de EKS a través de interacciones en lenguaje natural.
- Amazon Q en la consola de Amazon EKS: una integración de resolución de problemas basada en IA disponible directamente en la Consola de administración de AWS. Amazon Q analiza automáticamente los problemas de los clústeres y proporciona los resultados de la investigación, el análisis de la causa raíz y sugerencias de pasos de mitigación cuando encuentra errores o advertencias en la consola de EKS.

Estas herramientas funcionan en conjunto para proporcionar un soporte integral durante todo el ciclo de vida del clúster de EKS, desde la configuración inicial hasta las operaciones continuas y la resolución de problemas. Ya sea que esté desarrollando aplicaciones, administrando la infraestructura o resolviendo problemas de producción, estas herramientas de AWS lo ayudan a trabajar de manera más eficiente con clústeres de EKS.

Temas

- Servidor Protocolo de contexto para modelos (MCP) de Amazon EKS.
- Uso de Amazon Q Developer en la consola de Amazon EKS

Servidor Protocolo de contexto para modelos (MCP) de Amazon EKS.

El servidor MCP de Amazon EKS es un servicio completamente administrado que permite experiencias que utilizan IA para el desarrollo y las operaciones. El protocolo de contexto para modelos (MCP) proporciona una interfaz estandarizada que enriquece a las aplicaciones y los agentes de IA con un conocimiento contextual y en tiempo real de los clústeres de EKS y los recursos de Kubernetes, lo que permite respuestas más precisas y sensibles al contexto y flujos de trabajo basados en IA durante todo el ciclo de vida de la aplicación, desde la configuración inicial hasta la optimización de la producción y la solución de problemas.

> **Note**
>
> El servidor MCP de Amazon EKS se encuentra en versión preliminar para Amazon EKS y está sujeto a cambios.

Descripción general

El servidor MCP de EKS se puede integrar fácilmente con cualquier asistente de codificación de IA compatible con MCP, como Kiro, la CLI de Amazon Q Developer o herramientas de terceros, como Cursor o Cline. Para comenzar, el servidor MCP de EKS lo guiará por la creación de clústeres, el aprovisionamiento automático de los requisitos previos y la aplicación de las prácticas recomendadas de AWS. Durante el desarrollo, simplifica las operaciones de EKS y Kubernetes al proporcionar flujos de trabajo de alto nivel para la implementación de aplicaciones y la administración de clústeres. Para la depuración y la solución de problemas, el servidor acelera la resolución de problemas mediante las ayudas integradas para la solución de problemas y el acceso a la base de conocimientos, disponibles en la consola de EKS y en sus asistentes de IA favoritos. Se puede acceder a estas funcionalidades mediante interacciones en lenguaje natural, lo que le permite llevar a cabo operaciones complejas de Kubernetes de manera más intuitiva y eficiente.

El servidor MCP de EKS completamente administrado está alojado en la nube de AWS, lo que elimina la necesidad de instalación y mantenimiento locales. Proporciona funcionalidades empresariales, como actualizaciones y parches automáticos, seguridad centralizada mediante la integración de AWS IAM, registro de auditoría integral a través de AWS CloudTrail y la escalabilidad, fiabilidad y soporte demostrados de AWS. El servidor MCP de EKS completamente administrado y alojado en la nube de AWS ofrece las siguientes ventajas clave:

- Eliminación de la instalación y el mantenimiento. Como el servidor MCP de EKS está alojado en la nube de AWS, ya no es necesario administrar las actualizaciones de las versiones ni solucionar problemas con los servidores locales. Solo tiene que configurar su asistente de IA para que se conecte al punto de conexión del servidor MCP de EKS alojado y lo tendrá todo listo para comenzar a trabajar con sus clústeres de EKS.
- Administración de acceso centralizado. El servidor MCP de EKS se integra con IAM, lo que proporciona una forma centralizada y segura de controlar el acceso al servidor. Todas las solicitudes se firman mediante AWS SigV4 a través de un proxy ligero, lo que permite una integración perfecta con sus políticas de IAM y credenciales de AWS existentes.
- Supervisión y visibilidad mejoradas. La integración con AWS CloudTrail captura las llamadas a las herramientas de inicialización y acceso completo hechas a través del servicio alojado, lo que permite llevar a cabo registros de auditoría detallados e informes de cumplimiento.
- Actualización constante. Reciba nuevas herramientas, características y correcciones de errores automáticamente sin necesidad de actualizar las instalaciones locales. El servicio alojado se mejora continuamente en función de sus comentarios y de las prácticas recomendadas de AWS.

Ejemplos de integraciones

El servidor MCP de EKS proporciona varias herramientas que puede utilizar para lo siguiente:

- Administración del clúster: cree, configure y administre los clústeres de EKS con las prácticas recomendadas automatizadas.
- Administración de los recursos de Kubernetes: implemente aplicaciones, administre recursos de Kubernetes e inspeccione el estado del clúster.
- Solución de problemas del clúster: diagnostique los problemas mediante las herramientas de solución de problemas integradas y la base de conocimientos de los manuales de procedimientos.
- Consulta de la documentación: busque y recupere la documentación pertinentes de EKS de forma contextual.

Exploración de los datos

```
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
```

Comprobación de los recursos de Kubernetes

```
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
```

Solución de problemas de

```
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
```

Creación de recursos (si el modo de “escritura” está activado)

```
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
```

Introducción

Para empezar, consulte the section called “Introducción”.

Introducción al servidor MCP de Amazon EKS

En esta guía, se explican los pasos necesarios para configurar y utilizar el servidor MCP de EKS con los asistentes de código de IA. Obtendrá información sobre cómo configurar el entorno, conectarse al servidor y comenzar a administrar los clústeres de EKS mediante interacciones en lenguaje natural.

> **Note**
>
> El servidor MCP de Amazon EKS se encuentra en versión preliminar para Amazon EKS y está sujeto a cambios.

Requisitos previos

Antes de comenzar, asegúrese de haber realizado las siguientes tareas:

- Creación de una cuenta de AWS con acceso a Amazon EKS
- Instalación y configuración de la AWS CLI con credenciales

- Instalación de Python 3.10+
- Instalar uv

Configuración

1. Verificación de los requisitos previos de

```
# Check that your Python version is 3.10 or higher
python3 --version

# Check uv installation
uv --version

# Verify CLI configuration
aws configure list
```

2. Configuración de permisos de IAM

Para conectarse al servidor MCP de EKS, el rol de IAM debe tener adjuntas las siguientes políticas: **`eks-mcp:InvokeMcp`** (permisos necesarios para la inicialización y recuperación de información sobre las herramientas disponibles), **`eks-mcp:CallReadOnlyTool`** (permisos necesarios para el uso de herramientas de solo lectura) y **`eks-mcp:CallPrivilegedTool`** (permisos necesarios para el uso de herramientas de acceso completo [escritura]). Estos permisos de `eks-mcp` se incluyen en las políticas administradas de AWS de solo lectura y acceso completo que se indican a continuación.

- Abra la consola de IAM.
- En el panel de navegación izquierdo, elija Usuarios, Grupos de usuarios o Roles en función de la identidad a la que desee adjuntar la política y, a continuación, del nombre del usuario, grupo o rol específico.
- Elija la pestaña Permisos.
- Elija Adjuntar políticas (o Agregar permisos si es la primera vez).
- En la lista de políticas, busque y seleccione la política administrada que desee adjuntar:
- Operaciones de solo lectura: AmazonEKSMCPReadOnlyAccess
- Elija Adjuntar políticas (o Siguiente, seguido de Agregar permisos para confirmar).

De este modo, se adjunta la política y los permisos entran en vigor inmediatamente. Puede adjuntar varias políticas a la misma identidad y cada política puede incluir varios permisos. Para obtener más

información sobre estas políticas, consulte Políticas administradas de AWS para Amazon Elastic Kubernetes Service.

3. Elección de un asistente de IA

Elija uno de los siguientes asistentes de IA compatibles con MCP o cualquier herramienta compatible con MCP:

- Instalación de la CLI de Amazon Q Developer
- Instalación de Kiro
- Instalación de Cursor
- Instalación de la extensión de Cline de VS Code

Paso 1: configuración del asistente de IA

Elija una de las siguientes opciones para configurar el asistente de código de IA. Al completar este paso, se configura el asistente de código de IA para que utilice el proxy MCP para AWS, que es necesario para acceder de forma segura y autenticada al servidor MCP de Amazon EKS. Esto implica agregar o editar el archivo de configuración del MCP (por ejemplo, `~/.aws/amazonq/mcp.json` para la CLI de Amazon Q Developer). El proxy actúa como un puente del cliente, gestiona la autenticación de AWS SigV4 con sus credenciales de AWS locales y permite la detección dinámica de herramientas para interactuar con los servidores MCP AWS de backend, como el servidor MCP de EKS. Para obtener más información, consulte el repositorio del proxy MCP para AWS.

Opción A: CLI de Amazon Q Developer

La CLI de Q Developer proporciona la experiencia más integrada con el servidor MCP de EKS.

1. Búsqueda del archivo de configuración de MCP

- macOS/Linux: `~/.aws/q/mcp.json`
- Windows: `%USERPROFILE%\.aws\q\mcp.json`

2. Adición de la configuración del servidor MCP

Cree el archivo de configuración si no existe. Asegúrese de sustituir el marcador de posición de región (`{region}`) por la región deseada.

Para 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}"
      ]
    }
  }
}
```

Para 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}"
      ]
    }
```

```
  }
}
```

Nota de seguridad: Se puede utilizar `--read-only` para permitir únicamente operaciones de herramientas de solo lectura.

3. Verificación de la configuración

Reinicie la CLI de Q Developer y, a continuación, compruebe las herramientas disponibles:

```
q /tools
```

Opción B: IDE de Kiro

Kiro es un espacio de trabajo de codificación centrado en la IA y con soporte del MCP integrado.

1. Apertura de la configuración de Kiro

- Abra Kiro.
- Vaya a Kiro → Settings y busque “MCP Config”.
- O pulse `Cmd+Shift+P,` (Mac) o `Ctrl+Shift+P,` (Windows y Linux) y busque “MCP Config”.

2. Adición de la configuración del servidor MCP

- Haga clic en “Open Workspace MCP Config” o en “Open User MCP Config” para editar el archivo de configuración de MCP directamente.

Asegúrese de sustituir el marcador de posición de región (`{region}`) por la región deseada.

Para 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}"
      ]
    }
  }
}
```

Para 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}"
      ]
    }
  }
}
```

Nota de seguridad: Se puede utilizar `--read-only` para permitir únicamente operaciones de herramientas de solo lectura.

Opción C: IDE de Cursor

Cursor proporciona soporte de MCP integrado con una interfaz de configuración gráfica.

1. Apertura de la configuración de Cursor

- Abra Cursor.
- Vaya a Settings → Cursor Settings → Tools & MCP
- O pulse `Cmd+Shift+P` (Mac) o `Ctrl+Shift+P` (Windows) y busque “MCP”.

2. Adición de la configuración del servidor MCP

- Haga clic en “New MCP Server”.

Cree el archivo de configuración si no existe. Asegúrese de sustituir el marcador de posición de región (`{region}`) por la región deseada.

Para 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}"
      ]
    }
  }
}
```

Para 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}"
      ]
    }
  }
}
```

Nota de seguridad: Se puede utilizar `--read-only` para permitir únicamente operaciones de herramientas de solo lectura.

3. Reinicio de Cursor

Cierre y vuelva a abrir Cursor para que los cambios surtan efecto.

4. Comprobación en el chat de Cursor

Abra el panel de chat e intente lo siguiente:

```
What EKS MCP tools are available?
```

Debería aparecer una lista completa de las herramientas de administración de EKS disponibles.

Opción D: Cline (extensión de VS Code)

Cline es una popular extensión de VS Code que lleva la asistencia de la IA directamente a su editor.

1. Apertura de la configuración de Cline

- Abra Cline.
- Pulse `Cmd+Shift+P` (Mac) o `Ctrl+Shift+P` (Windows) y busque “MCP”.

2. Adición de la configuración del servidor MCP

- Haga clic en “Add Server”.
- Haga clic en “Open User Configuration”.

Cree el archivo de configuración si no existe. Asegúrese de sustituir el marcador de posición de región (`{region}`) por la región deseada.

Para 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}"
      ]
    }
  }
}
```

Para 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}"
      ]
    }
  }
}
```

Nota de seguridad: Se puede utilizar `--read-only` para permitir únicamente operaciones de herramientas de solo lectura.

2. Recarga de VS Code

Pulse `Cmd+Shift+P` o `Ctrl+Shift+P` y seleccione “Developer: Reload Window”.

3. Verificación de la configuración

Abra Cline y pregunte lo siguiente:

```
List the available MCP tools for EKS
```

Paso 2 (opcional): creación de una política de “escritura”

Si lo desea, puede crear una política de IAM administrada por el cliente que proporcione acceso completo al servidor MCP de Amazon EKS. Esta política otorga permisos para usar todas las herramientas del servidor MCP de EKS, lo que incluye tanto las herramientas con privilegios que pueden implicar operaciones de escritura como las herramientas de solo lectura. Tenga en cuenta que, en esta política, se incluyen los permisos de alto riesgo (todos aquellos que contengan Delete* o un recurso de IAM sin restricciones), ya que son necesarios para configurar o desmontar los recursos del clúster en la herramienta manage_eks_stacks.

```
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\": \"*\"}]}"
```

Paso 3: comprobación de la configuración

Prueba de conexión

Haga una pregunta sencilla a su asistente de IA para verificar la conexión:

```
List all EKS clusters in my {aws} account
```

Debería aparecer una lista de los clústeres de EKS.

Paso 4: ejecución de las primeras tareas

Ejemplo 1: exploración de los clústeres

```
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
```

Ejemplo 2: comprobación de los recursos de Kubernetes

```
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
```

Ejemplo 3: solución de problemas

```
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
```

Ejemplo 4: creación de recursos (si el modo de “escritura” está activado)

```
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
```

Configuraciones comunes

Escenario 1: varios perfiles de AWS

Si trabaja con varias cuentas de AWS, cree configuraciones de servidor MCP independientes.

Para 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"
      ]
    }
  }
}
```

Para 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"
      ]
    }
  }
}
```

Escenario 2: solo lectura para producción

Cree una configuración de solo lectura para los entornos de producción.

Para 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"
      ]
    }
  }
}
```

Para 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"
      ]
    }
  }
}
```

Escenario 3: desarrollo con acceso completo

Para entornos de desarrollo con acceso de escritura completo.

Para 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"
      ]
    }
  }
}
```

Para 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"
      ]
    }
```

```
  }
}
```

Consideraciones

Seguridad

No divulgue secretos ni información confidencial a través de los mecanismos de entrada permitidos:

- No incluya secretos ni credenciales en los archivos YAML aplicados con apply_yaml.
- No transfiera información confidencial directamente al modelo en las petición.
- No incluya secretos en las plantillas de CloudFormation ni en los manifiestos de aplicaciones.
- Evite usar herramientas de MCP para crear secretos de Kubernetes, ya que esto requeriría proporcionar los datos secretos al modelo.
- Evite registrar información confidencial en los registros de las aplicaciones dentro de pods de Kubernetes.

Seguridad del contenido de YAML:

- Utilice archivos YAML solo de orígenes fiables.
- El servidor depende de la validación de la API de Kubernetes para el contenido de YAML y no lleva a cabo su propia validación.
- Audite los archivos YAML antes de aplicarlos al clúster.

En lugar de transferir los secretos a través del MCP, haga lo siguiente:

- Utilice AWS Secrets Manager o el Almacén de parámetros para almacenar información confidencial.
- Configure el RBAC de Kubernetes adecuado para las cuentas de servicio.
- Utilice roles de IAM para cuentas de servicio (IRSA) para acceder al servicio de AWS desde los pods.

Tema siguiente

Para una lista completa de configuraciones y herramientas, consulte the section called “Herramientas”.

Referencia de la configuración de la herramienta del servidor MCP de Amazon EKS

En esta guía, se muestran todas las configuraciones disponibles para la herramienta de cliente mcp-proxy-for-aws que le permite conectarse al servidor MCP de Amazon EKS completamente administrado desde su IDE.

> **Note**
>
> El servidor MCP de Amazon EKS se encuentra en versión preliminar para Amazon EKS y está sujeto a cambios.

Ejemplo

```
{
  "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"
      ]
    }
  }
}
```

Permisos de IAM

El rol utilizado para conectarse al servidor MCP requiere los permisos **`eks-mcp:InvokeMcp`** para la inicialización y la recuperación de información sobre las herramientas disponibles. **`eks-`**

`mcp:CallReadOnlyTool` es necesario para el uso de herramientas de solo lectura y **`eks-mcp:CallPrivilegedTool`** es necesario para el uso de herramientas de acceso completo (escritura).

Variables de entorno

`AWS_PROFILE` (opcional): nombre del perfil de credenciales de AWS que se utilizará, se puede anular mediante el argumento de la línea de comandos `--profile`.

- Ejemplo:: `export AWS_PROFILE=production`

`AWS_REGION` (opcional): región de AWS para la firma de SigV4, el valor predeterminado es `us-west-2` si no está establecido.

- Ejemplo:: `export AWS_REGION=us-east-1`

Argumentos

URL del punto de conexión de MCP de SigV4 (obligatorio): URL del punto de conexión del MCP al que conectarse.

- Ejemplo:: https://eks-mcp.us-west-2.api.aws/mcp

`--service` (opcional): nombre del servicio de AWS para la firma de SigV4, se detecta automáticamente desde el nombre de host del punto de conexión si no se proporciona.

- Ejemplo:: `--service eks-mcp`

`--profile` (opcional): perfil de credenciales de AWS que se utilizará. Si no se especifica, se utiliza de manera predeterminada la variable de entorno `AWS_PROFILE`.

- Ejemplo:: `--profile production`

`--region`: región de AWS que se utilizará. Utiliza la variable de entorno `AWS_REGION` si no está establecida, el valor predeterminado es `us-east-1`.

- Ejemplo:: `--region us-west-2`

`--read-only` (opcional): deshabilite las herramientas que puedan requerir permisos de escritura (las herramientas que NO requieren permisos de escritura se anotan con readOnlyHint=true). De manera predeterminada, todas las herramientas están habilitadas.

- Ejemplo:: `--read-only`

Para más opciones de configuración, consulte Configuration parameters.

Herramientas

El servidor expone las siguientes herramientas de MCP.

Herramientas de solo lectura

En esta sección, se describen las herramientas de solo lectura disponibles para el servidor MCP de EKS. Tenga en cuenta que todas las operaciones de la API de Kubernetes de solo lectura pueden acceder a lo siguiente:

- Clústeres privados (consulte Punto de conexión privado del clúster)
- Clústeres públicos

search_eks_documentation Busca en la documentación de EKS para obtener información y orientación actualizadas. Esta herramienta proporciona acceso a la documentación más reciente de EKS, lo que incluye las nuevas características y las mejoras recientes que los agentes tal vez no conozcan.

Parámetros:

- query (obligatorio): pregunta específica o consulta de búsqueda relacionada con la documentación, las características o las prácticas recomendadas de EKS.
- limit (opcional): número máximo de resultados de documentación que se deben devolver (entre 1 y 10). Valor predeterminado: 5.

search_eks_troubleshooting_guide Busca en la Guía de solución de problemas de EKS para obtener información sobre solución de problemas en función de una consulta. Ayuda a identificar problemas comunes y proporciona soluciones paso a paso.

Parámetros:

- query (obligatorio): pregunta específica o descripción del problema relacionada con la solución de problemas de EKS.

describe_eks_resource Recupera información detallada sobre un recurso del clúster de EKS específico, como la configuración, el estado o los metadatos.

Parámetros:

- cluster_name (obligatorio): nombre del clúster de EKS (obligatorio para los recursos del ámbito del clúster).
- resource_type (obligatorio): el tipo de recurso de EKS que se describirá. Valores válidos:
- accessentry (necesita cluster_name y resource_name como principalArn)
- addon (necesita cluster_name y resource_name como nombre del complemento)
- cluster (necesita cluster_name), nodegroup (necesita cluster_name y resource_name como nombre del grupo de nodos).
- resource_name (opcional): nombre del recurso específico que se describirá (obligatorio para la mayoría de los tipos de recursos).

list_eks_resources Enumera los recursos de EKS de un tipo específico y devuelve un resumen de todos los recursos del tipo especificado que sean accesibles.

Parámetros:

- resource_type (obligatorio): el tipo de recurso de EKS que se enumerará. Valores válidos:
- accessentry (necesita cluster_name)
- addon (necesita cluster_name)
- cluster (no se necesitan parámetros adicionales)
- nodegroup (necesita cluster_name).
- cluster_name (opcional): nombre del clúster de EKS (obligatorio para los recursos del ámbito del clúster).

get_eks_insights Recupera la información del clúster de EKS y recomendaciones para la optimización. Proporciona información procesable para la seguridad, el rendimiento y la optimización de costos basada en las prácticas recomendadas de AWS y el análisis del clúster.

Parámetros:

- cluster_name (obligatorio): nombre del clúster de EKS.
- category (opcional): categoría opcional por la que se filtrará la información (por ejemplo, “MISCONFIGURATION” o “UPGRADE_READINESS”).
- insight_id (opcional): ID opcional de información específica para el que se obtendrá información detallada.
- next_token (opcional): token opcional de paginación para obtener el siguiente conjunto de resultados.

get_eks_vpc_config Recupera la configuración de la VPC de un clúster de EKS, lo que incluye las subredes, las tablas de enrutamiento y la conectividad de red.

Parámetros:

- cluster_name (obligatorio): nombre del clúster de EKS para el que se obtendrá la configuración de la VPC.
- vpc_id (opcional): ID de la VPC específica que se consultará (opcional, se usará la VPC del clúster si no se especifica).

get_k8s_events Recupera eventos de Kubernetes relacionados con los recursos específicos para la solución de problemas y la supervisión.

Parámetros:

- cluster_name (obligatorio): nombre del clúster de EKS en el que se encuentra el recurso.
- kind (obligatorio): tipo del objeto implicado (por ejemplo, “Pod”, “Deployment”, “Service”). Debe coincidir exactamente con el tipo de recurso.
- name (obligatorio): nombre del objeto implicado para el que se obtendrán eventos.
- namespace (opcional): espacio de nombres del objeto implicado. Obligatorio para los recursos con espacios de nombres (como “Pods” o “Deployments”). No es obligatorio para los recursos del ámbito del clúster (como “Nodes” o “PersistentVolumes”).

get_pod_logs Recupera registros de pods en un clúster de EKS con opciones de filtrado.

Parámetros:

- cluster_name (obligatorio): nombre del clúster de EKS en el que se ejecuta el pod.
- namespace (obligatorio): espacio de nombres de Kubernetes en el que se encuentra el pod.
- pod_name (obligatorio): nombre del pod del que se recuperarán los registros.
- container_name (opcional): nombre del contenedor específico del que se obtendrán los registros. Obligatorio solo si el pod contiene varios contenedores.
- limit_bytes (opcional): cantidad máxima de bytes que se devolverán. Valor predeterminado: 10 KB (10 240 bytes).
- previous (opcional): devuelve los registros de un contenedor terminado anteriormente, (el valor predeterminado es “false”). Es útil para obtener los registros de los pods que se están reiniciando.
- since_seconds (opcional): solo devuelve los registros que sean más recientes que la cantidad de segundos especificada. Es útil para obtener registros recientes sin recuperar todo el historial.
- tail_lines (opcional): cantidad de líneas que se devolverán al final de los registros. Valor predeterminado: 100.

list_api_versions Muestra todas las versiones de la API disponibles en el clúster de Kubernetes especificado.

Parámetros:

- cluster_name (obligatorio): nombre del clúster de EKS.

list_k8s_resources Enumera los recursos de Kubernetes de un tipo específico en un clúster de EKS.

Parámetros:

- cluster_name (obligatorio): nombre del clúster de EKS en el que se encuentran los recursos.
- kind (obligatorio): tipo de los recursos de Kubernetes que se enumerarán (por ejemplo, “Pod”, “Service”, “Deployment”). Utilice la herramienta `list_api_versions` para encontrar los tipos de recursos disponibles.
- api_version (obligatorio): versión de la API de los recursos de Kubernetes (por ejemplo, “v1”, “apps/v1”, “networking.k8s.io/v1”). Utilice la herramienta `list_api_versions` para encontrar las versiones de la API disponibles.
- field_selector (opcional): selector de campos para filtrar recursos (por ejemplo, “metadata.name=my-pod,status.phase=Running”). Utiliza la misma sintaxis que la marca --field-selector de kubectl.

- label_selector (opcional): selector de etiquetas para filtrar recursos (por ejemplo, “app=nginx,tier=frontend”). Utiliza la misma sintaxis que la marca `--selector` de kubectl.
- namespace (opcional): espacio de nombres de los recursos de Kubernetes que se enumerarán. Si no se proporciona, los recursos se enumerarán en todos los espacios de nombres (en el caso de los recursos con espacios de nombres).

read_k8s_resource Recupera información detallada sobre un recurso de Kubernetes específico en un clúster de EKS.

Parámetros:

- api_version (obligatorio): versión de la API del recurso de Kubernetes (por ejemplo, “v1”, “apps/v1”, “networking.k8s.io/v1”).
- cluster_name (obligatorio): nombre del clúster de EKS en el que se encuentra el recurso.
- kind (obligatorio): tipo del recurso de Kubernetes (por ejemplo, “Pod”, “Service”, “Deployment”).
- name (obligatorio): nombre del recurso de Kubernetes que se leerá.
- namespace (opcional): espacio de nombres del recurso de Kubernetes. Obligatorio para los recursos con espacio de nombres. No es obligatorio para los recursos del ámbito del clúster (como “Nodes” o “PersistentVolumes”).

generate_app_manifest Genera manifiestos de servicio e implementación estandarizados de Kubernetes para aplicaciones incluidas en contenedores.

Parámetros:

- app_name (obligatorio): nombre de la aplicación. Se utiliza para los nombres de implementación y servicio, así como para las etiquetas.
- image_uri (obligatorio): URI completo de la imagen de ECR con etiqueta (por ejemplo, 123456789012.dkr.ecr.region.amazonaws.com/repo:tag). Debe incluir la ruta y la etiqueta completas del repositorio.
- load_balancer_scheme (opcional): esquema del equilibrador de carga de AWS. Valores válidos:
- “internal” (solo VPC privada)
- “internet-facing” (acceso público)
- Valor predeterminado: “internal”.

- cpu (opcional): solicitud de CPU para cada contenedor (por ejemplo, “100m” para núcleos de 0,1 CPU, “500m” para medio núcleo). Valor predeterminado: “100m”.
- memory (opcional): solicitud de memoria para cada contenedor (por ejemplo, “128Mi” para 128 MiB, “1Gi” para 1 GiB). Valor predeterminado: “128Mi”
- namespace (opcional): espacio de nombres de Kubernetes en el que se implementará la aplicación. Valor predeterminado: “default”.
- port (opcional): puerto del contenedor en el que escucha la aplicación. Valor predeterminado: 80
- replicas (opcional): cantidad de réplicas que se implementarán. Valor predeterminado: 2

get_cloudwatch_logs Consulta los registros de CloudWatch con un filtrado basado en los parámetros de entrada y soporte para grupos de registro estándar que se utilizan para la observabilidad de EKS.

Parámetros:

- cluster_name (obligatorio): nombre del clúster de EKS en el que se encuentra el recurso. Se utiliza para crear el nombre del grupo de registro de CloudWatch.
- resource_type (obligatorio): tipo de recurso para el que se buscarán registros. Valores válidos: +”pod”, “node”, “container”, “cluster”. Determina cómo se filtran los registros.
- log_type (obligatorio): tipo de registro que se consultará. Valores válidos:
- “application”: registros de la aplicación o el contenedor
- “host”: registros del sistema del nodo
- “performance”: registros de métricas de rendimiento
- “control-plane”: registros del plano de control de EKS
- “your-log-group-name”: proporcione un nombre de grupo de registro de CloudWatch personalizado directamente.
- resource_name (opcional): nombre del recurso que se buscará en los mensajes del registro (por ejemplo, nombre del pod, del nodo o del contenedor). Se utiliza para filtrar los registros de un recurso específico.
- minutes (opcional): cantidad de minutos para revisar registros. Valor predeterminado: 15. Se omite si se proporciona start_time. Utilice valores más pequeños para los problemas recientes y valores más altos para el análisis histórico.
- start_time (opcional): hora de inicio en formato ISO (por ejemplo, “2023-01-01T00:00:00Z”). Si se proporciona, anula el parámetro minutes.

- end_time (opcional): hora de finalización en formato ISO (por ejemplo, “2023-01-01T01:00:00Z”). Si no se proporciona, se establece la hora actual de forma predeterminada.
- fields (opcional): campos personalizados que se incluirán en los resultados de la consulta (el valor predeterminado es “@timestamp, @message”). Utilice la sintaxis de campo de Información de registros de CloudWatch.
- filter_pattern (opcional): patrón de filtrado adicional de Registros de CloudWatch que se aplicará. Utiliza la sintaxis de Información de registros de CloudWatch (por ejemplo, “ERROR”, “field=value”).
- limit (opcional): cantidad máxima de entradas de registro que se devolverán. Utilice valores más bajos (entre 10 y 50) para consultas más rápidas y valores más altos (entre 100 y 1000) para obtener resultados más completos. Los valores más altos pueden repercutir en el desempeño.

get_cloudwatch_metrics Recupera puntos de datos y métricas de CloudWatch para el análisis de rendimiento y la supervisión del clúster de EKS. Gestiona las métricas de Información de contenedores, las métricas personalizadas y los periodos de tiempo y las dimensiones configurables.

Parámetros:

- cluster_name (obligatorio): nombre del clúster de EKS para el que se obtendrán las métricas.
- dimensions (obligatorio): dimensiones que se utilizarán para la consulta de métricas de CloudWatch como cadena JSON. Debe incluir las dimensiones adecuadas para el tipo de recurso y la métrica (por ejemplo, “{"ClusterName": "my-cluster", "PodName": "my-pod", "Namespace": "default"}”).
- metric_name (obligatorio): nombre de la métrica que se recuperará. Ejemplos comunes:
- `cpu_usage_total`: uso total de la CPU
- `memory_rss`: uso de memoria del tamaño del conjunto residente
- `network_rx_bytes`: bytes de red recibidos
- `network_tx_bytes`: bytes de red transmitidos
- namespace (obligatorio): espacio de nombres de CloudWatch en el que se almacena la métrica. Valores comunes:
- `"ContainerInsights"`: para métricas de contenedor
- `"AWS/EC2"`: para métricas de instancia de EC2
- `"AWS/EKS"`: para métricas de plano de control de EKS

- minutes (opcional): cantidad de minutos para revisar métricas. Valor predeterminado: 15. Se omite si se proporciona start_time.
- start_time (opcional): hora de inicio en formato ISO (por ejemplo, “2023-01-01T00:00:00Z”). Si se proporciona, anula el parámetro minutes.
- end_time (opcional): hora de finalización en formato ISO (por ejemplo, “2023-01-01T01:00:00Z”). Si no se proporciona, se establece la hora actual de forma predeterminada.
- limit (opcional): cantidad máxima de puntos de datos que se devolverán. Los valores más altos (entre 100 y 1000) proporcionan datos más detallados, pero pueden afectar al rendimiento. Valor predeterminado: 50.
- period (opcional): periodo en segundos para los puntos de datos de métricas. Valor predeterminado: 60 (1 minuto). Los valores más bajos (entre 1 y 60) proporcionan una resolución más alta, pero es posible que haya menos disponibilidad.
- stat (opcional): estadística que se utilizará para la agregación de métricas. Valor predeterminado: “Average”. Valores válidos:
- `Average`: valor medio durante el periodo
- `Sum:`: valor total durante el periodo
- `Maximum`: valor más alto durante el periodo
- `Minimum`: valor más bajo durante el periodo
- `SampleCount`: cantidad de muestras durante el periodo

get_eks_metrics_guidance Permite obtener orientación sobre las métricas de CloudWatch para tipos de recursos específicos en clústeres de EKS. Es útil para el agente al determinar las dimensiones correctas que debe utilizar con la herramienta get_cloudwatch_metrics.

Parámetros:

- resource_type (obligatorio): tipo de recurso para el que se obtendrán las métricas (clúster, nodo, pod, espacio de nombres, servicio).

get_policies_for_role Recupera todas las políticas adjuntas a un rol de IAM especificado, lo que incluye la política para asumir roles, las políticas administradas y las políticas en línea.

Parámetros:

- role_name (obligatorio): nombre del rol de IAM para el que se obtendrán las políticas. El rol debe existir en su cuenta de AWS.

Herramientas de acceso completo (escritura)

En esta sección, se describen las herramientas de solo lectura disponibles para el servidor MCP de EKS. Tenga en cuenta que (a fecha de “hoy”) todas las operaciones de escritura de la API de Kubernetes pueden acceder solo a lo siguiente:

- Clústeres públicos (endpointPublicAccess=true)

manage_k8s_resource Administra un solo recurso de Kubernetes con operaciones de escritura (“create”, “update”, “patch” o “delete”).

Parámetros:

- operation (obligatorio): operación que se llevará a cabo en el recurso. Valores válidos:
- `create:`: creación de un nuevo recurso
- `replace`: sustitución de un recurso existente
- `patch`: actualización de campos específicos de un recurso existente
- `delete`: eliminación de un recurso existente
- Nota: Utilice read_k8s_resource para leer recursos y list_k8s_resources para enumerar varios recursos.
- cluster_name (obligatorio): nombre del clúster de EKS en el que se encuentra o se creará el recurso.
- kind (obligatorio): tipo del recurso de Kubernetes (por ejemplo, “Pod”, “Service”, “Deployment”).
- api_version (obligatorio): versión de la API del recurso de Kubernetes (por ejemplo, “v1”, “apps/v1”, “networking.k8s.io/v1”).
- body (opcional): definición del recurso como cadena JSON. Obligatorio para las operaciones “create”, “replace” y “patch”. Para “create” y “replace”, debe ser una definición de recurso completa. Para “patch”, solo puede contener los campos que se actualizarán.
- name (opcional): nombre del recurso de Kubernetes. Obligatorio para todas las operaciones excepto “create” (donde se puede especificar en el cuerpo).

- namespace (opcional): espacio de nombres del recurso de Kubernetes. Obligatorio para los recursos con espacio de nombres. No es obligatorio para los recursos del ámbito del clúster (como “Nodes” o “PersistentVolumes”).

apply_yaml Aplica los manifiestos YAML de Kubernetes a un clúster de EKS.

Parámetros:

- cluster_name (obligatorio): nombre del clúster de EKS en el que se crearán o actualizarán los recursos.
- namespace (obligatorio): espacio de nombres de Kubernetes al que se aplicarán los recursos. Se utilizará para los recursos con espacio de nombres que no especifiquen ningún espacio de nombres.
- yaml_content (obligatorio): contenido YAML que se aplicará al clúster. Puede contener varios documentos separados por “---”.
- force (opcional): si se deben actualizar recursos que ya existan (similar a “apply” en kubectl). Establézcalo en “false” para crear solo nuevos recursos.

manage_eks_stacks Administra las pilas de CloudFormation de EKS con operaciones para generar plantillas, implementar, describir y eliminar clústeres de EKS y su infraestructura subyacente.
La creación del clúster suele tardar entre 15 y 20 minutos en completarse. Para las operaciones “deploy” y “delete”, la pila debe haberse creado con esta herramienta (es decir, debe tener la etiqueta CreatedBy=EksMcpServer).

Parámetros:

- cluster_name (obligatorio): nombre del clúster de EKS (para las operaciones “generate”, “deploy”, “describe” y “delete”). Este nombre se utilizará para derivar el nombre de la pila de CloudFormation y se incrustará en los recursos del clúster.
- operación (obligatorio): operación que se llevará a cabo. Valores válidos:
- `generate`: generación de una plantilla de CloudFormation
- `deploy`: implementación de una pila de CloudFormation (necesita `template_content`)
- `describe`: descripción o lectura de una pila de CloudFormation (solo lectura)
- `delete`: eliminación de una pila de CloudFormation

- template_content (opcional): contenido de la plantilla de CloudFormation (para las operaciones “deploy). Debe ser el contenido completo de la plantilla YAML o JSON. Admite contenido YAML de varios documentos y recursos únicos separados por “---”.

add_inline_policy Agregar una nueva política en línea a un rol de IAM.

Parámetros:

- permissions (obligatorio): permisos que se incluirán en la política como cadenas JSON que representan instrucciones de políticas de IAM. Puede ser una sola cadena JSON o un conjunto de cadenas JSON.
- policy_name (obligatorio): nombre de la política en línea que se creará. Debe ser único dentro del rol.
- role_name (obligatorio): nombre del rol de IAM al que se agregará la política. El rol debe existir.

Uso de Amazon Q Developer en la consola de Amazon EKS

Amazon Elastic Kubernetes Service (EKS) se integra con Amazon Q para solucionar problemas con tecnología de IA directamente en la Consola de administración de AWS. Esta integración lo ayuda a investigar y resolver rápidamente problemas de clústeres, planos de control, nodos y cargas de trabajo con la ayuda de la IA de Amazon Q.

Funcionamiento

La consola de Amazon EKS muestra los botones Inspeccionar con Amazon Q contextualmente junto con los errores o problemas en toda la consola. Al hacer clic en este botón, Amazon Q analiza automáticamente el problema y abre un panel de chat en la parte derecha de la consola con los resultados de la investigación, el análisis de la causa raíz y los pasos de mitigación sugeridos.

La integración aparece en las siguientes ubicaciones de la consola de Amazon EKS:

- Estado del clúster: investigue los problemas de estado del clúster y los mensajes de estado en la pestaña Estado del clúster del panel de observabilidad.
- Plano de control: solucione los errores y las advertencias de los componentes del plano de control en Supervisión del plano de control en el panel de observabilidad.
- Información sobre la actualización: analice los posibles bloqueos de actualización y los problemas de compatibilidad en Información sobre la actualización en el panel de observabilidad.

- Estado del nodo: investigue los problemas del nodo que afectan a la capacidad del clúster en Problemas de estado del nodo en el panel de observabilidad.
- Cargas de trabajo: analice los eventos de Kubernetes en los pods que indiquen errores o problemas.

Uso de Amazon Q para la solución de problemas

Cómo investigar un problema con Amazon Q

1. Abra la consola de Amazon EKS.
2. Elija el nombre del clúster que se investigará.
3. Cuando encuentre un mensaje de error o un indicador de problema, busque el botón Inspeccionar con Amazon Q. El botón aparece contextualmente junto al problema o en la vista de detalles del estado del error.
4. Elija Inspeccionar con Amazon Q.
5. Amazon Q investiga el problema de forma automática y muestra el análisis en un panel de chat en la parte derecha de la consola.
6. Revise los resultados de la investigación, lo que incluye el análisis de la causa raíz y los pasos de mitigación sugeridos.
7. Para continuar la conversación, haga preguntas de seguimiento a Amazon Q sobre el problema.

Nota: La integración de Amazon Q solo aparece cuando hay un error, una advertencia o un problema que deba investigarse. No aparece cuando el estado de los recursos es correcto.

Consideraciones

Tenga en cuenta lo siguiente al utilizar Amazon Q con Amazon EKS:

- Operaciones de solo lectura: la integración de Amazon Q solo lleva a cabo operaciones de lectura en los recursos del clúster. No hace ninguna acción de mutación ni escritura en la configuración del clúster ni en las cargas de trabajo.
- Procesamiento entre regiones: Amazon Q puede procesar datos en todas las regiones de AWS para proporcionar análisis basados en IA. Para obtener más información sobre el procesamiento entre regiones, consulte Procesamiento entre regiones en la Guía del usuario de Amazon Q Developer.

- Solo en la Consola de administración de AWS: esta integración solo está disponible a través de la Consola de administración de AWS. No está disponible a través de la AWS CLI, las API de AWS ni las herramientas de infraestructura como de código.

Más información

Para obtener más información sobre cómo usar Amazon Q, consulte Chat con Amazon Q en la Guía del usuario de Amazon Q Developer.

Control de versiones en Amazon EKS

Esta sección se ha diseñado para ayudarle a aprender cómo funciona el control de versiones de Kubernetes en Amazon Elastic Kubernetes Service (EKS). Encontrará detalles sobre las versiones compatibles, nuestras políticas de obsolescencia y soporte, los procesos de actualización y las consideraciones clave a la hora de administrar las versiones de sus clústeres.

Temas

- Descripción del ciclo de vida de las versiones de Kubernetes en EKS
- Visualización de las versiones de la plataforma Amazon EKS para cada versión de Kubernetes

Descripción del ciclo de vida de las versiones de Kubernetes en EKS

Kubernetes evoluciona rápidamente con nuevas características, actualizaciones de diseño y correcciones de errores. La comunidad publica nuevas versiones secundarias de Kubernetes (tales como `1.33`) una vez cada cuatro meses de media. Amazon EKS sigue el ciclo de publicación y obsolescencia de las versiones anteriores para las versiones secundarias. Cuando haya nuevas versiones de Kubernetes disponibles en Amazon EKS, le recomendamos que actualice proactivamente los clústeres para que utilicen la versión más reciente disponible.

Una versión secundaria dispone de soporte estándar de Amazon EKS durante los primeros 14 meses después de su publicación. Cuando una versión supera la fecha de finalización del soporte estándar, pasa al periodo de soporte extendido durante los 12 meses siguientes. El soporte extendido permite permanecer en una versión específica de Kubernetes durante más tiempo a cambio de un costo adicional por hora de clúster. Si no ha actualizado su clúster antes de que concluya el período de soporte extendido, el clúster se actualiza automáticamente a la versión extendida más antigua que tenga soporte actualmente.

El soporte extendido está habilitado de forma predeterminada. Para deshabilitar, consulte Deshabilitación del soporte extendido de EKS.

Se recomienda crear el clúster con la última versión de Kubernetes disponible compatible con Amazon EKS. Si su aplicación requiere una versión específica de Kubernetes, puede seleccionar versiones anteriores. Puede crear nuevos clústeres de Amazon EKS en cualquier versión para la que se ofrezca soporte estándar o extendido.

Versiones disponibles con soporte estándar

Las siguientes versiones de Kubernetes están disponibles actualmente con soporte estándar de Amazon EKS:

- `1.34`
- `1.33`
- `1.32`

Para ver cambios importantes que debe conocer sobre cada versión con soporte estándar, consulte Kubernetes versions standard support.

Versiones disponibles con soporte extendido

Las siguientes versiones de Kubernetes están disponibles actualmente con soporte extendido de Amazon EKS:

- `1.31`
- `1.30`
- `1.29`

Para ver cambios importantes que debe conocer sobre cada versión con soporte extendido, consulte Kubernetes versions extended support.

Calendario de lanzamiento de Amazon EKS Kubernetes

En la siguiente tabla aparecen las fechas importantes de publicación y soporte que deben tenerse en cuenta para cada versión de Kubernetes. La facturación del soporte extendido comienza a partir del día en que la versión finaliza el soporte estándar, en la zona horaria UTC+0. Las fechas en la siguiente tabla utilizan la zona horaria UTC+0.

> **Note**
>
> Las fechas con solo un mes y un año son aproximadas y se actualizan con una fecha exacta cuando se conoce.

Para recibir notificaciones de todos los cambios en el archivo de origen de esta página de documentación específica, puede suscribirse a la siguiente URL con un lector de RSS:

```
https://github.com/awsdocs/amazon-eks-user-guide/commits/mainline/latest/ug/clusters/kubernetes-versions.adoc.atom
```

Versión de Kubernetes	Versión anterior	Versión de Amazon EKS	Fecha de finalización del soporte estándar	Fecha de finalización del soporte extendido
1.34	27 de agosto de 2025	2 de octubre de 2025	2 de diciembre de 2026	2 de diciembre de 2027
1.33	23 de abril de 2025	29 de mayo de 2025	29 de julio de 2026	29 de julio de 2027
1.32	11 de diciembre de 2024	23 de enero de 2025	23 de marzo de 2026	23 de marzo de 2027
1.31	13 de agosto de 2024	26 de septiembre de 2024	26 de noviembre de 2025	26 de noviembre de 2026
1.30	17 de abril de 2024	23 de mayo de 2024	23 de julio de 2025	23 de julio de 2026
1.29	13 de diciembre de 2023	23 de enero de 2024	23 de marzo de 2025	23 de marzo de 2026

Obtenga información sobre la versión con AWS CLI

Puede utilizar AWS CLI para obtener información sobre las versiones de Kubernetes disponibles en EKS, como la fecha de finalización del soporte estándar.

Para obtener información sobre las versiones de Kubernetes disponibles en EKS mediante AWS CLI

1. Abra el terminal.

2. Asegúrese de que ha instalado y configurado AWS CLI. Para obtener más información, consulte Instalación o actualización de la versión más reciente de la CLI.
3. Use el siguiente comando:

```
aws eks describe-cluster-versions
```

4. El comando devolverá una salida JSON con detalles sobre las versiones de clúster disponibles. A continuación, se muestra un ejemplo del resultado:

```
{
    "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"
        }
    ]
}
```

El resultado proporciona la siguiente información sobre cada versión del clúster:

- `clusterVersion`: la versión de Kubernetes del clúster de EKS
- `clusterType`: el tipo de clúster (por ejemplo, “eks”)
- `defaultPlatformVersion`: la versión de la plataforma de EKS predeterminada
- `defaultVersion`: si se trata de la versión predeterminada
- `releaseDate`: la fecha en que se publicó esta versión
- `endOfStandardSupportDate`: la fecha en que finalizará el soporte estándar
- `endOfExtendedSupportDate`: la fecha en que finalizará el soporte ampliado
- `status`: el estado de soporte actual de la versión, por ejemplo `STANDARD_SUPPORT` o `EXTENDED_SUPPORT`
- `kubernetesPatchVersion`: la versión específica de la revisión de Kubernetes

Preguntas frecuentes sobre las versiones de Amazon EKS

Cuántas versiones de Kubernetes con soporte estándar hay disponibles?

En línea con el soporte que ofrece la comunidad de Kubernetes para las versiones de Kubernetes, Amazon EKS se compromete a proporcionar soporte a, por lo menos, tres versiones de Kubernetes en todo momento. Se anunciará la fecha de finalización del soporte estándar de una determinada versión secundaria de Kubernetes con una antelación mínima de 60 días. Debido al proceso de cualificación y lanzamiento de Amazon EKS de versiones de Kubernetes nuevas, la fecha de finalización del soporte estándar de la versión de Kubernetes en Amazon EKS será después de la fecha en que el proyecto de Kubernetes deje de ser compatible con la versión anterior.

Durante cuánto tiempo recibe soporte estándar una versión de Kubernetes por parte de Amazon EKS?

Una versión de Kubernetes recibe soporte estándar durante 14 meses después de encontrarse disponible por primera vez en Amazon EKS. Esto es cierto incluso si la versión anterior de Kubernetes ya no admite una versión disponible en Amazon EKS. Creamos parches de seguridad que se pueden aplicar a las versiones de Kubernetes que son compatibles con Amazon EKS.

Se me notifica cuándo finaliza el soporte estándar para una versión de Kubernetes en Amazon EKS?

Sí. Si alguno de los clústeres de su cuenta ejecuta una versión que está cerca del final del soporte, Amazon EKS envía un aviso a través del panel de AWS Health aproximadamente 12 meses después del lanzamiento de la versión de Kubernetes en Amazon EKS. El aviso incluye la fecha de finalización del soporte. Será como mínimo 60 días a partir de la fecha del aviso.

Qué características de Kubernetes son compatibles con Amazon EKS?

Amazon EKS admite todas las características disponibles con carácter general de la API de Kubernetes. Las nuevas API en versión beta no están habilitadas en los clústeres de forma predeterminada. Sin embargo, las API beta ya existentes y las nuevas versiones de las API beta existentes siguen habilitadas de forma predeterminada. Las características alfa no son compatibles.

Los grupos de nodos administrados de Amazon EKS se actualizan automáticamente junto con la versión del plano de control del clúster?

No. Un grupo de nodos administrados crea instancias de Amazon EC2 en su cuenta. Estas instancias no se actualizan de forma automática cuando usted o Amazon EKS actualizan su

plano de control. Para obtener más información, consulte the section called “Actualización”. Recomendamos mantener la misma versión de Kubernetes en el plano de control y los nodos.

Los grupos de nodos autoadministrados se actualizan automáticamente junto con la versión del plano de control del clúster?

No. Un grupo de nodos autoadministrados incluye instancias de Amazon EC2 en su cuenta. Estas instancias no se actualizan de forma automática cuando usted o Amazon EKS actualizan la versión del plano de control en su nombre. Un grupo de nodos autoadministrados no tiene indicaciones en la consola de que necesita actualizarse. Puede ver la versión de `kubelet` instalada en un nodo al seleccionar el nodo en la lista de Nodos en la pestaña Overview (Información general) del clúster para determinar qué nodos deben actualizarse. Debe actualizar los nodos de forma manual. Para obtener más información, consulte the section called “Métodos de actualización”.

El proyecto de Kubernetes comprueba la compatibilidad entre el plano de control y los nodos de hasta tres versiones secundarias. Por ejemplo, los nodos `1.30` continúan funcionando cuando se organicen mediante un plano de control `1.33`. No obstante, no se recomienda ejecutar un clúster con nodos que estén tres versiones secundarias por detrás del plano de control de forma constante. Para obtener más información, consulte la sección sobre la política de compatibilidad de versiones y diferencia de versiones de Kubernetes en la documentación de Kubernetes. Recomendamos mantener la misma versión de Kubernetes en el plano de control y los nodos.

Los pods que se ejecutan en Fargate se actualizan automáticamente con una actualización automática de la versión del plano de control de clúster?

No. Se recomienda encarecidamente ejecutar los pods de Fargate como parte de un controlador de replicación, tal como una implementación de Kubernetes. A continuación, lleve a cabo un reinicio continuo de todos los pods de Fargate. La versión nueva del pod de Fargate se implementa con una versión de `kubelet` que es la misma que la versión actualizada del plano de control de clúster. Para obtener más información, consulte Implementaciones en la documentación de Kubernetes.

> **Important**
>
> Si actualiza el plano de control, aún debe actualizar los nodos de Fargate por su cuenta. Para actualizar los nodos de Fargate, elimine el pod de Fargate representado por el nodo y vuelva a implementarlo. El nuevo pod se implementa con una versión de `kubelet` que es la misma versión del clúster.

Qué versiones de Kubernetes son compatibles con los nodos híbridos?

Los Nodos híbridos de Amazon EKS soportan las mismas versiones de Kubernetes que los clústeres de Amazon EKS con otros tipos de nodos informáticos, incluida la compatibilidad con versiones de Kubernetes estándar y ampliada. Los nodos híbridos no se actualizan automáticamente cuando se actualiza la versión del plano de control, por lo que será su responsabilidad mejorarlos. Para obtener más información, consulte the section called "Actualización de nodos híbridos".

Preguntas frecuentes sobre el soporte extendido de Amazon EKS

Los términos "soporte estándar" y "soporte extendido" son nuevos para mí. Qué significan esos términos?

El soporte estándar de una versión de Kubernetes en Amazon EKS comienza cuando se publica una versión de Kubernetes en Amazon EKS, y concluirá 14 meses después de la fecha de publicación. El soporte extendido de una versión de Kubernetes comenzará inmediatamente después de que finalice el soporte estándar, y concluirá al cabo de 12 meses a partir de ese momento. Por ejemplo, el soporte estándar de la versión `1.23` en Amazon EKS concluyó el 11 de octubre de 2023. El soporte extendido de la versión `1.23` comenzó el 12 de octubre de 2023 y concluyó el 11 de octubre de 2024.

Qué debo hacer para conseguir soporte extendido para los clústeres de Amazon EKS?

Deberá habilitar el soporte extendido (consulte EKS extended support) para su clúster cambiando la política de actualización del clúster a EXTENDIDO. De forma predeterminada, para todos los clústeres nuevos y existentes, la política de actualización se establece en EXTENDIDO, a menos que se especifique lo contrario. Consulte Cluster upgrade policy para ver la política de actualización de su clúster. El soporte estándar comenzará cuando se publique una versión de Kubernetes en Amazon EKS, y concluirá 14 meses después de la fecha de publicación. El soporte extendido de una versión de Kubernetes comenzará inmediatamente después de que finalice el soporte estándar, y concluirá al cabo de 12 meses a partir de ese momento.

Para qué versiones de Kubernetes se puede obtener soporte extendido?

Puede ejecutar clústeres en cualquier versión durante un máximo de 12 meses después de que concluya el soporte estándar para esa versión. Esto significa que cada versión recibirá soporte durante 26 meses en Amazon EKS (14 meses de soporte estándar más 12 meses de soporte extendido).

Qué sucede si no quiero usar el soporte extendido?

Si no desea recibir automáticamente soporte extendido, puede actualizar su clúster a una versión de Kubernetes que tenga soporte estándar de Amazon EKS. Para deshabilitar el soporte extendido, consulte Deshabilitación del soporte extendido de EKS. Nota: Si deshabilita el soporte extendido, se actualizará automáticamente el clúster a la siguiente versión cuando finalice el soporte estándar.

Qué sucederá cuando terminen los 12 meses de soporte extendido?

Los clústeres que se ejecuten en una versión de Kubernetes que haya completado su ciclo de vida de 26 meses (14 meses de soporte estándar más 12 meses de soporte extendido) se actualizarán automáticamente a la siguiente versión. La actualización automática solo incluye el plano de control de Kubernetes. Si tiene nodos de modo automático de EKS, es posible que se actualicen automáticamente. Los nodos autoadministrados y los grupos de nodos administrados por EKS se mantendrán en la versión anterior.

Cuando llegue la fecha de finalización del soporte extendido, ya no podrá crear nuevos clústeres de Amazon EKS con la versión no soportada. Amazon EKS actualiza los planos de control existentes a la primera versión admitida de forma automática mediante un proceso de implementación gradual tras la fecha de finalización del soporte. Después de la actualización automática del plano de control, asegúrese de actualizar los complementos del clúster y los nodos de Amazon EC2 de forma manual. Para obtener más información, consulte the section called “Actualización de una versión de Kubernetes”.

Cuándo se actualiza exactamente mi plano de control de manera automática después de la fecha de finalización del soporte extendido?

Amazon EKS no puede facilitar plazos concretos. Las actualizaciones automáticas pueden producirse en cualquier momento después de la fecha de finalización del soporte extendido. No recibirá ninguna notificación antes de la actualización. Recomendamos que actualice de manera proactiva su plano de control sin depender del proceso de actualización automática de Amazon EKS. Para obtener más información, consulte the section called “Actualización de una versión de Kubernetes”.

Puedo dejar mi plano de control en una versión de Kubernetes de forma indefinida?

No. La seguridad en la nube de AWS es la mayor prioridad. Pasado cierto punto (normalmente un año), la comunidad de Kubernetes deja de publicar parches de exposiciones y vulnerabilidades comunes (CVE) y desalienta el envío de CVE para versiones obsoletas. Esto significa que es posible que ni siquiera se denuncien las vulnerabilidades específicas de una

versión anterior de Kubernetes. Esto deja expuestos los clústeres sin aviso y sin opciones de corrección en caso de vulnerabilidad. Debido a esto, Amazon EKS no permite que los planos de control permanezcan en una versión que haya llegado al final del soporte extendido.

El soporte extendido conlleva un costo adicional?

Sí, los clústeres de Amazon EKS que se ejecuten con soporte extendido conllevan un costo adicional. Para obtener más información sobre los precios, consulte el Soporte ampliado de Amazon EKS para conocer los precios de las versiones de Kubernetes en el blog de AWS o nuestra página de precios.

Qué incluye el soporte extendido?

Los clústeres de Amazon EKS con soporte extendido reciben revisiones de seguridad continuas para el plano de control de Kubernetes. Además, Amazon EKS lanzará parches para los complementos CNI de Amazon VPC, `kube-proxy` y CoreDNS para las versiones con soporte extendido. Amazon EKS también lanzará parches para las AMI optimizadas de Amazon EKS publicadas por AWS para Amazon Linux, Bottlerocket y Windows, además de nodos de Fargate de Amazon EKS para esas versiones. Todos los clústeres con soporte extendido seguirán teniendo acceso a soporte técnico de AWS.

Existen limitaciones en cuanto a revisiones para componentes ajenos a Kubernetes en el soporte extendido?

Si bien el soporte extendido cubre todos los componentes específicos de Kubernetes de AWS, solo ofrecerá soporte a las AMI optimizadas de Amazon EKS publicadas por AWS para Amazon Linux, Bottlerocket y Windows en todo momento. Esto significa que, mientras utilice el soporte extendido, es posible que tenga componentes más recientes (tales como sistema operativo o kernel) en una AMI optimizada de Amazon EKS. Por ejemplo, cuando Amazon Linux 2 llegue al final de su ciclo de vida en 2025, las AMI de Amazon Linux optimizadas de Amazon EKS se crearán con un sistema operativo Amazon Linux más reciente. Amazon EKS anunciará y documentará discrepancias importantes en el ciclo de vida del soporte tales como esta para cada versión de Kubernetes.

Puedo crear nuevos clústeres con una versión con soporte extendido?

Sí.

Revisión de las notas de la versión estándar de Kubernetes con soporte extendido

En este tema se detallan cambios importantes que debe conocer sobre cada versión de Kubernetes con soporte estándar. Al actualizar, revise detenidamente los cambios que haya habido entre la versión antigua y la nueva de su clúster.

Kubernetes 1.34

Kubernetes `1.34` ya se encuentra disponible en Amazon EKS. Para obtener más información acerca de Kubernetes `1.34`, consulte el anuncio del lanzamiento oficial.

Important

- containerd se actualizó a 2.1 en la versión 1.34 para el lanzamiento.
  - Si tiene algún problema después de la actualización, consulte las Notas de la versión de containerd 2.1.
- AWS no publica una AMI de Amazon Linux 2 optimizada para EKS para Kubernetes 1.34.
  - AWS le anima a migrar a Amazon Linux 2023. Información sobre cómo the section called “Actualización a AL2023”.
  - Para obtener más información, consulte the section called “Obsolescencia de la AMI de Amazon Linux 2”.
- AppArmor está obsoleto en Kubernetes 1.34.
  - Recomendamos migrar a soluciones de seguridad de contenedores alternativas, como seccomp o Estándares de seguridad de pods.
- VolumeAttributesClass (VAC) pasa a GA en Kubernetes 1.34, lo que migra de la API beta (`storage.k8s.io/v1beta1`) a la API estable (`storage.k8s.io/v1`).
  - Si utiliza el controlador CSI de EBS con contenedores sidecar administrados de AWS (de Componentes de CSI en la galería de ECR), la modificación del volumen continuará funcionando sin problemas en los clústeres de las versiones 1.31 a 1.33 de EKS. AWS parcheará los contenedores sidecar para que sean compatibles con las API de VAC en versión beta hasta que finalice la compatibilidad con el estándar de EKS 1.33 (29 de julio de 2026).

- Si autoadministra sus contenedores sidecar de CSI, es posible que tenga que fijar versiones antiguas de sidecar en clústeres anteriores a la versión 1.34 para mantener la funcionalidad de VAC.
- Para utilizar las características de VolumeAttributesClass de GA (como la reversión de modificaciones), actualice a EKS 1.34 o una versión posterior.

- API principales de asignación dinámica de recursos (DRA) (GA): la asignación dinámica de recursos ha pasado a ser estable, lo que permite una administración eficiente de hardware especializado, como las GPU, mediante interfaces de asignación estandarizadas, lo que simplifica la administración de recursos para los aceleradores de hardware y mejora el uso de los recursos especializados.
- Tokens de ServiceAccount proyectados para Kubelet (beta): esta mejora aumenta la seguridad al utilizar credenciales de corta duración para la extracción de imágenes de contenedores en lugar de secretos de larga duración, lo que reduce el riesgo de exposición de las credenciales y refuerza la posición de seguridad general de los clústeres.
- Solicitudes y límites de recursos de pods (beta): esta característica simplifica la administración de recursos, ya que permite compartir grupos de recursos para pods de varios contenedores, lo que permite una asignación y un uso más eficientes de los recursos para aplicaciones complejas con varios contenedores.
- Recuento asignable de nodos de CSI mutables (beta): la puerta de características `MutableCSINodeAllocatableCount` está activada de forma predeterminada en EKS 1.34, lo que hace que el atributo de recuento del máximo de volúmenes adjuntables de CSINode sea mutable e ingresa un mecanismo para actualizarlo dinámicamente en función de la configuración del usuario en el controlador CSI. Estas actualizaciones se pueden activar mediante intervalos periódicos o mediante la detección de errores, lo que mejora la fiabilidad de la programación de pods con estado al abordar las discrepancias entre la capacidad de adjunción notificada y la real en los nodos.
  - Para obtener más información, consulte Kubernetes v1.34: Mutable CSI Node Allocatable Count en el Blog de Kubernetes.
- Aviso de obsolescencia, configuración del controlador cgroup: la configuración manual del controlador cgroup se sustituirá por la detección automática y quedará obsoleta.
  - Impacto para el cliente: si actualmente establece el indicador `--cgroup-driver` manualmente en la configuración de kubelet, debe prepararse para eliminar esta configuración.

- Acción necesaria: planifique actualizar los scripts de arranque de los nodos y las configuraciones de AMI personalizadas para eliminar la configuración manual del controlador cgroup antes de que se elimine la característica en una versión futura de Kubernetes.
- Para obtener más información, consulte la documentación del controlador cgroup.

Para completar el registro de cambios de Kubernetes `1.34`, consulte https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.34.md.

Kubernetes 1.33

Kubernetes `1.33` ya se encuentra disponible en Amazon EKS. Para obtener más información acerca de Kubernetes `1.33`, consulte el anuncio del lanzamiento oficial.

> **Important**
>
> - La API de Kubernetes de asignación dinámica de recursos en versión beta está habilitada.
>   - Esta API en versión beta mejora la experiencia de programación y supervisión de las cargas de trabajo que requieren recursos, como las GPU.
>   - La API en versión beta la define la comunidad de Kubernetes y puede cambiar en futuras versiones de Kubernetes.
>   - Revisa detenidamente Feature stages en la documentación de Kubernetes para entender las implicaciones del uso de las API en versión beta.
> - AWS no publica una AMI de Amazon Linux 2 optimizada para EKS para Kubernetes 1.33.
>   - AWS le anima a migrar a Amazon Linux 2023. Información sobre cómo the section called “Actualización a AL2023”.
>   - Para obtener más información, consulte the section called “Obsolescencia de la AMI de Amazon Linux 2”.

- Redimensionamiento de recursos de pods in situ (versión beta): el redimensionamiento de recursos in situ ha pasado a una versión beta, lo que permite actualizar dinámicamente los recursos de CPU y memoria de los pods existentes sin necesidad de reiniciarlos, lo que a su vez permite escalar verticalmente las cargas de trabajo con estado sin tiempo de inactividad y ajustar los recursos sin problemas en función de los patrones de tráfico.

- Los contenedores sidecar ahora son estables: los contenedores sidecar han pasado a ser estables, ya que se implementan como contenedores init especiales con `restartPolicy: Always` que se inician antes que los contenedores de aplicaciones, se ejecutan durante todo el ciclo de vida del pod y admiten sondas para la señalización del estado operativo.
  - Para obtener más información, consulte Sidecar Containers en la documentación de Kubernetes.
- Obsolescencia de la API de puntos de conexión: la API de puntos de conexión ya está oficialmente obsoleta y devolverá advertencias cuando se acceda a ella. Migre las cargas de trabajo y los scripts para utilizar la API de EndpointSlices, que admite características modernas, como las redes de doble pila, y gestiona varios EndpointSlices por servicio.
  - Para obtener más información, consulte Kubernetes v1.33: Continuing the transition from Endpoints to EndpointSlice en el Blog de Kubernetes.
- Compatibilidad con Elastic Fabric Adapter: el grupo de seguridad predeterminado para los clústeres de Amazon EKS ahora admite el tráfico de Elastic Fabric Adapter (EFA). El grupo de seguridad predeterminado tiene una nueva regla de salida que permite el tráfico de EFA con el destino del mismo grupo de seguridad. Esto permite el tráfico de EFA dentro del clúster.
  - Para obtener más información, consulte Elastic Fabric Adapter para cargas de trabajo de HPC e IA o ML en Amazon EC2 en la Guía del usuario de Amazon Elastic Compute Cloud.

Para completar el registro de cambios de Kubernetes `1.33`, consulte https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.33.md.

Kubernetes 1.32

Kubernetes `1.32` ya se encuentra disponible en Amazon EKS. Para obtener más información acerca de Kubernetes `1.32`, consulte el anuncio del lanzamiento oficial.

> **Important**
>
> - La versión de la API `flowcontrol.apiserver.k8s.io/v1beta3` de FlowSchema y PriorityLevelConfiguration se eliminó en la versión `1.32`. Si utiliza estas API, debe actualizar las configuraciones de modo que se utilice la última versión compatible antes de realizar la actualización.
> - ServiceAccount `metadata.annotations[kubernetes.io/enforce-mountable-secrets]` ha quedado obsoleta en la versión `1.32` y se eliminará en una próxima versión secundaria de Kubernetes. Se recomienda utilizar espacios de nombres separados para aislar el acceso a los secretos montados.

- La versión `1.32` de Kubernetes es la última versión para la que Amazon EKS lanzará las AMI de Amazon Linux 2 (AL2). A partir de la versión `1.33`, Amazon EKS seguirá lanzando AMI basadas en Amazon Linux 2023 (AL2023) y Bottlerocket.

- La característica Administrador de memoria ahora se encuentra disponible de forma general (GA) en la versión `1.32` de Kubernetes. Esta mejora permite una asignación de memoria más eficiente y predecible para las aplicaciones en contenedores, lo que es especialmente beneficioso para las cargas de trabajo con requisitos de memoria específicos.
- Los PersistentVolumeClaims (PVC) creados por StatefulSets ahora incluyen funcionalidad de limpieza automática. Cuando los PVC ya no sean necesarios, se eliminarán automáticamente, a la vez que se mantendrá la persistencia de los datos durante las actualizaciones de StatefulSet y las operaciones de mantenimiento de nodos. Esta característica simplifica la administración del almacenamiento y ayuda a evitar que los PVC queden huérfanos en el clúster.
- Se ha introducido la función de selección de campos de recursos personalizados, que permite a los desarrolladores agregar selectores de campos a recursos personalizados. Esta característica ofrece las mismas capacidades de filtrado para los objetos integrados de Kubernetes que para los recursos personalizados, lo que permite un filtrado de recursos más preciso y eficiente, además de fomentar las prácticas recomendadas en el diseño de API.

Para completar el registro de cambios de Kubernetes `1.32`, consulte https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.32.md.

Cambios en la autenticación anónima

A partir de Amazon EKS `1.32`, la autenticación anónima se restringe a los siguientes puntos de conexión de comprobación de estado del servidor de la API:

- `/healthz`
- `/livez`
- `/readyz`

Las solicitudes a cualquier otro punto de conexión que utilice el usuario `system:unauthenticated` recibirán una respuesta HTTP `401 Unauthorized`. Esta mejora de seguridad ayuda a prevenir accesos no deseados al clúster que podrían derivarse de configuraciones incorrectas en las políticas de control de acceso basado en roles (RBAC).

> **Note**
>
> El rol de RBAC `public-info-viewer` se mantiene válido para los puntos de conexión de comprobación de estado enumerados anteriormente.

Obsolescencia de la AMI de Amazon Linux 2

La versión `1.32` de Kubernetes es la última versión para la que Amazon EKS lanzó las AMI de AL2. A partir de la versión `1.33`, Amazon EKS seguirá lanzando AMI basadas en AL2023 y Bottlerocket. Para obtener más información, consulte the section called “Obsolescencia de las AMI AL2”.

Revisión de las notas de la versión para las versiones de Kubernetes con soporte extendido

Amazon EKS ofrece soporte para las versiones de Kubernetes durante más tiempo que el soporte ofrecido por la versión original de Kubernetes, con soporte estándar para versiones menores de Kubernetes durante 14 meses a partir de su lanzamiento en Amazon EKS y soporte extendido para versiones menores de Kubernetes por 12 meses adicionales (un total de 26 meses por versión).

En este tema se detallan cambios importantes que debe conocer sobre cada versión de Kubernetes con soporte extendido. Al actualizar, revise detenidamente los cambios que haya habido entre la versión antigua y la nueva de su clúster.

Kubernetes 1.31

Kubernetes `1.31` ya se encuentra disponible en Amazon EKS. Para obtener más información acerca de Kubernetes `1.31`, consulte el anuncio del lanzamiento oficial.

> **Important**
>
> - La marca `--keep-terminated-pod-volumes` de kubelet, obsoleta desde 2017, se ha eliminado como parte de la versión `1.31`. Este cambio afecta a la forma en que el kubelet gestiona los volúmenes de pods terminados. Si utiliza esta marca en las configuraciones de nodo, debe actualizar los scripts de arranque y las plantillas de lanzamiento para eliminarla antes de actualizar.

- La puerta y el recurso de la API de la característica `VolumeAttributesClass` en su versión beta están habilitados en la versión `1.31` de Amazon EKS. Esta característica permite a los operadores de clústeres modificar las propiedades mutables de los volúmenes persistentes (PV) administrados por controladores de CSI compatibles, incluido el controlador de CSI de Amazon EBS. Para aprovechar esta característica, asegúrese de que el controlador de CSI es compatible con la característica `VolumeAttributesClass` (para el controlador de CSI de Amazon EBS, actualice a la versión `1.35.0` o a una posterior para habilitar automáticamente la característica). Podrá crear objetos `VolumeAttributesClass` para definir los atributos de volumen deseados, como el tipo de volumen y el rendimiento, y asociarlos a las solicitudes de volumen persistente (PVC). Consulte la documentación oficial de Kubernetes, así como la documentación del controlador de CSI para obtener más información.
  - Para obtener más información sobre el Controlador de CSI de Amazon EBS, consulte the section called “Amazon EBS”.
- La compatibilidad de Kubernetes con AppArmor ha pasado a estable y ahora se encuentra disponible de forma general para uso público. Esta característica permite proteger a los contenedores con AppArmor al configurar el campo `appArmorProfile.type` en el `securityContext` del contenedor. Antes de la versión `1.30` de Kubernetes, AppArmor se controlaba mediante anotaciones. A partir de la versión `1.30`, se controla mediante campos. Para aprovechar esta característica, recomendamos dejar de utilizar anotaciones y, en su lugar, utilizar el campo `appArmorProfile.type` con el objetivo de garantizar que las cargas de trabajo sean compatibles.
- La característica de tiempo de transición de la última fase de PersistentVolume ha pasado a estable y ahora se encuentra disponible de forma general para uso público en la versión `1.31` de Kubernetes. Esta característica introduce un nuevo campo, `.status.lastTransitionTime`, en el PersistentVolumeStatus, que proporciona una marca de tiempo de la última vez que un PersistentVolume pasó a una fase diferente. Esta mejora permite realizar un mejor seguimiento y gestión de los PersistentVolumes, especialmente en situaciones en las que es importante comprender el ciclo de vida de los volúmenes.

Para completar el registro de cambios de Kubernetes `1.31`, consulte https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.31.md.

Kubernetes 1.30

Kubernetes `1.30` ya se encuentra disponible en Amazon EKS. Para obtener más información acerca de Kubernetes `1.30`, consulte el anuncio del lanzamiento oficial.

- A partir de la versión `1.30` o posterior de Amazon EKS, todos los grupos de nodos administrados recién creados utilizarán automáticamente Amazon Linux 2023 (AL2023) como sistema operativo de nodos de forma predeterminada. Para obtener más información sobre cómo especificar el sistema operativo de un grupo de nodos administrado, consulte the section called “Creación”.
- Con Amazon EKS `1.30`, la etiqueta `topology.k8s.aws/zone-id` se añade a los nodos de trabajo. Puede usar IDs de zona de disponibilidad (AZ IDs) para determinar la ubicación de los recursos de una cuenta respecto de los recursos de otra. Para obtener más información, consulte ID de zona de disponibilidad para los recursos de AWS en la Guía del usuario de AWS RAM.
- A partir de la versión `1.30`, Amazon EKS ya no incluye la anotación `default` en el recurso `gp2 StorageClass` aplicado a los clústeres recién creados. Esto no tiene ningún impacto si hace referencia a esta clase de almacenamiento por su nombre. Debe tomar medidas si confiaba en tener un `StorageClass` predeterminado en el clúster. Debe hacer referencia a `StorageClass` por su nombre `gp2`. Como alternativa, para implementar la clase de almacenamiento predeterminada recomendada por Amazon EBS, puede configurar el parámetro `defaultStorageClass.enabled` en verdadero al instalar la versión `1.31.0` o posterior de `aws-ebs-csi-driver add-on`.
- La política de IAM mínima requerida para el rol de IAM del clúster de Amazon EKS ha cambiado. La acción `ec2:DescribeAvailabilityZones` es obligatoria. Para obtener más información, consulte the section called “Rol de IAM de clúster”.

Para completar el registro de cambios de Kubernetes `1.30`, consulte https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.30.md.

Kubernetes 1.29

Kubernetes `1.29` ya se encuentra disponible en Amazon EKS. Para obtener más información acerca de Kubernetes `1.29`, consulte el anuncio del lanzamiento oficial.

> **Important**
>
> - La versión de la API `flowcontrol.apiserver.k8s.io/v1beta2` en desuso de `FlowSchema` y de `PriorityLevelConfiguration` ya no se ofrece en la versión `1.29` de Kubernetes. Si tiene manifiestos o software de cliente que utiliza el grupo de API beta en desuso, debe cambiarlos antes de actualizar a la versión `1.29`.

- El campo `.status.kubeProxyVersion` para los objetos de nodo ahora está en desuso y el proyecto de Kubernetes propone eliminarlo en una versión futura. El campo obsoleto no es preciso e históricamente ha sido administrado por `kubelet`, el cual, en realidad, no conoce la versión `kube-proxy` ni si se ejecuta `kube-proxy`. Si utilizó este campo en un software cliente, deje de hacerlo; la información no es fiable y el campo está en desuso.
- En Kubernetes `1.29`, para reducir la posible superficie expuesta a ataques, la característica `LegacyServiceAccountTokenCleanUp` etiqueta los tokens heredados basados en secretos generados automáticamente como no válidos si no se utilizaron durante mucho tiempo (1 año de forma predeterminada) y los elimina automáticamente si no se intenta usarlos por mucho tiempo después de marcarlos como no válidos (1 año adicional de forma predeterminada). Para identificar estos tokens, puede ejecutar lo siguiente:

```
kubectl get cm kube-apiserver-legacy-service-account-token-tracking -n kube-system
```

Para completar el registro de cambios de Kubernetes `1.29`, consulte https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.29.md#changelog-since-v1280.

Visualización del periodo actual de soporte del clúster

La sección sobre el periodo de soporte del clúster de la consola de AWS indica si el clúster cuenta actualmente con soporte estándar o extendido. Si el periodo de soporte del clúster es Soporte extendido, se le cobrará por el soporte extendido de EKS.

Para obtener más información sobre el soporte estándar y extendido, consulte Descripción del ciclo de vida de las versiones de Kubernetes en EKS.

1. Navegue a la página Clústeres de la sección de EKS de la consola de AWS. Confirme que la consola esté configurada en la misma región de AWS que el clúster que desee revisar.
2. Revise la columna Periodo de soporte. Si el valor es Soporte estándar hasta…, actualmente no se le cobra por el soporte extendido. Se encuentra dentro del periodo de soporte estándar. Si el valor es Soporte extendido…, actualmente se le cobra por el soporte extendido de este clúster.

> **Note**
>
> El periodo de soporte no se puede recuperar con la CLI ni la API de AWS.

Visualización de la política actual de actualización de clústeres

La política de actualización de un clúster determina lo que ocurre cuando un clúster llega al final del periodo de soporte estándar. Si su política de actualización es `EXTENDED`, el clúster no se actualizará automáticamente y entrará en el soporte extendido. Si su política de actualización es `STANDARD`, se actualizará automáticamente.

Los controles de Amazon EKS para la política de la versión de Kubernetes le permiten elegir el comportamiento al final del soporte estándar de sus clústeres de EKS. Con estos controles, puede decidir qué clústeres deben recibir soporte extendido y cuáles se deben actualizar automáticamente al finalizar el soporte estándar de una versión de Kubernetes.

Una versión secundaria dispone de soporte estándar de Amazon EKS durante los primeros 14 meses después de su publicación. Cuando una versión supera la fecha de finalización del soporte estándar, pasa al periodo de soporte extendido durante los 12 meses siguientes. El soporte extendido permite permanecer en una versión específica de Kubernetes durante más tiempo a cambio de un costo adicional por hora de clúster. Puede habilitar o deshabilitar el soporte extendido de un clúster de EKS. Si deshabilita el soporte extendido, AWS actualizará automáticamente el clúster a la siguiente versión cuando finalice el soporte estándar. Si habilita el soporte extendido, puede quedarse con la versión actual por un costo adicional durante un periodo de tiempo limitado. Planifique la actualización de su clúster de Kubernetes forma periódica, incluso si utiliza el soporte extendido.

Puede establecer la política de versiones de los clústeres nuevos y existentes mediante la propiedad `supportType`. Hay dos opciones que se pueden usar para establecer la política de soporte de versiones:

- **`STANDARD`** : su clúster de EKS puede actualizarse automáticamente al finalizar el soporte estándar. Con esta configuración, no se le cobrarán cargos por soporte extendido, pero su clúster de EKS se actualizará automáticamente a la siguiente versión de Kubernetes admitida en soporte estándar.
- **`EXTENDED`** : su clúster de EKS pasará a tener soporte extendido una vez que la versión de Kubernetes finalice el soporte estándar. Con esta configuración, incurrirá en cargos por soporte extendido. Puede actualizar su clúster a una versión de Kubernetes admitida por el soporte estándar para no incurrir en cargos por soporte extendido. Los clústeres que se ejecuten con soporte extendido podrán actualizarse automáticamente al finalizar este periodo.

El soporte extendido está habilitado de forma predeterminada tanto para los clústeres nuevos como para los existentes. Use la Consola de administración de AWS o la CLI de AWS para ver si el soporte extendido está habilitado en un clúster.

> **Important**
>
> Si desea que su clúster mantenga la versión de Kubernetes actual para aprovechar el periodo de soporte extendido, debe habilitar la política de actualización de soporte extendido antes de que finalice el periodo de soporte estándar.

Solo puede configurar la política de soporte de versiones de sus clústeres mientras se ejecute en la versión de Kubernetes con soporte estándar. Una vez que la versión entre en el soporte extendido, no podrá cambiar esta configuración hasta que utilice una versión con soporte estándar.

Por ejemplo, si su política de soporte de versiones es `standard`, no podrá cambiar esta configuración una vez que la versión de Kubernetes que se ejecuta en su clúster finalice el soporte estándar. Si su política de soporte de versiones es `extended`, no podrá cambiar esta configuración una vez que la versión de Kubernetes que se ejecuta en su clúster finalice el soporte estándar. Para cambiar la configuración de la política de soporte de versiones, el clúster se debe ejecutar en una versión de Kubernetes con soporte estándar.

Visualización de la política de actualización de clústeres (consola de AWS)

1. Navegue a la página Clústeres de la sección de EKS de la consola de AWS. Confirme que la consola esté configurada en la misma región de AWS que el clúster que desee revisar.
2. Revise la columna Política de actualización. Si el valor es Soporte estándar, el clúster no entrará en el soporte extendido. Si el valor es Soporte extendido, el clúster entrará en el soporte extendido.

Visualización de la política de actualización de clústeres (consola de AWS)

1. Compruebe que la CLI de AWS esté instalada y que haya iniciado sesión. Aprenda a actualizar e instalar la CLI de AWS.
2. Determine el nombre del clúster de EKS. Configure la CLI en la misma región de AWS que su clúster de EKS.
3. Ejecuta el siguiente comando:

```
aws eks describe-cluster \
--name <cluster-name> \
--query "cluster.upgradePolicy.supportType"
```

4. Si el valor es `STANDARD`, el clúster no entrará en el soporte extendido. Si el valor es `EXTENDED`, el clúster entrará en el soporte extendido.

Habilitación del soporte extendido de EKS para agregar flexibilidad a fin de planificar las actualizaciones de las versiones de Kubernetes

En este tema se describe cómo configurar la política de actualización de un clúster de EKS para habilitar el soporte extendido. La política de actualización de un clúster de EKS determina lo que ocurre cuando un clúster llega al final del periodo de soporte estándar. Si una política de actualización de clústeres tiene habilitado el soporte extendido, entrará en el periodo de soporte extendido al final del periodo de soporte estándar. El clúster no se actualizará automáticamente al final del periodo de soporte estándar.

Los clústeres que se encuentran realmente en el periodo de soporte extendido generan costos más altos. Si un clúster solo tiene la política de actualización establecida para habilitar el soporte extendido y, por lo demás, se encuentra en el periodo de soporte estándar, generará costos estándar.

Si crea un clúster en la consola de AWS, tendrá la política de actualización configurada para deshabilitar el soporte extendido. Si crea un clúster de otra forma, tendrá la política de actualización configurada para deshabilitar el soporte extendido. Por ejemplo, los clústeres creados con la API de AWS tienen habilitado el soporte extendido.

Para obtener más información sobre las políticas de actualización, consulte Cluster upgrade policy.

> **Important**
>
> Si desea que su clúster mantenga la versión de Kubernetes actual para aprovechar el periodo de soporte extendido, debe habilitar la política de actualización de soporte extendido antes de que finalice el periodo de soporte estándar.
> Si no habilita el soporte extendido, el clúster se actualizará automáticamente.

Habilitación del soporte extendido de EKS (consola de AWS)

1. Vaya a su clúster de EKS en la consola de AWS. Seleccione la pestaña Información general en la página Información del clúster.
2. En la sección Configuración de la versión de Kubernetes, seleccione Administrar.
3. Seleccione Soporte extendido y, a continuación, Guardar cambios.

Habilitación del soporte extendido de EKS (CLI de AWS)

1. Compruebe que la CLI de AWS esté instalada y que haya iniciado sesión. Aprenda a actualizar e instalar la CLI de AWS.
2. Determine el nombre del clúster de EKS.
3. Ejecuta el siguiente comando:

```
aws eks update-cluster-config \
--name <cluster-name> \
--upgrade-policy supportType=EXTENDED
```

Deshabilitación del soporte extendido de EKS para evitar el aumento de los costos del clúster

En este tema se describe cómo configurar la política de actualización de un clúster de EKS para deshabilitar el soporte extendido. La política de actualización de un clúster de EKS determina lo que ocurre cuando un clúster llega al final del periodo de soporte estándar. Si el soporte extendido de una política de actualización de clústeres está deshabilitado, se actualizará automáticamente a la siguiente versión de Kubernetes.

Para obtener más información sobre las políticas de actualización, consulte Cluster upgrade policy.

> **Important**
>
> No se puede deshabilitar el soporte extendido luego de que se aplique al clúster. Solo se puede deshabilitar el soporte extendido para los clústeres con soporte estándar.
> AWS recomienda actualizar el clúster a una versión que se encuentre en el periodo de soporte estándar.

Deshabilitación del soporte extendido de EKS (consola de AWS)

1. Vaya a su clúster de EKS en la consola de AWS. Seleccione la pestaña Información general en la página Información del clúster.
2. En la sección Configuración de la versión de Kubernetes, seleccione Administrar.
3. Seleccione Soporte estándar y, a continuación, Guardar cambios.

Deshabilitación del soporte extendido de EKS (CLI de AWS)

1. Compruebe que la CLI de AWS esté instalada y que haya iniciado sesión. Aprenda a actualizar e instalar la CLI de AWS.
2. Determine el nombre del clúster de EKS.
3. Ejecuta el siguiente comando:

```
aws eks update-cluster-config \
--name <cluster-name> \
--upgrade-policy supportType=STANDARD
```

Visualización de las versiones de la plataforma Amazon EKS para cada versión de Kubernetes

Las versiones de la plataforma de Amazon EKS representan las funciones del plano de control del clúster de Amazon EKS, como qué marcas de servidor de API de Kubernetes están habilitadas, así como la versión de parche de Kubernetes actual. Cada versión secundaria de Kubernetes tiene una o varias versiones de la plataforma de Amazon EKS asociadas. Las versiones de la plataforma para las diferentes versiones secundarias de Kubernetes son independientes. Puede recuperar la versión de la plataforma actual de su clúster mediante AWS CLI o Consola de administración de AWS. Si tiene un clúster local en AWS Outposts, consulte the section called “Versiones de la plataforma de EKS” en lugar de este tema.

Cuando se encuentra disponible una versión secundaria de Kubernetes nueva en Amazon EKS, por ejemplo, la 1.33, la versión de la plataforma inicial de Amazon EKS para esa versión secundaria de Kubernetes comienza con `eks.1`. Sin embargo, Amazon EKS lanza nuevas versiones de la plataforma de forma periódica para habilitar nuevos ajustes de plano de control de Kubernetes y proporcionar revisiones de seguridad.

Cuando nuevas versiones de la plataforma de Amazon EKS se encuentran disponibles para una versión secundaria:

- El número de versión de la plataforma de Amazon EKS se incrementa (`eks.<n+1>`).
- Amazon EKS actualiza automáticamente todos los clústeres existentes a la última versión de la plataforma de Amazon EKS para su versión secundaria de Kubernetes correspondiente. Las actualizaciones automáticas de las versiones de la plataforma de Amazon EKS existentes se implementan de forma incremental. El proceso de implementación puede tardar algún tiempo. Si necesita las características de la última versión de la plataforma de Amazon EKS de forma inmediata, debe crear un nuevo clúster de Amazon EKS.

  Si el clúster está más de dos versiones de plataforma por detrás de la versión de la plataforma actual, es posible que Amazon EKS no haya podido actualizar automáticamente el clúster. Para obtener más información sobre lo que puede causar esto, consulte the section called “La versión de la plataforma Amazon EKS está más de dos versiones por detrás de la versión de plataforma actual”.
- Amazon EKS puede publicar una nueva AMI de nodo con la versión de parche correspondiente. Sin embargo, todas las versiones de parche son compatibles entre el plano de control de EKS y las AMI de nodo para una versión secundaria de Kubernetes determinada.

Las nuevas versiones de la plataforma de Amazon EKS no introducen cambios bruscos ni provocan interrupciones de servicio.

Los nuevos clústeres siempre se crean con la última versión de la plataforma de Amazon EKS disponible (`eks.<n>`) para la versión de Kubernetes especificada. Si actualiza su clúster a una nueva versión secundaria de Kubernetes, el clúster recibe la versión de la plataforma de Amazon EKS actual para la versión secundaria de Kubernetes a la que se actualizó.

Las versiones de la plataforma de Amazon EKS actuales y recientes se describen en las siguientes tablas.

> **Note**
>
> AWS deshabilitó recientemente algunas versiones de la plataforma publicadas en junio de 2024. Las versiones de la plataforma tenían problemas de estabilidad. No es necesario ninguna acción.

Para recibir notificaciones de todos los cambios en el archivo de origen de esta página de documentación específica, puede suscribirse a la siguiente URL con un lector de RSS:

```
https://github.com/awsdocs/amazon-eks-user-guide/commits/mainline/latest/ug/clusters/platform-versions.adoc.atom
```

Versión de Kubernetes `1.34`

Los siguientes controladores de admisión están habilitados para todas las versiones de plataforma de `1.34`: `NodeRestriction`, `ExtendedResourceToleration`, `NamespaceLifecycle`, `LimitRanger`, `ServiceAccount`, `TaintNodesByCondition`, `PodSecurity`, `Priority`, `DefaultTolerationSeconds`, `DefaultStorageClass`, `StorageObjectInUseProtection`, `PersistentVolumeClaimResize`, `RuntimeClass`, `CertificateApproval`, `CertificateSigning`, `CertificateSubjectRestriction`, `DefaultIngressClass`, `MutatingAdmissionWebhook`, `ValidatingAdmissionWebhook`, `ResourceQuota`.

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
`1.34.1`	`eks.9`	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	18 de noviembre de 2025
`1.34.1`	`eks.8`	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	17 de noviembre de 2025
`1.34.1`	`eks.7`	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	6 de noviembre de 2025
`1.34.1`	`eks.6`	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	30 de octubre de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.34.1	eks.5	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	15 de octubre de 2025
1.34.1	eks.4	Versión inicial de Kubernetes 1.34 para Amazon EKS. Para obtener más información, consulte the section called "Kubernetes 1.34".	2 de octubre de 2025

Versión de Kubernetes **1.33**

Los siguientes controladores de admisión están habilitados para todas las versiones de plataforma de 1.33: NodeRestriction, ExtendedResourceToleration, NamespaceLifecycle, LimitRanger, ServiceAccount, TaintNodesByCondition, PodSecurity, Priority, DefaultTolerationSeconds, DefaultStorageClass, StorageObjectInUseProtection, PersistentVolumeClaimResize, RuntimeClass, CertificateApproval, CertificateSigning, CertificateSubjectRestriction, DefaultIngressClass, MutatingAdmissionWebhook, ValidatingAdmissionWebhook, ResourceQuota.

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.33.5	eks.23	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	18 de noviembre de 2025
1.33.5	eks.22	Nueva versión de la plataforma con	17 de noviembre de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
		mejoras y correcciones de seguridad.	
1.33.5	eks.21	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	6 de noviembre de 2025
1.33.5	eks.20	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	30 de octubre de 2025
1.33.5	eks.19	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	28 de octubre de 2025
1.33.5	eks.18	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	15 de octubre de 2025
1.33.5	eks.17	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	15 de octubre de 2025
1.33.5	eks.16	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	2 de octubre de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.33.4	eks.15	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	29 de septiembre de 2025
1.33.4	eks.14	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	29 de septiembre de 2025
1.33.4	eks.13	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	16 de septiembre de 2025
1.33.4	eks.12	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	5 de septiembre de 2025
1.33.3	eks.11	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	25 de agosto de 2025
1.33.3	eks.10	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	17 de agosto de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.33.2	eks.9	Nueva versión de la plataforma con mejoras y correcciones de seguridad. Se descartaron internamente 1.33 eks.7 y 1.33 eks.8, y nunca se lanzaron.	30 de julio de 2025
1.33.1	eks.6	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	26 de junio de 2025
1.33.1	eks.5	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	11 de junio de 2025
1.33.1	eks.4	Versión inicial de Kubernetes 1.33 para Amazon EKS. Para obtener más información, consulte the section called “Kubernetes 1.33”.	28 de mayo de 2025

Versión de Kubernetes **1.32**

Los siguientes controladores de admisión están habilitados para todas las versiones de plataforma de 1.32: NodeRestriction, ExtendedResourceToleration, NamespaceLifecycle, LimitRanger, ServiceAccount, TaintNodesByCondition, PodSecurity, Priority, DefaultTolerationSeconds, DefaultStorageClass, StorageObjectInUseProtection, PersistentVolumeClaimResize, RuntimeClass, CertificateApproval,

CertificateSigning, CertificateSubjectRestriction, DefaultIngressClass, MutatingAdmissionWebhook, ValidatingAdmissionWebhook, ResourceQuota.

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.32.9	eks.30	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	18 de noviembre de 2025
1.32.9	eks.29	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	17 de noviembre de 2025
1.32.9	eks.28	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	6 de noviembre de 2025
1.32.9	eks.27	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	30 de octubre de 2025
1.32.9	eks.26	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	28 de octubre de 2025
1.32.9	eks.25	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	15 de octubre de 2025
1.32.9	eks.24	Nueva versión de la plataforma con	15 de octubre de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
		mejoras y correcciones de seguridad.	
1.32.9	eks.23	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	2 de octubre de 2025
1.32.8	eks.22	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	29 de septiembre de 2025
1.32.8	eks.21	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	29 de septiembre de 2025
1.32.8	eks.20	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	16 de septiembre de 2025
1.32.8	eks.19	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	5 de septiembre de 2025
1.32.7	eks.18	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	25 de agosto de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.32.7	eks.17	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	17 de agosto de 2025
1.32.6	eks.16	Nueva versión de la plataforma con mejoras y correcciones de seguridad. Se descartaron internamente 1.32 eks.14 y 1.32 eks.15, y nunca se lanzaron.	30 de julio de 2025
1.32.5	eks.13	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	26 de junio de 2025
1.32.5	eks.12	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	11 de junio de 2025
1.32.5	eks.11	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	30 de mayo de 2025
1.32.3	eks.10	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	16 de mayo de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.32.3	eks.9	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	29 de abril de 2025
1.32.3	eks.8	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	18 de abril de 2025
1.32.3	eks.7	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	18 de abril de 2025
1.32.3	eks.6	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	2 de abril de 2025
1.32.2	eks.5	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	17 de marzo de 2025
1.32.2	eks.4	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	4 de marzo de 2025
1.32.1	eks.3	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	24 de febrero de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.32.0	eks.2	Versión inicial de Kubernetes 1.32 para Amazon EKS. Para obtener más información, consulte the section called "Kubernetes 1.32".	Enero de 2025

Versión de Kubernetes 1.31

Los siguientes controladores de admisión están habilitados para todas las versiones de plataforma 1.31: NodeRestriction, ExtendedResourceToleration, NamespaceLifecycle, LimitRanger, ServiceAccount, TaintNodesByCondition, PodSecurity, Priority, DefaultTolerationSeconds, DefaultStorageClass, StorageObjectInUseProtection, PersistentVolumeClaimResize, RuntimeClass, CertificateApproval, CertificateSigning, CertificateSubjectRestriction, DefaultIngressClass, MutatingAdmissionWebhook, ValidatingAdmissionWebhook, ResourceQuota, ObjectCount.

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.31.13	eks.46	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	18 de noviembre de 2025
1.31.13	eks.45	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	17 de noviembre de 2025
1.31.13	eks.44	Nueva versión de la plataforma con	6 de noviembre de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
		mejoras y correcciones de seguridad.	
1.31.13	eks.43	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	30 de octubre de 2025
1.31.13	eks.42	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	28 de octubre de 2025
1.31.13	eks.41	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	15 de octubre de 2025
1.31.13	eks.40	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	15 de octubre de 2025
1.31.13	eks.39	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	2 de octubre de 2025
1.31.12	eks.38	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	29 de septiembre de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.31.12	eks.37	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	29 de septiembre de 2025
1.31.12	eks.36	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	16 de septiembre de 2025
1.31.12	eks.35	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	5 de septiembre de 2025
1.31.11	eks.34	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	25 de agosto de 2025
1.31.11	eks.33	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	17 de agosto de 2025
1.31.10	eks.32	Nueva versión de la plataforma con mejoras y correcciones de seguridad. Se descartaron internamente 1.31 eks.30 y 1.31 eks.31, y nunca se lanzaron.	30 de julio de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.31.9	eks.29	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	26 de junio de 2025
1.31.9	eks.28	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	11 de junio de 2025
1.31.9	eks.27	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	30 de mayo de 2025
1.31.7	eks.26	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	16 de mayo de 2025
1.31.7	eks.25	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	29 de abril de 2025
1.31.7	eks.24	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	18 de abril de 2025
1.31.7	eks.23	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	18 de abril de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.31.7	eks.22	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	2 de abril de 2025
1.31.6	eks.21	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	17 de marzo de 2025
1.31.6	eks.20	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	4 de marzo de 2025
1.31.5	eks.19	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	24 de febrero de 2025
1.31.5	eks.18	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	24 de febrero de 2025
1.31.4	eks.17	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	17 de enero de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.31.2	eks.12	Nueva versión de la plataforma compatible con los Nodos híbridos de Amazon EKS y mejoras en la observabilidad del plano de control. Consulte the section called “Nodos híbridos”, así como Amazon EKS mejora la observabilidad del rendimiento, respectivamente.	15 de noviembre de 2024
1.31.1	eks.6	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	21 de octubre de 2024
1.31.0	eks.4	Versión inicial de Kubernetes 1.31 para Amazon EKS. Para obtener más información, consulte the section called “Kubernetes 1.31”.	26 de septiembre de 2024

Versión de Kubernetes **1.30**

Los siguientes controladores de admisión están habilitados para todas las versiones de plataforma de 1.30: NodeRestriction, ExtendedResourceToleration, NamespaceLifecycle, LimitRanger, ServiceAccount, TaintNodesByCondition, PodSecurity, Priority,

DefaultTolerationSeconds, DefaultStorageClass, StorageObjectInUseProtection, PersistentVolumeClaimResize, RuntimeClass, CertificateApproval, CertificateSigning, CertificateSubjectRestriction, DefaultIngressClass, MutatingAdmissionWebhook, ValidatingAdmissionWebhook, ResourceQuota.

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.30.14	eks.54	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	18 de noviembre de 2025
1.30.14	eks.53	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	17 de noviembre de 2025
1.30.14	eks.52	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	6 de noviembre de 2025
1.30.14	eks.51	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	30 de octubre de 2025
1.30.14	eks.50	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	28 de octubre de 2025
1.30.14	eks.49	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	15 de octubre de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.30.14	eks.48	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	15 de octubre de 2025
1.30.14	eks.47	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	2 de octubre de 2025
1.30.14	eks.46	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	29 de septiembre de 2025
1.30.14	eks.45	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	29 de septiembre de 2025
1.30.14	eks.44	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	16 de septiembre de 2025
1.30.14	eks.43	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	5 de septiembre de 2025
1.30.14	eks.42	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	25 de agosto de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.30.14	eks.41	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	17 de agosto de 2025
1.30.14	eks.40	Nueva versión de la plataforma con mejoras y correcciones de seguridad. Se descartaron internamente 1.30 eks.38 y 1.30 eks.39, y nunca se lanzaron.	30 de julio de 2025
1.30.13	eks.37	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	26 de junio de 2025
1.30.13	eks.36	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	11 de junio de 2025
1.30.13	eks.35	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	30 de mayo de 2025
1.30.11	eks.34	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	16 de mayo de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.30.11	eks.33	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	29 de abril de 2025
1.30.11	eks.32	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	18 de abril de 2025
1.30.11	eks.31	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	18 de abril de 2025
1.30.11	eks.30	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	2 de abril de 2025
1.30.10	eks.29	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	17 de marzo de 2025
1.30.10	eks.28	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	4 de marzo de 2025
1.30.9	eks.27	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	24 de febrero de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.30.8	eks.25	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	17 de enero de 2025
1.30.8	eks.24	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	3 de enero de 2025
1.30.7	eks.23	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	13 de diciembre de 2024
1.30.6	eks.22	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	13 de diciembre de 2024
1.30.6	eks.21	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	13 de diciembre de 2024

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.30.6	eks.20	Nueva versión de la plataforma compatible con los Nodos híbridos de Amazon EKS y mejoras en la observabilidad del plano de control. Consulte the section called “Nodos híbridos”, así como Amazon EKS mejora la observabilidad del rendimiento, respectivamente.	15 de noviembre de 2024
1.30.5	eks.12	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	21 de octubre de 2024
1.30.4	eks.8	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	3 de septiembre de 2024
1.30.3	eks.7	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	28 de agosto de 2024
1.30.3	eks.6	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	9 de agosto de 2024

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.30.2	eks.5	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	2 de julio de 2024
1.30.0	eks.2	Versión inicial de Kubernetes 1.30 para Amazon EKS. Para obtener más información, consulte the section called "Kubernetes 1.30".	23 de mayo de 2024

Versión de Kubernetes **1.29**

Los siguientes controladores de admisión están habilitados para todas las versiones de plataforma de 1.29: NodeRestriction, ExtendedResourceToleration, NamespaceLifecycle, LimitRanger, ServiceAccount, TaintNodesByCondition, PodSecurity, Priority, DefaultTolerationSeconds, DefaultStorageClass, StorageObjectInUseProtection, PersistentVolumeClaimResize, RuntimeClass, CertificateApproval, CertificateSigning, CertificateSubjectRestriction, DefaultIngressClass, MutatingAdmissionWebhook, ValidatingAdmissionWebhook, ResourceQuota.

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.29.15	eks.57	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	18 de noviembre de 2025
1.29.15	eks.56	Nueva versión de la plataforma con	17 de noviembre de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
		mejoras y correcciones de seguridad.	
1.29.15	eks.55	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	6 de noviembre de 2025
1.29.15	eks.54	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	30 de octubre de 2025
1.29.15	eks.53	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	28 de octubre de 2025
1.29.15	eks.52	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	15 de octubre de 2025
1.29.15	eks.51	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	15 de octubre de 2025
1.29.15	eks.50	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	2 de octubre de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.29.15	eks.49	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	29 de septiembre de 2025
1.29.15	eks.48	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	29 de septiembre de 2025
1.29.15	eks.47	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	16 de septiembre de 2025
1.29.15	eks.46	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	5 de septiembre de 2025
1.29.15	eks.45	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	25 de agosto de 2025
1.29.15	eks.44	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	17 de agosto de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.29.15	eks.43	Nueva versión de la plataforma con mejoras y correcciones de seguridad. Se descartaron internamente 1.29 eks.41 y 1.29 eks.42, y nunca se lanzaron.	30 de julio de 2025
1.29.15	eks.40	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	26 de junio de 2025
1.29.15	eks.39	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	11 de junio de 2025
1.29.15	eks.38	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	30 de mayo de 2025
1.29.15	eks.37	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	16 de mayo de 2025
1.29.15	eks.36	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	29 de abril de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.29.15	eks.35	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	18 de abril de 2025
1.29.15	eks.34	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	18 de abril de 2025
1.29.15	eks.33	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	2 de abril de 2025
1.29.14	eks.32	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	17 de marzo de 2025
1.29.14	eks.31	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	4 de marzo de 2025
1.29.13	eks.30	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	24 de febrero de 2025
1.29.13	eks.29	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	24 de febrero de 2025

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.29.12	eks.28	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	20 de enero de 2025
1.29.12	eks.27	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	3 de enero de 2025
1.29.11	eks.26	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	13 de diciembre de 2024
1.29.10	eks.25	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	13 de diciembre de 2024
1.29.10	eks.24	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	13 de diciembre de 2024

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.29.10	eks.23	Nueva versión de la plataforma compatible con los Nodos híbridos de Amazon EKS y mejoras en la observabilidad del plano de control. Consulte the section called “Nodos híbridos”, así como Amazon EKS mejora la observabilidad del rendimiento, respectivamente.	15 de noviembre de 2024
1.29.9	eks.17	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	21 de octubre de 2024
1.29.8	eks.13	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	3 de septiembre de 2024
1.29.7	eks.12	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	28 de agosto de 2024
1.29.7	eks.11	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	9 de agosto de 2024

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.29.6	eks.10	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	2 de julio de 2024
1.29.4	eks.7	Nueva versión de la plataforma con escalado automático de CoreDNS, mejoras y correcciones de seguridad. Para obtener más información sobre el escalado automático de CoreDNS, consulte the section called “Cómo escalar para atender un tráfico elevado”.	16 de mayo de 2024
1.29.3	eks.6	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	18 de abril de 2024
1.29.1	eks.5	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	29 de marzo de 2024
1.29.1	eks.4	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	20 de marzo de 2024

Versión de Kubernetes	Versión de la plataforma de EKS	Notas de la versión	Fecha de lanzamiento de la nueva versión
1.29.1	eks.3	Nueva versión de la plataforma con mejoras y correcciones de seguridad.	12 de marzo de 2024
1.29.0	eks.1	Versión inicial de Kubernetes 1.29 para Amazon EKS. Para obtener más información, consulte the section called “Kubernetes 1.29”.	23 de enero de 2024

Obtenga la versión actual de la plataforma

1. Abra la consola de Amazon EKS.
2. En el panel de navegación, seleccione Clusters (Clústeres).
3. En la lista de clústeres, elija el nombre del clúster para comprobar la versión de la plataforma.
4. Elija la pestaña Información general.
5. La versión de la plataforma está disponible en la sección de detalles.
6. Determine el nombre del clúster del que desea comprobar la versión de la plataforma.
7. Use el siguiente comando:

```
aws eks describe-cluster --name my-cluster --query cluster.platformVersion
```

Un ejemplo de salida sería el siguiente.

```
"eks.10"
```

Cambio de la versión de la plataforma

No puede cambiar la versión de la plataforma de un clúster de EKS. Cuando hay nuevas versiones de la plataforma de Amazon EKS disponibles para una versión de Kubernetes, EKS actualiza automáticamente todos los clústeres existentes a la última versión de la plataforma de Amazon EKS para su versión de Kubernetes correspondiente. Las actualizaciones automáticas de las versiones de la plataforma de Amazon EKS existentes se implementan de forma incremental. No puede usar la CLI ni la consola de AWS para cambiar la versión de la plataforma.

Si actualiza la versión de Kubernetes, el clúster pasará a la versión de la plataforma más reciente para la versión de Kubernetes.

Solución de problemas con los clústeres y nodos de Amazon EKS

En este capítulo se tratan algunos errores habituales que pueden aparecer al utilizar Amazon EKS y cómo solucionarlos. Si necesita solucionar problemas en áreas específicas de Amazon EKS, consulte los temas aparte the section called “Solución de problemas”, the section called “Solución de problemas del conector de EKS” y Troubleshooting for ADOT using EKS Add-Ons.

Para obtener más información sobre la solución de problemas, consulte el contenido del Centro de conocimientos sobre Amazon Elastic Kubernetes Service en AWS re:Post.

Capacidad insuficiente

Si aparece el siguiente error al intentar crear un clúster de Amazon EKS, significa que una de las zonas de disponibilidad que ha especificado no tiene capacidad suficiente para admitir un clúster.

```
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
```

Intente crear de nuevo el clúster con subredes de la VPC de clúster alojadas en las zonas de disponibilidad indicadas en el mensaje de error.

Hay zonas de disponibilidad en las que un clúster no puede residir. Compare las zonas de disponibilidad en las que se encuentran las subredes con la lista de zonas de disponibilidad en Requisitos y consideraciones de subredes.

Los nodos no pueden unirse al clúster

Hay algunos motivos que suelen impedir que los nodos se unan al clúster:

- Si los nodos son nodos administrados, Amazon EKS añade entradas al `ConfigMap` de `aws-auth` al crear el grupo de nodos. Si la entrada se eliminó o modificó, tendrá que volver a agregarla. Para obtener más información, ingrese `eksctl create iamidentitymapping --help` en su terminal. Puede comprobar las entradas actuales de `aws-auth` `ConfigMap` reemplazando *`my-cluster`* en el siguiente comando por el nombre de su clúster y luego ejecutando el comando modificado: `eksctl get iamidentitymapping --cluster` *`my-cluster`* . El ARN del

rol que especifique no puede incluir una ruta que no sea /. Por ejemplo, si el nombre del rol es `development/apps/my-role`, deberá cambiarlo a `my-role` cuando especifique el ARN del rol. Asegúrese de especificar el ARN del rol de IAM correspondiente al nodo (y no el ARN del perfil de instancia).

Si los nodos son autoadministrados y no ha creado entradas de acceso para el ARN del rol de IAM del nodo, ejecute los mismos comandos indicados para los nodos administrados. Si ha creado una entrada de acceso para el ARN del rol de IAM de su nodo, es posible que no esté configurado correctamente en la entrada de acceso. Asegúrese de especificar el ARN del rol de IAM del nodo (y no el ARN del perfil de instancia) como el ARN de la entidad principal en su entrada o entrada de acceso del `ConfigMap` de `aws-auth`. Para obtener más información acerca de las entradas de acceso, consulte the section called “Entradas de los registros”.

- El valor de Nombre de clúster en la plantilla de AWS CloudFormation del nodo no coincide exactamente con el nombre del clúster al que desea que se unan los nodos. Si se especifica un valor incorrecto en este campo, la configuración del archivo `/var/lib/kubelet/kubeconfig` del nodo no será correcta y los nodos no podrán unirse al clúster.
- El nodo no está etiquetado como propiedad del clúster. Los nodos deben tener la siguiente etiqueta aplicada a ellos, donde *`my-cluster`* se reemplaza por el nombre del clúster.

Clave	Valor
`kubernetes.io/cluster/` *`my-cluster`*	`owned`

- Es posible que los nodos no puedan tener acceso al clúster mediante una dirección IP pública. Asegúrese de que los nodos implementados en subredes públicas tengan asignada una dirección IP pública. Si no es así, puede asociar una dirección IP elástica a un nodo después de que se lance. Para obtener más información, consulte Asociación de una dirección IP elástica a una instancia de ejecución o una interfaz de red. Si la subred pública no está configurada para asignar automáticamente direcciones IP públicas a instancias implementadas en ella, recomendamos habilitar esa configuración. Para obtener más información, consulte Modificación del atributo de direcciones IPv4 públicas de su subred. Si se implementa el nodo en una subred privada, la subred debe tener una ruta a una gateway NAT que tenga asignada una dirección IP pública.
- El punto de conexión de AWS STS de la región de AWS en la que está implementando los nodos no está habilitado para su cuenta. Para habilitar la región, consulte Activación y desactivación de AWS STS en una región de AWS.

- El nodo no tiene una entrada DNS privada, lo que provoca que el registro de `kubelet` contenga un error de `node "" not found`. Asegúrese de que la VPC donde se crea el nodo tenga valores establecidos para `domain-name` y `domain-name-servers` como `Options` en un `DHCP options set`. Los valores predeterminados son `domain-name:<region>.compute.internal` y `domain-name-servers:AmazonProvidedDNS`. Para más información, consulte Conjuntos de opciones de DHCP en la Guía del usuario de Amazon VPC.
- Si los nodos del grupo de nodos administrados no se conectan al clúster dentro de 15 minutos, se emitirá un problema de estado con el nombre “NodeCreationFailure” y el estado de la consola se establecerá en `Create failed`. En el caso de las AMI de Windows con tiempos de inicio lentos, este problema se puede resolver con el inicio rápido.

Para identificar y solucionar problemas de causas comunes que impiden que los nodos de trabajo se unan a un clúster, puede usar el manual de procedimientos `AWSSupport-TroubleshootEKSWorkerNode`. Para obtener más información, consulte `AWSSupport-TroubleshootEKSWorkerNode` en la Referencia del manual de procedimientos de AWS Systems Manager Automation.

Acceso denegado o no autorizado (**`kubectl`**)

Si aparece uno de los siguientes errores al ejecutar los comandos de `kubectl`, eso significa que su `kubectl` no está configurado correctamente para Amazon EKS o las credenciales para la entidad principal de IAM (rol u usuario) que utiliza no están asignadas a un nombre de usuario de Kubernetes con suficientes permisos para los objetos de Kubernetes en su clúster de Amazon EKS.

- `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"`

Esto podría deberse a una de las siguientes razones:

- El clúster se creó con credenciales para una entidad principal de IAM y `kubectl` utiliza credenciales para otra entidad principal de IAM. Para resolver este problema, actualice el archivo `kube config` para usar las credenciales que crearon el clúster. Para obtener más información, consulte the section called “Acceso al clúster con kubectl”.

- Si el clúster cumple los requisitos mínimos de la plataforma que se indican en la sección de requisitos previos Concesión de acceso a Kubernetes a usuarios de IAM con entradas de acceso de EKS, no existe una entrada de acceso con la entidad principal de IAM. Si existe, no tiene definidos los nombres de grupo de Kubernetes necesarios o no tiene asociada la política de acceso adecuada. Para obtener más información, consulte the section called “Entradas de los registros”.
- Si el clúster no cumple los requisitos mínimos de la plataforma que se indican en Concesión de acceso a Kubernetes a usuarios de IAM con entradas de acceso de EKS, no existe una entrada con la entidad principal de IAM en `aws-auth ConfigMap`. Si existe, no está asignada a nombres de grupos de Kubernetes vinculados a un `Role` o `ClusterRole` de Kubernetes con los permisos necesarios. Para obtener más información sobre los objetos de autorización basada en roles (RBAC) de Kubernetes, consulte Using RBAC authorization en la documentación de Kubernetes. Puede comprobar las entradas actuales de `aws-auth ConfigMap` reemplazando *`my-cluster`* en el siguiente comando por el nombre de su clúster y luego ejecutando el comando modificado: `eksctl get iamidentitymapping --cluster` *`my-cluster`* . Si no hay una entrada para el ARN de la entidad principal de IAM en el `ConfigMap`, ingrese `eksctl create iamidentitymapping --help` en el terminal para aprender a crear una.

Si instala y configura la AWS CLI, puede configurar las credenciales de IAM que utiliza. Para obtener más información, consulte Configurar la AWS CLI, en la Guía del usuario de la interfaz de línea de comandos de AWS. También puede configurar `kubectl` para utilizar un rol de IAM si asume un rol de IAM para acceder a los objetos de Kubernetes en su clúster. Para obtener más información, consulte the section called “Acceso al clúster con kubectl”.

`hostname doesn’t match`

La versión de Python de su sistema debe ser `2.7.9` o posterior. En caso contrario, se producirán errores `hostname doesn’t match` en las llamadas de la AWS CLI a Amazon EKS. Para obtener más información, consulte ¿Qué significan los errores “el nombre del host no coincide”? en Preguntas frecuentes sobre las solicitudes de Python.

`getsockopt: no route to host`

Docker se ejecuta en el rango de CIDR `172.17.0.0/16` en los clústeres de Amazon EKS. Le recomendamos que las subredes de la VPC de su clúster no se superpongan. De lo contrario, recibirá el siguiente error:

```
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`

Si aparece el error `Instances failed to join the Kubernetes cluster` en la Consola de administración de AWS, asegúrese de que el acceso privado al punto de conexión del clúster esté habilitado o de que haya configurado correctamente los bloques de CIDR para el acceso público al punto de conexión. Para obtener más información, consulte the section called “Acceso al punto de conexión del clúster”.

Códigos de error del grupo de nodos administrado

Si el grupo de nodos administrado encuentra un problema de estado del hardware, Amazon EKS devuelve un código de error para ayudarlo a diagnosticar el problema. Estas comprobaciones de estado no detectan problemas de software porque se basan en comprobaciones de estado de Amazon EC2. En la siguiente lista se describen los códigos de error.

AccessDenied

Amazon EKS, o uno o varios de sus nodos administrados, no pueden autenticarse ni autorizarse con su servidor de la API del clúster de Kubernetes. Para obtener más información sobre la resolución de una causa común, consulte the section called “Resolución de una causa común de errores `AccessDenied` para grupos de nodos administrados.”. Las AMI privadas de Windows también pueden provocar este código de error junto con el mensaje de error `Not authorized for images`. Para obtener más información, consulte the section called “`Not authorized for images`”.

AmiIdNotFound

No hemos podido encontrar el ID de la AMI asociado a su plantilla de lanzamiento. Asegúrese de que la AMI existe y de que se comparte con su cuenta.

AutoScalingGroupNotFound

No se pudo encontrar el grupo de escalado automático asociado al grupo de nodos administrados. Es posible que pueda volver a crear un grupo de Auto Scaling con la misma configuración para recuperarlo.

ClusterUnreachable

Amazon EKS, o uno o varios de los nodos administrados, no pueden comunicarse con el servidor de la API del clúster de Kubernetes. Esto puede suceder si hay interrupciones en la red o si los servidores de la API están agotando el tiempo de espera de las solicitudes de procesamiento.

Ec2SecurityGroupNotFound

No se pudo encontrar el grupo de seguridad del clúster para el clúster. Debe volver a crear el clúster.

Ec2SecurityGroupDeletionFailure

No hemos podido eliminar el grupo de seguridad de acceso remoto para el grupo de nodos administrados. Elimine cualquier dependencia del grupo de seguridad.

Ec2LaunchTemplateNotFound

No hemos podido encontrar la plantilla de lanzamiento de Amazon EC2 para el grupo de nodos administrados. Debe volver a crear el grupo de nodos para recuperarlo.

Ec2LaunchTemplateVersionMismatch

La versión de la plantilla de lanzamiento de Amazon EC2 para el grupo de nodos administrados no coincide con la versión creada por Amazon EKS. Es posible que pueda volver a la versión que creó Amazon EKS para recuperarla.

IamInstanceProfileNotFound

No hemos podido encontrar el perfil de instancia de IAM para su grupo de nodos administrados. Es posible que pueda volver a crear un perfil de instancia con la misma configuración para recuperarlo.

IamNodeRoleNotFound

No hemos podido encontrar el rol de IAM para su grupo de nodos administrados. Es posible que pueda volver a crear un rol de IAM con la misma configuración para recuperarlo.

AsgInstanceLaunchFailures

Su grupo de escalado automático está experimentando errores al intentar lanzar instancias.

NodeCreationFailure

Las instancias lanzadas no pueden registrarse en el clúster de Amazon EKS. Las causas comunes de este error son permisos del rol de IAM del nodo insuficientes o falta de acceso a Internet saliente para los nodos. Los nodos deben cumplir cualquiera de los siguientes requisitos:

- Capaz de acceder a internet mediante una dirección IP pública. El grupo de seguridad asociado a la subred en la que se encuentra el nodo debe permitir la comunicación. Para obtener más información, consulte the section called “Requisitos y consideraciones de la subred” y the section called “Requisitos del grupo de seguridad”.
- Los nodos y la VPC deben cumplir los requisitos que aparecen en la sección Implementación de clústeres privados con acceso limitado a Internet.

InstanceLimitExceeded

Su cuenta de AWS no puede lanzar más instancias del tipo de instancia especificado. Es posible que pueda solicitar un aumento del límite de instancias de Amazon EC2 para recuperarlas.

InsufficientFreeAddresses

Una o varias de las subredes asociadas al grupo de nodos administrados no tienen suficientes direcciones IP disponibles para nuevos nodos.

InternalFailure

Estos errores suelen ser causados por un problema del lado del servidor de Amazon EKS.

Resolución de una causa común de errores **`AccessDenied`** para grupos de nodos administrados.

La causa más común de los errores de `AccessDenied` al realizar operaciones en grupos de nodos administrados es la falta de `eks:node-manager`, `ClusterRole` o `ClusterRoleBinding`. Amazon EKS configura estos recursos en el clúster como parte de la incorporación con grupos de nodos administrados, y estos son necesarios para administrar los grupos de nodos.

`ClusterRole` puede cambiar con el tiempo, pero debe parecerse al siguiente ejemplo:

```
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
```

`ClusterRoleBinding` puede cambiar con el tiempo, pero debe parecerse al siguiente ejemplo:

```
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
```

Verifique que el `ClusterRole` de `eks:node-manager` existe.

```
kubectl describe clusterrole eks:node-manager
```

Si está presente, compare la salida con el ejemplo de `ClusterRole` anterior.

Verifique que el `ClusterRoleBinding` de `eks:node-manager` existe.

```
kubectl describe clusterrolebinding eks:node-manager
```

Si está presente, compare la salida con el ejemplo de `ClusterRoleBinding` anterior.

Si ha identificado un `ClusterRole` o `ClusterRoleBinding` ausente o defectuoso como la causa de un error de `AcessDenied` al solicitar operaciones de grupo de nodos administrados, puede restaurarlos. Guarde los siguientes contenidos en un archivo con el nombre *`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
```

Aplique el archivo.

```
kubectl apply -f eks-node-manager-role.yaml
```

Vuelva a intentar la operación del grupo de nodos para ver si se resolvió el problema.

`Not authorized for images`

Una posible causa de un mensaje de error `Not authorized for images` es el uso de una AMI privada de Windows de Amazon EKS para lanzar grupos de nodos administrados. Tras lanzar nuevas AMI de Windows, AWS convierte en privadas las AMI de más de cuatro meses de antigüedad, por lo que ya no se puede acceder a ellas. Si el grupo de nodos que administra utiliza una AMI de Windows privada, considere la posibilidad de actualizar su grupo de nodos administrados de Windows. Si bien no podemos garantizar que podamos proporcionar acceso a las AMI que se han convertido en privadas, puede solicitar el acceso enviando un ticket a AWS Support. Para obtener más información, consulte Parches en la Guía del usuario de Amazon EC2.

El nodo está en estado **`NotReady`**

Si el nodo entra en estado `NotReady`, es probable que esto indique que el nodo está en mal estado y no está disponible para programar nuevos pods. Esto puede ocurrir por varios motivos, como que el nodo carezca de recursos suficientes para la CPU, la memoria o el espacio disponible en el disco.

En el caso de las AMI de Windows optimizadas para Amazon EKS, no hay reservas para los recursos de computación especificados de forma predeterminada en la configuración de `kubelet`. Para evitar problemas de recursos, puede reservar recursos de cómputos para los procesos del sistema proporcionando valores de configuración de kube-reserved o system-reserved al `kubelet`. Para ello, utilice el parámetro de línea de comandos `-KubeletExtraArgs` del script de arranque. Para obtener más información, consulte Reserve Compute Resources for System Daemons en la

documentación de Kubernetes y los the section called “Parámetros de configuración del script de arranque” de esta guía del usuario.

Recopilador del registro de EKS

Para solucionar problemas con los nodos de Amazon EKS, hay un script prediseñado disponible en los nodos ubicado en `/etc/eks/log-collector-script/eks-log-collector.sh`. Puede utilizar el script a fin de recopilar registros de diagnóstico para casos de soporte y solución de problemas general.

Utilice el siguiente comando para ejecutar el script en su nodo:

```
sudo bash /etc/eks/log-collector-script/eks-log-collector.sh
```

> **Note**
>
> Si el script no está presente en esa ubicación. Puede descargar y ejecutar manualmente el script con el siguiente comando:
>
> ```
> curl -O https://amazon-eks.s3.amazonaws.com/support/log-collector-script/linux/
> eks-log-collector.sh
> sudo bash eks-log-collector.sh
> ```

El script recopila la siguiente información de diagnóstico:

```
$ 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
```

La información de diagnóstico se recopila y se almacena en:

```
/var/log/eks_i-EXAMPLE_2025-03-25_0000-UTC_0.7.8.tar.gz
```

Para recuperar el paquete de registros de los nodos de Bottlerocket, consulte el registro de Bottlerocket para obtener más información.

La red de tiempo de ejecución del contenedor no está lista

Puede recibir un error `Container runtime network not ready` y errores de autorización similares a los siguientes:

```
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
```

Esto puede deberse a una de las siguientes razones:

1. O bien no tiene un `ConfigMap` de `aws-auth` en su clúster, o este no incluye entradas para el rol de IAM con el que configuró sus nodos.

   Para resolver este problema, consulte las entradas existentes en su `ConfigMap` reemplazando *`my-cluster`* en el siguiente comando con el nombre de su clúster y, luego, ejecute el comando modificado: `eksctl get iamidentitymapping --cluster` *`my-cluster`* . Si recibe un mensaje de error del comando, puede que esto se deba a que el clúster no tiene un `aws-auth` `ConfigMap`. El siguiente comando añade una entrada al `ConfigMap`. Si `ConfigMap` no existe, el comando también lo crea. Sustituya *`111122223333`* por el ID de la cuenta de AWS correspondiente al rol de IAM y *`myAmazonEKSNodeRole`* por el nombre del rol del nodo.

   ```
   eksctl create iamidentitymapping --cluster my-cluster \
       --arn arn:aws:iam::111122223333:role/myAmazonEKSNodeRole --group
    system:bootstrappers,system:nodes \
       --username system:node:{{EC2PrivateDNSName}}
   ```

El ARN del rol que especifique no puede incluir una ruta que no sea `/`. Por ejemplo, si el nombre de su rol es `development/apps/my-role`, tendrá que cambiarlo por `my-role` cuando especifique el ARN del rol. Asegúrese de especificar el ARN del rol de IAM correspondiente al nodo (y no el ARN del perfil de instancia).

2. Los nodos autoadministrados se encuentran en un clúster con una versión de la plataforma que es la versión mínima indicada en los requisitos previos del tema Concesión de acceso a Kubernetes a usuarios de IAM con entradas de acceso de EKS, pero no aparece ninguna entrada en el `aws-auth ConfigMap` (consulte el punto anterior) correspondiente al rol de IAM del nodo o no existe una entrada de acceso para ese rol. Para resolver el problema, consulte las entradas de acceso existentes reemplazando *`my-cluster`* en el siguiente comando por el nombre de su clúster y luego ejecutando el comando modificado: `aws eks list-access-entries --cluster-name` *`my-cluster`* . El siguiente comando agrega una entrada de acceso para el rol de IAM del nodo. Sustituya *`111122223333`* por el ID de la cuenta de AWS correspondiente al rol de IAM y *`myAmazonEKSNodeRole`* por el nombre del rol del nodo. Si tiene un nodo de Windows, reemplace *`EC2_Linux`* por `EC2_Windows`. Asegúrese de especificar el ARN del rol de IAM correspondiente al nodo (y no el ARN del perfil de instancia).

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/myAmazonEKSNodeRole --type EC2_LINUX
```

Tiempo de espera de protocolo de enlace TLS

Cuando un nodo no puede establecer una conexión con el punto de conexión del servidor de la API público, podría ver un error similar al siguiente.

```
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\""
```

El proceso `kubelet` reaparecerá continuamente y probará el punto de conexión del servidor de API. El error también puede producirse temporalmente durante cualquier procedimiento que realice una actualización sucesiva del clúster en el plano de control, como un cambio de configuración o una actualización de versión.

Para resolver el problema, verifique la tabla de enrutamiento y los grupos de seguridad para asegurarse de que el tráfico de los nodos puede llegar al punto de conexión público.

InvalidClientTokenId

Si está utilizando roles de IAM en las cuentas de servicio de un pod o DaemonSet implementado en un clúster de la región de AWS China y no ha establecido la variable de entorno `AWS_DEFAULT_REGION` en la especificación, el pod o el DaemonSet pueden recibir el siguiente error:

```
An error occurred (InvalidClientTokenId) when calling the GetCallerIdentity operation:
 The security token included in the request is invalid
```

Para resolver el problema, debe agregar la variable de entorno `AWS_DEFAULT_REGION` a la especificación del pod o DaemonSet, tal y como se muestra en el siguiente ejemplo de especificación de un pod.

```
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"
```

Los grupos de nodos deben coincidir con la versión de Kubernetes antes de actualizar el plano de control

Antes de actualizar un plano de control a una nueva versión de Kubernetes, la versión secundaria de los nodos administrados y de Fargate en el clúster debe ser la misma que la de la versión actual del plano de control. La API `update-cluster-version` de Amazon EKS rechazará las solicitudes hasta que usted actualice todos los nodos administrados de Amazon EKS a la versión del clúster actual. Amazon EKS proporciona las API para actualizar los nodos administrados. Para obtener información sobre la actualización de la versión de Kubernetes del grupo de nodos administrados, consulte the section called “Actualización”. Para actualizar la versión de un nodo de Fargate, elimine

el pod que representa el nodo y vuelva a implementar el pod después de actualizar el plano de control. Para obtener más información, consulte the section called “Actualización de una versión de Kubernetes”.

Al lanzar muchos nodos, hay errores de `Too Many Requests`

Si lanza muchos nodos al mismo tiempo, es posible que vea un mensaje de error en Datos de usuario de Amazon EC2registros de ejecución que dice `Too Many Requests`. Esto puede ocurrir porque el plano de control está sobrecargado con llamadas `describeCluster`. La sobrecarga da como resultado una limitación, es decir, que los nodos no ejecutan el script de arranque ni se unen por completo al clúster.

Asegúrese de que los argumentos `--apiserver-endpoint`, `--b64-cluster-ca` y `--dns-cluster-ip` se estén pasando al script de arranque del nodo. Al incluir estos argumentos, no es necesario que el script de arranque haga una llamada `describeCluster`, lo que ayuda a evitar que el plano de control se sobrecargue. Para obtener más información, consulte the section called “Proporcione datos de usuario a fin de pasar argumentos al archivo `bootstrap.sh` incluido con una AMI optimizada de Linux o Bottlerocket para Amazon EKS.”.

Respuesta de error no autorizada HTTP 401 en solicitudes del servidor API de Kubernetes

Verá estos errores si el token de la cuenta de servicio de un pod caducó en un clúster.

El servidor de la API de Kubernetes del clúster de Amazon EKS rechaza las solicitudes con tokens de más de 90 días. En versiones anteriores de Kubernetes, los tokens no tenían caducidad. Esto significa que los clientes que confían en estos tokens deben actualizarlos en una hora. Para evitar que el servidor de API de Kubernetes rechace su solicitud debido a un token no válido, la versión del SDK de cliente de Kubernetes utilizada por la carga de trabajo debe ser la misma o posterior a las siguientes versiones:

- Versión de Go `0.15.7` y posteriores
- Versión de Python `12.0.0` y posteriores
- Versión de Java `9.0.0` y posterior
- Versión de JavaScript `0.10.3` y posterior
- Rama de Ruby `master`

- Versión de Haskell `0.3.0.0`
- Versión `7.0.5` y posteriores de C#

Puede identificar todos los pods existentes de su clúster que utilizan tokens obsoletos. Para obtener más información, consulte the section called “Tokens de cuenta de servicio”.

La versión de la plataforma Amazon EKS está más de dos versiones por detrás de la versión de plataforma actual

Esto puede ocurrir cuando Amazon EKS no puede actualizar automáticamente la versión de la plataforma del clúster. Aunque hay muchas causas para esto, se presentan algunas de las causas más comunes. Si alguno de estos problemas se aplica a su clúster, es posible que siga funcionando. Amazon EKS no actualizará la versión de la plataforma.

Problema

El rol de IAM de clúster se eliminó: este rol se especificó cuando se creó el clúster. Puede ver qué rol se ha especificado con el siguiente comando. Reemplace *my-cluster* por el nombre de su clúster.

```
aws eks describe-cluster --name my-cluster --query cluster.roleArn --output text | cut
 -d / -f 2
```

Un ejemplo de salida sería el siguiente.

```
eksClusterRole
```

Solución

Cree un nuevo rol de IAM de clúster con el mismo nombre.

Problema

Se eliminó una subred especificada durante la creación del clúster: las subredes que se usarían con el clúster se especificaron durante la creación del clúster. Puede ver qué subredes se especificaron con el siguiente comando. Reemplace *my-cluster* por el nombre de su clúster.

```
aws eks describe-cluster --name my-cluster --query cluster.resourcesVpcConfig.subnetIds
```

Un ejemplo de salida sería el siguiente.

```
[
"subnet-EXAMPLE1",
"subnet-EXAMPLE2"
]
```

Solución

Confirme si los ID de subred existen en su cuenta.

```
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"
```

Un ejemplo de salida sería el siguiente.

```
[
"subnet-EXAMPLE3",
"subnet-EXAMPLE4"
]
```

Si los ID de subredes devueltos en la salida no coinciden con los ID de subredes que se especificaron cuando se creó el clúster y desea que Amazon EKS actualice el clúster, debe cambiar las subredes utilizadas por el clúster. Esto se debe a que, si especificó más de dos subredes cuando creó el clúster, Amazon EKS seleccionará aleatoriamente las subredes que especificó para crear nuevas interfaces de red elástica en ellas. Estas interfaces de red permiten que el plano de control se comunique con los nodos. Amazon EKS no actualizará el clúster si la subred que selecciona no existe. No tiene control sobre las subredes que especificó en la creación del clúster en las que Amazon EKS elige crear una nueva interfaz de red.

Cuando inicia una actualización de la versión de Kubernetes del clúster, la actualización puede fallar por el mismo motivo.

Problema

Se eliminó un grupo de seguridad especificado durante la creación del clúster: si especificó grupos de seguridad durante la creación del clúster, puede ver sus ID con el siguiente comando. Reemplace *`my-cluster`* por el nombre de su clúster.

```
aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.securityGroupIds
```

Un ejemplo de salida sería el siguiente.

```
[
    "sg-EXAMPLE1"
]
```

Si se devuelve `[]`, no se especificó ningún grupo de seguridad cuando se creó el clúster y el problema no es que falte un grupo de seguridad. Si se devuelven grupos de seguridad, confirme que los grupos de seguridad existen en su cuenta.

Solución

Confirme si estos grupos de seguridad existen en su cuenta.

```
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"
```

Un ejemplo de salida sería el siguiente.

```
[
"sg-EXAMPLE2"
]
```

Si los ID de grupos de seguridad devueltos en la salida no coinciden con los ID de grupos de seguridad que se especificaron cuando se creó el clúster y desea que Amazon EKS actualice el clúster, debe cambiar los grupos de seguridad utilizados por el clúster. Amazon EKS no actualizará un clúster si los ID de grupos de seguridad especificados en la creación del clúster no existen.

Cuando inicia una actualización de la versión de Kubernetes del clúster, la actualización puede fallar por el mismo motivo.

- No tiene al menos seis (aunque recomendamos 16) direcciones IP disponibles en cada una de las subredes que especificó al crear el clúster. Si no tiene suficientes direcciones IP disponibles en la subred, debe liberar direcciones IP en la subred o bien cambiar las subredes utilizadas por el clúster para que emplee subredes con suficientes direcciones IP disponibles.

- Ha habilitado el cifrado de secretos cuando creó el clúster y se ha eliminado la clave de AWS KMS especificada. Si desea que Amazon EKS actualice el clúster, deberá crear un clúster nuevo

Preguntas frecuentes sobre el estado de los clústeres y los códigos de error con rutas de resolución

Amazon EKS detecta problemas en los clústeres de EKS y en la infraestructura del clúster, y los almacena en el objeto health del recurso de clúster de EKS. Puede detectar, solucionar y abordar los problemas del clúster más rápido con la ayuda de la información sobre el estado del clúster. Esto le permite crear entornos de aplicación más seguros y actualizados. Además, es posible que no pueda actualizar a versiones más recientes de Kubernetes o que Amazon EKS no pueda instalar actualizaciones de seguridad en un clúster degradado debido a problemas con la infraestructura necesaria o la configuración del clúster. Amazon EKS puede tardar 3 horas en detectar problemas o detectar que se resolvió un problema.

El estado de un clúster de Amazon EKS es una responsabilidad compartida entre Amazon EKS y sus usuarios. Usted es responsable de la infraestructura previa de los roles de IAM y las subredes de Amazon VPC, así como de cualquier otra infraestructura necesaria que deba proporcionarse con antelación. Amazon EKS detecta cambios en la configuración de esta infraestructura y del clúster.

Para acceder al estado del clúster en la consola de Amazon EKS, busque una tabla llamada Problemas de estado en la pestaña Problemas de estado del clúster del panel de observabilidad, al que se accede desde la página de detalles del clúster en Amazon EKS. Estos datos también estarán disponibles al llamar a la acción `DescribeCluster` en la API de EKS, por ejemplo, desde AWS Command Line Interface.

¿Por qué debo utilizar esta característica?

> Obtendrá una mayor visibilidad del estado de su clúster de Amazon EKS, diagnosticará y solucionará rápidamente cualquier problema, sin necesidad de perder tiempo depurando o abriendo casos de asistencia de AWS. Por ejemplo: si accidentalmente eliminó una subred del clúster de Amazon EKS, Amazon EKS no podrá crear interfaces entre cuentas ni comandos de la AWS CLI de Kubernetes, tales como ejecuciones de `kubectl` o registros de `kubectl`. Estos fallarán con el error: `Error from server: error dialing backend: remote error: tls: internal error.` Ahora verá un problema de estado de Amazon EKS que dice: `subnet-da60e280 was deleted: could not create network interface`.

¿Cómo se relaciona o funciona esta característica con otros servicios de AWS?

Los roles de IAM y las subredes de Amazon VPC son dos ejemplos de infraestructura previa donde el estado del clúster detecta problemas. Esta característica devolverá información detallada si esos recursos no están configurados correctamente.

¿Se cobran cargos por un clúster con problemas de estado?

Sí, todos los clústeres de Amazon EKS se facturan al precio estándar de Amazon EKS. La característica estado del clúster está disponible sin costo adicional.

¿Funciona esta característica con los clústeres de Amazon EKS en AWS Outposts?

Sí, se detectan problemas con los clústeres de EKS en la Nube de AWS, incluidos los clústeres extendidos en AWS Outposts y los clústeres locales en AWS Outposts. El estado del clúster no detecta problemas con Amazon EKS Anywhere o Amazon EKS Distro (EKS-D).

¿Puedo recibir una notificación cuando se detecten nuevos problemas?

Sí. AWS envía un correo electrónico y una notificación al panel de Health personal cuando se detectan nuevos problemas de estado en el clúster.

¿La consola me avisa en caso de problemas de estado?

Sí, cualquier clúster con problemas de estado incluirá un cartel en la parte superior de la consola.

Las primeras dos columnas son las que se necesitan para los valores de respuesta de la API. El tercer campo del objeto Problemas en el estado del clúster es resourceIds, cuya devolución depende del tipo de problema.

Código	Mensaje	ResourceIds	¿Clúster recuperable?
SUBNET_NOT_FOUND	No logramos encontrar una o más subredes actualmente asociadas a su clúster. Llame a la API update-cluster-config de Amazon EKS para actualizar las subredes.	ID de subred	Sí

Código	Mensaje	ResourceIds	¿Clúster recuperable?
SECURITY_GROUP_NOT_FOUND	No logramos encontrar uno o más grupos de seguridad actualmente asociados a su clúster. Llame a la API update-cluster-config de Amazon EKS para actualizar los grupos de seguridad	ID de grupos de seguridad	Sí
IP_NO_DISPONIBLE	Una o varias de las subredes asociadas a su clúster no tienen suficientes direcciones IP disponibles para que Amazon EKS realice operaciones de administración del clúster. Libere direcciones en las subredes o asocie diferentes subredes a su clúster mediante la API update-cluster-config de Amazon EKS.	ID de subred	Sí
VPC_NOT_FOUND	No logramos encontrar la VPC asociada a su clúster. Debe eliminar y volver a crear su clúster.	ID de la VPC	No

Código	Mensaje	ResourceIds	¿Clúster recuperable?
ASSUME_ROLE_ACCESS_DENIED	Su clúster no utiliza el rol vinculado al servicio de Amazon EKS. No logramos asumir el rol asociado a su clúster para realizar las operaciones de administración necesarias de Amazon EKS. Compruebe si el rol existe y tiene la política de confianza requerida.	El rol de IAM del clúster	Sí
PERMISSION_ACCESS_DENIED	Su clúster no utiliza el rol vinculado al servicio de Amazon EKS. El rol asociado a su clúster no otorga permisos suficientes para que Amazon EKS lleve a cabo las operaciones de administración requeridas. Compruebe las políticas asociadas al rol del clúster y si se aplica alguna política de denegación independiente.	El rol de IAM del clúster	Sí

Código	Mensaje	ResourceIds	¿Clúster recuperable?
ASSUME_ROLE_ACCESS_DENIED_USING_SLR	No logramos asumir el rol vinculado al servicio de administración del clúster de Amazon EKS. Compruebe si el rol existe y tiene la política de confianza requerida.	El rol vinculado al servicio de Amazon EKS	Sí
PERMISSION_ACCESS_DENIED_USING_SLR	El rol vinculado al servicio de administración del clúster de Amazon EKS no concede permisos suficientes para que Amazon EKS lleve a cabo las operaciones de administración requeridas. Compruebe las políticas asociadas al rol del clúster y si se aplica alguna política de denegación independiente.	El rol vinculado al servicio de Amazon EKS	Sí
OPT_IN_REQUIRED	Su cuenta no tiene una suscripción al servicio Amazon EC2. Actualice las suscripciones de su cuenta en la página de configuración de su cuenta.	N/A	Sí

Código	Mensaje	ResourceIds	¿Clúster recuperable?
STS_REGIONAL_ENDPOINT_DISABLED	El punto de conexión regional de STS está deshabilitado. Habilite el punto de conexión para que Amazon EKS lleve a cabo las operaciones de administración del clúster necesarias.	N/A	Sí
KMS_KEY_DISABLED	La clave de AWS KMS asociada a su clúster está deshabilitada. Vuelva a habilitar la clave para recuperar su clúster.	El Arn de la clave de KMS	Sí
KMS_KEY_NOT_FOUND	No logramos encontrar la clave de AWS KMS asociada a su clúster. Debe eliminar y volver a crear el clúster.	El ARN de la clave de KMS	No
KMS_GRANT_REVOKED	Se revocan las concesiones de la clave de AWS KMS asociada al clúster. Debe eliminar y volver a crear el clúster.	El Arn de la clave de KMS	No

Ampliación de las capacidades de Amazon EKS con proyectos de código abierto

Estos proyectos de código abierto extienden la funcionalidad de los clústeres de Kubernetes que se ejecutan en AWS o por fuera, incluidos los clústeres administrados por Amazon EKS.

Soporte del software implementado en EKS

Al revisar los documentos de Amazon EKS, encontrará referencias a diversas herramientas y software de código abierto en todos nuestros procedimientos y ejemplos. Estas herramientas incluyen el Servidor de métricas de Kubernetes y Cert Manager.

Tenga en cuenta que cualquier software de terceros o de código abierto que decida implementar queda fuera del ámbito de sus acuerdos de AWS Support. Una de las ventajas de usar Kubernetes es la comunidad de código abierto activa. Recomendamos trabajar directamente con las comunidades de código abierto y los responsables del proyecto pertinentes para establecer los canales de soporte adecuados para dichos componentes. Para obtener más información, consulte los proyectos graduados y en incubación asociadas a la Cloud Native Computing Foundation (CNCF).

El ecosistema de Kubernetes incluye numerosos proyectos y componentes que cuentan con diferentes niveles de soporte comunitario, tiempos de respuesta y casos de uso previstos. Al implementar estas tecnologías junto con EKS, asegúrese de entender la matriz de soporte de cada componente.

AWS mantiene los componentes de código abierto que integramos en el plano de control de EKS. Esto incluye nuestra exhaustiva canalización de seguridad que cubre la verificación de creación, el escaneo de vulnerabilidades, las pruebas de validación y la administración de parches para todos los binarios y las imágenes de contenedor que distribuimos. Por ejemplo, AWS es responsable del Servidor de la API de Kubernetes. El Servidor de la API de Kubernetes está cubierto por el Acuerdo de nivel de servicio de Amazon EKS. Puede usar el plan de soporte de Amazon Web Services para resolver problemas con el servidor de API de Kubernetes u obtener orientación general.

Debe revisar detenidamente el soporte que se ofrece para los complementos de Amazon EKS. Los complementos de AWS son el único tipo de complemento de Amazon EKS totalmente compatible con AWS. AWS Los complementos de Marketplace son compatibles principalmente con los socios de

AWS. Los complementos comunitarios reciben soporte básico durante todo el ciclo de vida de AWS. Para más información, consulte Soporte de complementos.

Cada complemento de EKS, independientemente del tipo, recibe el soporte básico de EKS durante todo el ciclo de vida, lo que incluye los complementos de Marketplace. El soporte básico durante todo el ciclo de vida incluye la instalación y la desinstalación del complemento. Para obtener más información sobre los tipos de complementos de Amazon EKS disponibles y los niveles de soporte asociados, consulte Alcance del soporte de los complementos de Amazon EKS. Para ver los complementos totalmente compatibles de AWS, consulte Complementos de Amazon Web Services.

- Para obtener más información sobre nuestras prácticas de seguridad y los límites del soporte, consulte Seguridad en Amazon EKS.
- Para obtener más información sobre los complementos de la comunidad y de AWS Marketplace disponibles a través de complementos de Amazon EKS, consulte Soporte de complementos de EKS.

Herramientas de administración

Herramientas de administración relacionadas para clústeres de Kubernetes y Amazon EKS.

eksctl

`eksctl` es una herramienta de CLI sencilla para crear clústeres de Amazon EKS.

- URL del proyecto
- Documentación del proyecto
- Blog de código abierto de AWS: eksctl: Amazon EKS cluster with one command

Controladores de AWS para Kubernetes

Con los controladores de AWS para Kubernetes, puede crear y administrar recursos de AWS directamente desde su clúster de Kubernetes.

Disponible en Capacidades de EKS.

- URL del proyecto
- Blog de código abierto de AWS: AWS Service Operator for Kubernetes now available

kro (Kube Resource Orchestrator)

kro le permite crear API de Kubernetes personalizadas que componen varios recursos en abstracciones de nivel superior. Los equipos de plataformas pueden definir patrones reutilizables con barreras de protección, mientras que los equipos de aplicaciones utilizan API sencillas y de alto nivel para aprovisionar y administrar recursos.

Disponible en Capacidades de EKS.

- URL del proyecto
- Documentación del proyecto

Argo CD

Argo CD es una herramienta declarativa de entrega continua de GitOps para Kubernetes. Supervisa continuamente los repositorios de Git y sincroniza automáticamente los cambios en los clústeres.

Disponible en Capacidades de EKS.

- URL del proyecto
- Documentación del proyecto

Flux CD

Flux es una herramienta que puede utilizar para administrar la configuración del clúster con Git. Utiliza un operador en el clúster para desencadenar las implementaciones dentro de Kubernetes. Para obtener más información sobre los operadores, consulte OperatorHub.io en GitHub.

- URL del proyecto
- Documentación del proyecto

CDK para Kubernetes

Con el CDK para Kubernetes (cdk8s), puede definir aplicaciones y componentes de Kubernetes mediante lenguajes de programación conocidos. Las aplicaciones cdk8s se sintetizan en manifiestos estándar de Kubernetes, que se pueden aplicar a cualquier clúster de Kubernetes.

- URL del proyecto

- Documentación del proyecto
- Blog de contenedores de AWS: Introducing cdk8s+: Intent-driven APIs for Kubernetes objects

Red

Proyectos de redes relacionados para clústeres de Kubernetes y Amazon EKS.

Complemento CNI de Amazon VPC para Kubernetes

Amazon EKS admite redes de VPC nativas gracias al complemento CNI de Amazon VPC para Kubernetes. El complemento asigna una dirección IP de su VPC a cada pod.

- URL del proyecto
- Documentación del proyecto

AWSControlador del balanceador de carga de para Kubernetes

El controlador del balanceador de carga de AWS ayuda a administrar los Elastic Load Balancer de AWS para un clúster de Kubernetes. Satisface los recursos de entrada de Kubernetes mediante el aprovisionamiento de los balanceadores de carga de aplicaciones de AWS. Satisface los recursos de Kubernetes Service mediante el aprovisionamiento de los equilibradores de carga de red de AWS.

- URL del proyecto
- Documentación del proyecto

ExternalDNS

ExternalDNS sincroniza las entradas y los servicios de Kubernetes expuestos con proveedores de DNS como Amazon Route 53 y detección de servicios de AWS.

- URL del proyecto
- Documentación del proyecto

Machine learning

Proyectos de machine learning relacionados para clústeres de Kubernetes y Amazon EKS.

Kubeflow

Un conjunto de herramientas de machine learning para Kubernetes.

- URL del proyecto
- Documentación del proyecto
- Blog de código abierto de AWS: Kubeflow on Amazon EKS

Auto Scaling

Proyectos de escalado automático relacionados para clústeres de Kubernetes y Amazon EKS.

Escalador automático del clúster

Es una herramienta que ajusta automáticamente el tamaño del clúster de Kubernetes en función de la presión de memoria y CPU.

- URL del proyecto
- Documentación del proyecto
- Taller de Amazon EKS: Cluster Autoscaler

Karpenter

Karpenter es un Escalador automático de nodos de Karpenter creado para ofrecer flexibilidad, rendimiento y simplicidad.

- URL del proyecto
- Documentación del proyecto
- Taller de Amazon EKS: Karpenter

Escalador

Es un escalador automático horizontal optimizado de trabajo o lote para Kubernetes.

- URL del proyecto
- Documentación del proyecto

Supervisión

Proyectos de monitoreo relacionados para clústeres de Kubernetes y Amazon EKS.

Prometheus

Prometheus es un conjunto de herramientas de alerta y monitoreo de sistemas de código abierto.

- URL del proyecto
- Documentación del proyecto
- Taller de Amazon EKS: https://eksworkshop.com/intermediate/240_monitoring/

Integración continua/implementación continua

Proyectos de CI/CD imperativos relacionados para clústeres de Amazon EKS y Kubernetes.

Jenkins X

Solución de integración continua/implementación continua para aplicaciones en la nube modernas en clústeres de Kubernetes y Amazon EKS.

- URL del proyecto
- Documentación del proyecto

Información sobre nuevas características y hoja de ruta de Amazon EKS

Para obtener más información sobre las nuevas características de Amazon EKS, desplácese hasta la fuente en la página Novedades de AWS. También puede revisar el plan de desarrollo en GitHub, que le permite conocer las próximas características y prioridades para poder planificar cómo desea utilizar Amazon EKS en el futuro. Puede proporcionarnos comentarios directos sobre las prioridades del plan de desarrollo.

Historial de documentos

En la siguiente tabla se describen algunas de las principales actualizaciones y nuevas características de la Guía del usuario de Amazon EKS. Para obtener notificaciones cuando esta tabla reciba una nueva entrada, puede suscribirse a la siguiente URL con un lector de RSS:

```
https://docs.aws.amazon.com/eks/latest/userguide/doc-history.rss
```

Cambio	Descripción	Fecha
Capacidades de Amazon EKS	Amazon EKS ahora proporcio na capacidades de clúster completamente administradas con capacidades de EKS, con soporte para la implement ación continua declarativa con soporte para Argo CD, administración de recursos de AWS con soporte para Controladores de AWS para Kubernetes y composición y orquestación de recursos de Kubernetes con soporte para Kube Resource Orchestrator.	30 de noviembre de 2025
Nueva política administrada por AWS	Amazon EKS ha lanzado una nueva política administrada `AmazonEKSMCPReadOnlyAccess` para activar herramientas de solo lectura en el servidor MCP de Amazon EKS para la observabilidad y la solución de problemas. Para obtener más información, consulte Actualizaciones de Amazon	21 de noviembre de 2025

	EKS a políticas administradas de AWS.	
Observabilidad de la red	Amazon EKS ahora proporciona una mejor observabilidad de la red de contenedores con supervisión del rendimiento y visibilidad del tráfico de cargas de trabajo.	19 de noviembre de 2025
AWS Actualizaciones de políticas administradas por	Se agregó el permiso `ec2:CopyVolumes` a `AmazonEBSCSIDriverPolicy` para permitir que el controlador CSI de EBS copie los volúmenes de EBS directamente.	17 de noviembre de 2025
AWS Actualizaciones de políticas administradas por	Se agregaron los permisos `ec2:DescribeRouteTables` y `ec2:DescribeNetworkAcls` a `AmazonEKSServiceRolePolicy` . De este modo, Amazon EKS puede detectar problemas de configuración con las tablas de enrutamiento de VPC y las ACL de red para nodos híbridos como parte de la información de los clústeres.	22 de octubre de 2025
Actualización de roles vinculados al servicio	Se agregó el permiso `ssmmessages:OpenDataChannel` a `AmazonEKSConnectorServiceRolePolicy` .	15 de octubre de 2025

Versión de Kubernetes 1.34	Se ha agregado compatibilidad con la versión 1.34 de Kubernetes para nuevas actualizaciones de versión y clústeres.	2 de octubre de 2025
Reparación automática de nodos configurables	Amazon EKS ahora proporciona un control más detallado sobre el comportamiento de reparación automática de nodos.	22 de septiembre de 2025
Actualización de la información del clúster	No puede actualizar manualmente la información del clúster.	27 de agosto de 2025
AWS Actualizaciones de políticas administradas por	Se agregó el permiso a AmazonEKSServiceRolePolicy . Este rol puede adjuntar la nueva política de acceso AmazonEKSEventPolicy . Permisos restringidos para ec2:DeleteLaunchTemplate y ec2:TerminateInstances .	26 de agosto de 2025

Prevención de la sustitución confusa entre servicios	Se ha agregado un tema con una política de confianza de ejemplo que puede aplicar para la prevención del suplente confuso entre servicios. Amazon EKS acepta las condiciones `aws:SourceArn` y `aws:SourceAccount` en la política de confianza de un rol de clúster de EKS.	19 de agosto de 2025
Actualización de la versión de la plataforma de Amazon EKS	Se trata de una nueva versión de la plataforma con mejoras y correcciones de seguridad. Esto incluye las nuevas versiones de parches de Kubernetes `1.33.2`, `1.32.6`, `1.31.10` y `1.30.14`.	30 de julio de 2025

Característica de NIC múltiple con CNI de la VPC para pods con varios hosts	Amazon EKS añade pods con varios hosts a la CNI de la VPC. Ahora puede configura r una carga de trabajo y las direcciones IP asignadas por la CNI de la VPC desde cada NIC de la instancia de EC2 a cada pod. La aplicació n puede realizar conexione s simultáneas para utilizar el ancho de banda de cada NIC. Todas las interfaces de red están configuradas en la misma subred y en los mismos grupos de seguridad que el nodo. Anteriormente, era necesario utilizar CNI Multus para ejecutar varias CNI adicionales y crear pods con varios hosts. La documenta ción y los pasos para hacerlo se han trasladado a eks/latest/ userguide/pod-multus.html.	15 de julio de 2025
Actualización del contenido de solución de problemas de la CNI de la VPC	Se ha ampliado la página de solución de problemas de la política de red de Kubernetes en la CNI de la VPC. Se agregaron los CRD y los permisos RBAC, y 12 problemas conocidos.	30 de junio de 2025

Actualización de la versión de la plataforma de Amazon EKS	Se trata de una nueva versión de la plataforma con mejoras y correcciones de seguridad . No hay nuevas versiones de revisión de Kubernetes en esta versión de la plataforma.	26 de junio de 2025
AWS Actualizaciones de políticas administradas por	Se agregó el permiso `ssmmessages:OpenDataChannel` a `AmazonEKSLocalOutpostServiceRolePolicy` .	26 de junio de 2025
AWS Actualizaciones de políticas administradas por	Se agregaron permisos en `AmazonEKSServiceRolePolicy` y `AmazonEKSComputePolicy` para permitir que el modo automático de Amazon EKS ejecute instancias mediante las reservas de capacidad bajo demanda de EC2 en su cuenta. Además, se agregaron permisos en `AmazonEKSComputePolicy` para que Amazon EKS cree el rol vinculado al servicio EC2 Spot en su nombre.	20 de junio de 2025
Actualizaciones de políticas administradas	Se agregó la política `AmazonEKSDashboardConsoleReadOnly` .	19 de junio de 2025

Actualización del modo automático de Amazon EKS a `NodeClass`	La plantilla `NodeClass` para los nodos del modo automático agregó la configuración para subredes de pods independientes. Esto agrega las claves opcionales `podSubnetSelectorTerms` y `podSecurityGroupSelectorTerms` para configurar las subredes y los grupos de seguridad de los pods.	13 de junio de 2025
Roles entre cuentas y secundarios de destino con EKS Pod Identities	Amazon EKS agrega roles de IAM de destino a EKS Pod Identities para el encadenamiento de roles automatizado. Puede utilizarlo para asumir automáticamente un rol en otra cuenta y EKS Pod Identity rota las credenciales temporales. Cada asociación de Pod Identity debe tener un rol de IAM en la misma cuenta para asumirlo primero y, después, utilizará ese rol para asumir el rol de destino.	11 de junio de 2025
Actualización de la versión de la plataforma de Amazon EKS	Se trata de una nueva versión de la plataforma con mejoras y correcciones de seguridad. No hay nuevas versiones de revisión de Kubernetes en esta versión de la plataforma.	11 de junio de 2025

Ampliación de las regiones de AWS Amazon EKS	Amazon EKS ya está disponible en la región de AWS Asia-Pacífico (Taipéi) (`ap-east-2`).	6 de junio de 2025
`IPv6` Control de acceso `IPv6` para puntos de conexión públicos de doble pila para nuevos clústeres	Amazon EKS agrega bloques de CIDR `IPv6` para controlar el acceso al punto de conexión del clúster público para nuevos clústeres `IPv6`. Anteriormente, solo podía agregar bloques de CIDR `IPv4` para permitir el tráfico al punto de conexión del clúster público, incluso en el caso de los puntos de conexión del clúster de doble pila.	5 de junio de 2025
Actualización de la versión de la plataforma de Amazon EKS	Se trata de una nueva versión de la plataforma con mejoras y correcciones de seguridad. Esto incluye las nuevas versiones de parches de Kubernetes `1.32.5`, `1.31.9` y `1.30.13`.	30 de mayo de 2025
Nueva información del clúster para nodos híbridos de EKS	Amazon EKS agrega nueva información del clúster que comprueba la configuración de los nodos híbridos. Estas comprobaciones de información le avisarán sobre problemas relacionados con los nodos y pods en las instalaciones y con la configuración de red remota del clúster.	29 de mayo de 2025

Versión de Kubernetes `1.33`	Se ha agregado compatibilidad con la versión `1.33` de Kubernetes para nuevas actualizaciones de versión y clústeres.	29 de mayo de 2025
Compatibilidad del complemento para el controlador de CSI de Amazon FSx	A partir de ahora, puede utilizar la Consola de administración de AWS, la AWS CLI y la API para administrar el controlador de CSI de Amazon FSx.	23 de mayo de 2025
Edición de raspadores de Prometheus en la consola	Ahora puede editar raspadores de Amazon Managed Service para Prometheus en la consola de Amazon EKS.	22 de mayo de 2025
Actualizaciones de políticas administradas	Se agregó la política `AmazonEKSDashboardServiceRolePolicy` .	21 de mayo de 2025
Actualización de la versión de la plataforma de Amazon EKS	Se trata de una nueva versión de la plataforma con mejoras y correcciones de seguridad . No hay nuevas versiones de revisión de Kubernetes en estas versiones de plataforma.	16 de mayo de 2025
Nuevas páginas sobre el rendimiento de Amazon FSx para Lustre	Se agregaron nuevos temas con detalles sobre la optimización del rendimiento de Amazon FSx para Lustre.	2 de mayo de 2025

Actualización del modo automático de Amazon EKS a `NodeClass`	La plantilla de `NodeClass` para nodos en modo automático agregó la configuración para proxys de red directa. Esto agrega la clave opcional `advancedNetworking` para configurar el proxy HTTPS.	30 de abril de 2025
Bottlerocket para nodos híbridos	Bottlerocket ahora se encuentra disponible para los Nodos híbridos de EKS.	29 de abril de 2025
Actualización de la versión de la plataforma de Amazon EKS	Se trata de una nueva versión de la plataforma con mejoras y correcciones de seguridad. No hay nuevas versiones de revisión de Kubernetes en estas versiones de plataforma.	29 de abril de 2025
Nuevas páginas de conceptos sobre redes híbridas	Se agregaron páginas para los conceptos de los nodos híbridos de EKS. Abarcan en detalle las redes en las instalaciones y en la nube con diagramas.	18 de abril de 2025

AWS Actualizaciones de políticas administradas por	Se agregaron permisos a `AmazonEKSClusterPolicy` para permitir que Amazon EKS utilice interfaces de red elásticas creadas por la CNI de la VPC. Esto es necesario para que EKS pueda limpiar las interfaces de red elásticas que se excluyen si la CNI de la VPC se cierra inesperadamente.	16 de abril de 2025
AWS Actualizaciones de políticas administradas por	Se agregaron permisos a `AmazonEKSServiceRolePolicy` para permitir a los clientes de IA/ML de EKS añadir reglas de salida al grupo de seguridad predeterminado del clúster de EKS.	14 de abril de 2025
Estado de los nodos híbridos de EKS	Puede utilizar `eks-node-monitoring-agent` en nodos híbridos, a partir de la versión `1.2.0-eksbuild.1` . Ejecute `eks-node-monitoring-agent` como complemento de Amazon EKS para detectar y mostrar problemas de estado.	31 de marzo de 2025

Nodos híbridos de EKS para clústeres existentes	Ahora puede añadir, cambiar o eliminar la configuración de los nodos híbridos de los clústeres existentes. Anteriormente, solo podía añadir la configuración de nodos híbridos a los nuevos clústeres cuando los creaba. Con los Nodos híbridos de Amazon EKS, puede utilizar la infraestructura en las instalaciones y periférica como nodos en los clústeres de Amazon EKS. AWS administra el plano de control de Kubernetes alojado enAWS del clúster de Amazon EKS y usted administra los nodos híbridos que se ejecutan en los entornos en las instalaciones o periféricos.	31 de marzo de 2025
Reversión: evite las actualizaciones accidentales con información sobre clústeres	Amazon EKS ha revertido temporalmente una característica que obligaba a utilizar una marca `--force` para actualizar el clúster cuando se producían determinados problemas de conocimiento del clúster. Para obtener más información, consulte Temporary rollback of enforcing upgrade insights on update cluster version en GitHub.	28 de marzo de 2025

AMI aprobadas por el FIPS de Bottlerocket	Las AMI aprobadas por el FIPS de Bottlerocket ahora están disponibles en grupos de nodos administrados estándar.	27 de marzo de 2025
AWS Actualizaciones de políticas administradas por	Se agregaron permisos a `AmazonEKSServiceRolePolicy` para permitir a Amazon EKS terminar instancias de EC2 creadas por el modo automático. Se agregaron permisos a `AmazonEKSServiceRolePolicy` para permitir a Amazon EKS terminar instancias de EC2 creadas por el modo automático.	28 de febrero de 2025
Actualización de las estrategias para grupos de nodos administrados	Ahora puede usar estrategias de actualización para configurar el proceso de actualización de versiones para grupos de nodos administrados. Se introduce la estrategia de actualización mínima para terminar los nodos antes de crear otros nuevos, lo que resulta útil en entornos con capacidad limitada. La estrategia de actualización predeterminada mantiene el comportamiento existente.	27 de enero de 2025

Versión de Kubernetes 1.32	Se ha agregado compatibilidad con la versión 1.32 de Kubernetes para nuevas actualizaciones de versión y clústeres.	23 de enero de 2025
Ampliación de las regiones de AWS Amazon EKS	Amazon EKS ya se encuentra disponible en las regiones Asia-Pacífico (Tailandia) (ap-southeast-7) y México (centro) (mx-central-1) de AWS. El modo automático de EKS y los puntos de conexión de VPC para la API de EKS no se encuentran disponibles en ninguna de las regiones.	14 de enero de 2025
AWS Actualizaciones de políticas administradas por	Se agregaron varios permisos a AmazonEBSCSIDriverPolicy para permitir que el controlador de CSI de Amazon EBS restaure todas las instantáneas, habilite la restauración rápida de instantáneas (FSR) en los volúmenes de EBS y modifique las etiquetas de los volúmenes.	13 de enero de 2025
AWS Actualizaciones de políticas administradas por	Se agregaron permisos a AmazonEKSLoadBalancingPolicy .	26 de diciembre de 2024

Información actualizada del clúster	La información de actualización de Amazon EKS ahora advertirá sobre más problemas de estado del clúster y de compatibilidad de versiones Puede detectar problemas entre distintos componentes de Kubernetes y Amazon EKS, como `kubelet`, `kube-proxy` y los complementos de Amazon EKS.	20 de diciembre de 2024
Agente de supervisión de nodos y reparación automática	Puede utilizar el nuevo `eks-node-monitoring-agent` como complemento de Amazon EKS para detectar y mostrar problemas de estado. También puede habilitar la reparación automática de nodos para sustituirlos automáticamente cuando se detecten problemas.	16 de diciembre de 2024

Nodos híbridos de Amazon EKS	Ahora puede ejecutar nodos en las instalaciones conectados a clústeres de Amazon EKS. Con los Nodos híbridos de Amazon EKS, puede utilizar la infraestructura en las instalaciones y periférica como nodos en los clústeres de Amazon EKS. AWS administra el plano de control de Kubernetes alojado enAWS del clúster de Amazon EKS y usted administra los nodos híbridos que se ejecutan en los entornos en las instalaciones o periféricos.	1 de diciembre de 2024
Modo automático de Amazon EKS	El modo automático de Amazon EKS automatiza por completo la administración de la infraestructura de clústeres de Kubernetes para la computación, el almacenamiento y las redes en AWS. Simplifica la administración de Kubernetes al aprovisionar automáticamente la infraestructura, seleccionar las instancias de computación óptimas, escalar dinámicamente los recursos, optimizar continuamente los costos, aplicar revisiones a los sistemas operativos e integrarse con los servicios de seguridad de AWS.	1 de diciembre de 2024

Actualización de la versión de la plataforma de Amazon EKS	Se trata de una nueva versión de la plataforma con mejoras y correcciones de seguridad . Esto incluye las nuevas versiones de parches de Kubernetes `1.31.2`, `1.30.6`, `1.29.10` y `1.28.15`.	22 de noviembre de 2024
AWS Actualizaciones de políticas administradas por	`AWSServiceRoleForAmazonEKSNodegroup` se actualizó de modo que sea compatible con las regiones de China.	22 de noviembre de 2024
La versión 1.30 de Kubernetes ya está disponible para clústeres locales en AWS Outposts	Ahora puede crear un clúster local de Amazon EKS en AWS Outposts con la versión 1.30 de Kubernetes.	21 de noviembre de 2024
AWS Actualizaciones de políticas administradas por	EKS actualizó la política `AmazonEKSLocalOutpostClusterPolicy` administrada por AWS. Se agregó el permiso `ec2:DescribeAvailabilityZones` para que el administrador de controladores de la nube de AWS en el plano de control del clúster pueda identificar la zona de disponibilidad en la que se encuentra cada nodo.	21 de noviembre de 2024

AMI de Bottlerocket que usan FIPS 140-3	Existen AMI de Bottlerocket preconfiguradas para utilizar módulos criptográficos validados por FIPS 140-3. Entre otras, el módulo criptográfico Kernel Crypto API de Amazon Linux 2023 y el módulo criptográfico AWS-LC.	20 de noviembre de 2024
AWS Actualizaciones de políticas administradas por	Se actualizó la política `AWSServiceRoleForAmazonEKSNodegroup` para permitir `ec2:RebootInstances` para las instancias creadas por grupos de nodos administrados por Amazon EKS. Se restringieron los permisos `ec2:CreateTags` para los recursos de Amazon EC2.	20 de noviembre de 2024
Panel de observabilidad	El panel de observabilidad ayuda a detectar, solucionar y remediar rápidamente los problemas. También hay nuevas métricas suministradas por CloudWatch disponibles en el espacio de nombres de `AWS/EKS`.	18 de noviembre de 2024

AWS Actualizaciones de políticas administradas por	EKS actualizó la política administrada por AWS `AmazonEKSServiceRolePolicy` . Se han agregado permisos para las políticas de acceso a EKS, la administración del equilibrador de carga y la limpieza automatizada de los recursos del clúster.	16 de noviembre de 2024
Nueva creación de roles en la consola para los complementos compatibles con EKS Pod Identities	Se han introducido nuevos pasos al utilizar la consola para crear o actualizar complementos compatibles con EKS Pod Identities, que permiten generar automáticamente roles de IAM con el nombre, la política de roles y la política de confianza correspondientes para el complemento.	15 de noviembre de 2024
Grupos de nodos administrados en zonas locales de AWS	Ahora se pueden crear y usar grupos de nodos en zonas locales de AWS.	15 de noviembre de 2024
Nuevas métricas disponibles	Hay nuevas métricas disponibles en el grupo de la API `metrics.eks.amazonaws.com` .	11 de noviembre de 2024

AWS Actualizaciones de políticas administradas por	EKS actualizó la política administrada por AWS `AmazonEKSComputePolicy` . Se actualizaron los permisos de recursos para la acción `iam:AddRoleToInstanceProfile` .	7 de noviembre de 2024
AWS Actualizaciones de políticas administradas por	Amazon EKS ha agregado una nueva política administrada de AWS: `AmazonEKSComputePolicy`	1 de noviembre de 2024
AWS Actualizaciones de políticas administradas por	Se agregaron permisos a `AmazonEKSClusterPolicy` . Se agregó el permiso `ec2:DescribeInstanceTopology` para permitir que Amazon EKS asocie información de topología al nodo como etiquetas.	1 de noviembre de 2024
AWS Actualizaciones de políticas administradas por	Amazon EKS ha agregado una nueva política administrada de AWS: `AmazonEKSBlockStoragePolicy`	30 de octubre de 2024
AWS Actualizaciones de políticas administradas por	Amazon EKS ha agregado una nueva política administrada de AWS: `AmazonEKSLoadBalancingPolicy`	30 de octubre de 2024

AWS Actualizaciones de políticas administradas por	Se agregaron permisos `cloudwatch:PutMetricData` a `AmazonEKSServiceRolePolicy` para permitir que Amazon EKS publique métricas en Amazon CloudWatch.	29 de octubre de 2024
AWS Actualizaciones de políticas administradas por	Amazon EKS ha agregado una nueva política administrada de AWS: `AmazonEKSNetworkingPolicy`	28 de octubre de 2024
Puntos de conexión de doble pila para nuevos clústeres IPv6	Conéctese a nuevos clústeres IPv6 con un punto de conexión `eks-cluster. region .api.aws` de doble pila. Este punto de conexión se devuelve al describir estos clústeres . `kubectl` y otros clientes de la API de Kubernetes en IPv4 IPv6 o los entornos de doble pila pueden resolver estos puntos de conexión y conectarse a ellos para clústeres públicos o privados.	21 de octubre de 2024

AWS Actualizaciones de políticas administradas por	Se agregaron `autoscaling:ResumeProcesses` , `autoscaling:SuspendProcesses` y los permisos asociados a `AWSServiceRoleForAmazonEKSNodegroup` en las regiones de China para integrarlos con el Controlador de recuperación de aplicaciones de Amazon para EKS. Sin cambios en otras regiones.	21 de octubre de 2024
AMI aceleradas de AL2023	Ahora puede utilizar las instancias aceleradas de AWS Neuron y `NVIDIA` para las AMI basadas en AL2023.	11 de octubre de 2024
Nuevo formato de origen	Hemos cambiado a un nuevo formato fuente con algunos cambios de diseño. Estamos solucionando algunos problemas temporales de formato.	10 de octubre de 2024
AWS Actualizaciones de políticas administradas por	Se han añadido permisos a `AmazonEKSServicePolicy` y `AmazonEKSServiceRolePolicy` . Se han añadido `ec2:GetSecurityGroupsForVpc` y asociado permisos de etiquetas para permitir a EKS leer la información del grupo de seguridad y actualizar las etiquetas relacionadas.	10 de octubre de 2024

AWS Actualizaciones de políticas administradas de : nueva política	EKS ha agregado una nueva política administrada de AWS.	3 de octubre de 2024
Versión de Kubernetes `1.31`	Se ha agregado compatibilidad con la versión `1.31` de Kubernetes para nuevas actualizaciones de versión y clústeres.	24 de septiembre de 2024
AWS Actualizaciones de política administrada de : actualización de una política existente	Amazon EKS ha actualizado una política administrada existente de AWS.	21 de agosto de 2024
La versión 1.29 de Kubernetes ya está disponible para clústeres locales en AWS Outposts	Ahora puede crear un clúster local de Amazon EKS en AWS Outposts con la versión 1.29 de Kubernetes.	20 de agosto de 2024
Pod Identity de EKS en AWS GovCloud (EE. UU)	Pod Identities de Amazon EKS asocian el rol de IAM con una cuenta de servicio de Kubernetes. Con esta característica, ya no es necesario proporcionar permisos extendidos al rol de IAM del nodo. De esta forma, los pods de ese nodo pueden llamar a las API de AWS. A diferencia de los roles de IAM para cuentas de servicio, las Pod Identities de EKS están completamente integradas en EKS, lo que significa que no necesita un proveedor de identidades de OIDC.	14 de agosto de 2024

Actualizaciones de contenido basadas en escenarios	Cambiamos el nombre de los temas y los actualizamos para que se basen más en los escenarios en toda la guía.	9 de agosto de 2024
Puntos de conexión de interfaz de VPC de doble pila para Amazon EKS	Ahora puede crear puntos de conexión de interfaz de VPC de doble pila para Amazon EKS con direcciones IP `IPv4` y `IPv6` y nombres de DNS.	7 de agosto de 2024
Nuevos puntos de conexión de doble pila para las API de Amazon EKS con direcciones `IPv6`	La API de EKS para crear y administrar clústeres, así como las URL de los emisores de OIDC para los clústeres , ahora cuentan con nuevos puntos de conexión de doble pila. El nuevo nombre de DNS de la API de Amazon EKS es `eks.`*`region`*`.api.aws`, que se resuelve en direccion es `IPv4` y `IPv6`. Los nuevos clústeres tienen una nueva URL de emisor de OIDC de doble pila (`oidc-eks.`*`region`*`.api.aws`).	1 de agosto de 2024
Bloques de capacidad para grupos de nodos administrados	Ahora puede usar bloques de capacidad para grupos de nodos administrados.	1 de julio de 2024

La recopilación de métricas de grupo de escalado automático se habilita de forma predeterminada	Los grupos de nodos administrados de Amazon EKS ahora tienen las métricas de grupos de Amazon EC2 Auto Scaling habilitadas de forma predeterminada y sin cargo adicional. Anteriormente, tenía que llevar a cabo varios pasos para habilitar esta característica.	28 de junio de 2024
AWS Actualizaciones de política administrada de : actualización de una política existente	Amazon EKS ha actualizado una política administrada existente de AWS.	27 de junio de 2024
Versión de Kubernetes `1.26`	La versión `1.26` de Kubernetes ahora cuenta con soporte extendido.	12 de junio de 2024
Mejoras en las referencias de información de la AMI	Hemos mejorado las referencias de información de la AMI, en particular, para Bottlerocket.	12 de junio de 2024
Versión de Kubernetes `1.30`	Se ha agregado compatibilidad con la versión `1.30` de Kubernetes para nuevas actualizaciones de versión y clústeres.	23 de mayo de 2024

Escalado automático de CoreDNS	El Escalador automático de CoreDNS adaptará dinámicamente la cantidad de réplicas de la implementación de CoreDNS en un clúster de EKS en función de la cantidad de nodos y núcleos de CPU. Esta característica funciona en CoreDNS `v1.9` y en la versión de la plataforma de lanzamiento de EKS `1.25` y versiones posteriores.	14 de mayo de 2024
Actualización de la versión de la plataforma de Amazon EKS	Se trata de una nueva versión de la plataforma con mejoras y correcciones de seguridad. Esto incluye las nuevas versiones de parches de Kubernetes `1.29.4`, `1.28.9` y `1.27.13`.	14 de mayo de 2024
Compatibilidad con Información de contenedores de CloudWatch para Windows	El complemento Operador de observabilidad de Amazon CloudWatch ahora también permite Información de contenedores en nodos de trabajo de Windows en el clúster.	10 de abril de 2024
Conceptos de Kubernetes	Se ha agregado un nuevo tema sobre los conceptos de Kubernetes.	5 de abril de 2024

Reestructuración del acceso y el contenido de IAM	Mueva las páginas existentes relacionadas con temas de acceso e IAM, como el mapa de configuración de autenticación, las entradas de acceso, el ID de pod y el IRSA, a una nueva sección. Revise el contenido de la descripción general.	2 de abril de 2024
Sistema operativo de Bottlerocket compatible con el controlador CSI de Amazon S3	El controlador CSI de Mountpoint para Amazon S3 ya es compatible con Bottlerocket.	13 de marzo de 2024
AWS Actualizaciones de política administrada de : actualización de una política existente	Amazon EKS ha actualizado una política administrada existente de AWS.	4 de marzo de 2024
Amazon Linux 2023	Amazon Linux 2023 (AL2023) es un nuevo sistema operativo basado en Linux diseñado para proporcionar un entorno seguro, estable y de alto rendimiento para las aplicaciones en la nube.	29 de febrero de 2024

Pod Identity de EKS e IRSA admiten asociados en Kubernetes 1.29	En Kubernetes 1.29, los contenedores asociados están disponibles en los clústeres de Amazon EKS. Los contenedores asociados son compatibles con los roles de IAM para las cuentas de servicio o Pod Identity de EKS. Para obtener más información sobre estos asociados, consulte Sidecar Containers en la documentación de Kubernetes.	26 de febrero de 2024
Versión de Kubernetes 1.29	Se ha agregado compatibilidad con la versión 1.29 de Kubernetes para nuevas actualizaciones de versión y clústeres.	23 de enero de 2024
Versión completa: soporte extendido de Amazon EKS para las versiones de Kubernetes	El soporte extendido de las versiones de Kubernetes permite permanecer en una versión específica de Kubernetes durante más de 14 meses.	16 de enero de 2024

Detección del estado del clúster de Amazon EKS en la Nube de AWS	Amazon EKS detecta problemas con los clústeres de Amazon EKS y con la infraestructura de los requisitos previos del clúster en el estado del clúster. Puede ver los problemas relacionados con sus clústeres de EKS dentro de la Consola de administración de AWS y en el `health` del clúster en la API de EKS. Estos problemas se suman a los problemas detectados y mostrados por la consola. Anteriormente, el estado del clúster solo estaba disponible para los clústeres locales en AWS Outposts.	28 de diciembre de 2023
Información sobre clústeres	Ahora puede obtener recomendaciones sobre su clúster en función de las comprobaciones periódicas.	20 de diciembre de 2023
Ampliación de las regiones de AWS en Amazon EKS	Amazon EKS ya está disponible en la región de AWS del Oeste de Canadá (Calgary) (`ca-west-1`).	20 de diciembre de 2023

Ahora puede conceder a los usuarios y roles de IAM el acceso a su clúster mediante las entradas de acceso	Antes de introducir las entradas de acceso, concedió a los usuarios y roles de IAM el acceso a su clúster añadiendo entradas a `aws-auth ConfigMap` . Ahora, cada clúster tiene un modo de acceso y puede cambiar a utilizar las entradas de acceso según su horario. Tras cambiar de modo, puede añadir usuarios agregando entradas de acceso en la AWS CLI, AWS CloudFormation y en los SDK de AWS.	18 de diciembre de 2023
Actualización de la versión de la plataforma de Amazon EKS	Se trata de una nueva versión de la plataforma con mejoras y correcciones de seguridad . Esto incluye las nuevas versiones de parches de Kubernetes `1.28.4`, `1.27.8`, `1.26.11` y `1.25.16`.	12 de diciembre de 2023
Controlador CSI de Mountpoint para Amazon S3	Ahora puede instalar el controlador CSI de Mountpoint para Amazon S3 en los clústeres de Amazon EKS.	27 de noviembre de 2023
Activación de las métricas de Prometheus al crear un clúster	En la Consola de administración de AWS, ahora puede activar las métricas de Prometheus al crear un clúster. También puede ver los detalles del raspador de Prometheus en la pestaña Observabilidad.	26 de noviembre de 2023

Pod Identities de Amazon EKS	Pod Identities de Amazon EKS asocian el rol de IAM con una cuenta de servicio de Kubernetes. Con esta característica, ya no es necesario proporcionar permisos extendidos al rol de IAM del nodo. De esta forma, los pods de ese nodo pueden llamar a las API de AWS. A diferencia de los roles de IAM para cuentas de servicio, las Pod Identities de EKS están completamente integradas en EKS, lo que significa que no necesita un proveedor de identidades de OIDC.	26 de noviembre de 2023
AWS Actualizaciones de política administrada de : actualización de una política existente	Amazon EKS ha actualiza do una política administrada existente de AWS.	26 de noviembre de 2023
Controlador de instantáneas CSI	Ahora puede instalar el controlador de instantáneas CSI para usarlo con controlad ores CSI compatibles, como el controlador CSI de Amazon EBS.	17 de noviembre de 2023

Reescritura del tema ADOT Operator	La sección de soporte del complemento de Amazon EKS para ADOT Operator era redundante con la documentación de AWS Distro para OpenTelemetry. Migramos la información esencial restante a ese recurso para reducir la información obsoleta e incoherente.	14 de noviembre de 2023
Compatibilidad del complemento de EKS CoreDNS para las métricas de Prometheus	Las versiones `v1.10.1-eksbuild.5` , `v1.9.3-eksbuild.9` y `v1.8.7-eksbuild.8` del complemento de EKS para CoreDNS exponen el puerto en el que CoreDNS publicó las métricas, en el servicio `kube-dns`. Esto facilita la inclusión de las métricas de CoreDNS en sus sistemas de supervisión.	10 de noviembre de 2023
Complemento Amazon EKS CloudWatch Observability Operator	Se agregó la página Amazon EKS CloudWatch Observability Operator.	6 de noviembre de 2023
Bloques de capacidad para instancias P5 autoadministradas en la región Este de EE. UU. (Ohio)	En la región Este de EE. UU. (Ohio), ahora puede utilizar bloques de capacidad para instancias P5 autoadministradas.	31 de octubre de 2023

Los clústeres permiten modificar las subredes y los grupos de seguridad	Puede actualizar el clúster para cambiar las subredes y los grupos de seguridad que utiliza el clúster. Puede actualizar desde la Consola de administración de AWS, la última versión de la AWS CLI, AWS CloudFormation, y `eksctl` de la versión `v0.164.0-rc.0` o posterior . Es posible que tenga que hacer esto para proporcionar a las subredes más direccion es IP disponibles con las que poder actualizar correctam ente la versión de un clúster.	24 de octubre de 2023
El rol del clúster y el rol del grupo de nodos administr ados son compatibles con las políticas de AWS Identity and Access Management administr adas por el cliente	Puede utilizar una política de IAM personalizada en el rol del clúster en lugar de la política administrada AmazonEKS ClusterPolicy de AWS. Además, puede utilizar una política de IAM personalizada en el rol del nodo de un grupo de nodos administrados en lugar de la política administr ada AmazonEKSWorkerNod ePolicy de AWS. Haga esto para crear una política con el privilegio mínimo para cumplir con exigencias de conformid ad estrictas.	23 de octubre de 2023

Corregido el enlace a la instalación de eksctl	Corregido el enlace de instalación de eksctl después de trasladar la página.	6 de octubre de 2023
Versión preliminar: soporte extendido de Amazon EKS para las versiones de Kubernetes	El soporte extendido de las versiones de Kubernetes permite permanecer en una versión específica de Kubernetes durante más de 14 meses.	4 de octubre de 2023
Eliminación de las referencias a la integración de AWS App Mesh	Las integraciones de Amazon EKS con AWS App Mesh solo se conservarán para los clientes actuales de App Mesh.	29 de septiembre de 2023
Versión de Kubernetes `1.28`	Se ha agregado compatibilidad con la versión `1.28` de Kubernetes para nuevas actualizaciones de versión y clústeres.	26 de septiembre de 2023
Los clústeres existentes admiten la aplicación de la política de red de Kubernetes en el complemento CNI de Amazon VPC para Kubernetes	Puede utilizar la política de red de Kubernetes en los clústeres existentes con el complemento CNI de Amazon VPC para Kubernetes, en lugar de tener que usar una solución de terceros. Puede utilizar la política de red de Kubernetes en los clústeres existentes con el complemento CNI de Amazon VPC para Kubernetes, en lugar de tener que usar una solución de terceros.	15 de septiembre de 2023

El complemento de Amazon EKS para CoreDNS admite la modificación de PDB	Puede modificar `PodDisruptionBudget` del complemento de EKS para CoreDNS en las versiones `v1.9.3-eksbuild.7` y posteriores, así como `v1.10.1-eksbuild.4` y posteriores.	15 de septiembre de 2023
Soporte de Amazon EKS para subredes compartidas	Nuevos Requisitos y consideraciones sobre las subredes compartidas para crear clústeres de Amazon EKS en subredes compartidas.	7 de septiembre de 2023
Actualizaciones de ¿Qué es Amazon EKS?	Se agregaron los temas nuevos Casos de uso comunes y Arquitectura. Se actualizaron otros temas.	6 de septiembre de 2023
Aplicación de la política de red de Kubernetes en el complemento CNI de Amazon VPC para Kubernetes	Puede utilizar la política de red de Kubernetes con el complemento CNI de Amazon VPC para Kubernetes, en lugar de tener que usar una solución de terceros. Puede utilizar la política de red de Kubernetes con el complemento CNI de Amazon VPC para Kubernetes, en lugar de tener que usar una solución de terceros.	29 de agosto de 2023
Ampliación de las regiones de AWS de Amazon EKS	Amazon EKS ya está disponible en la región de AWS de Israel (Tel Aviv) (`il-central-1`).	1 de agosto de 2023

Almacenamiento efímero configurable para Fargate	Puede aumentar la cantidad total de almacenamiento efímero para cada pod que se ejecuta en Fargate de Amazon EKS.	31 de julio de 2023
Compatibilidad del complemento para el controlador de CSI de Amazon EFS	A partir de ahora, puede utilizar la Consola de administración de AWS, la CLI de AWS y la API para administrar el controlador de CSI de Amazon EFS.	26 de julio de 2023
AWS Actualizaciones de políticas administradas de : nueva política	Amazon EKS ha agregado una nueva política administrada de AWS.	26 de julio de 2023
Las actualizaciones de Kubernetes de las versiones 1.27, 1.26, 1.25 y 1.24 ya están disponibles para los clústeres locales en AWS Outposts	Las actualizaciones de Kubernetes de las versiones 1.27.3, 1.26.6, 1.25.11 y 1.24.15 ya están disponibles para los clústeres locales en AWS Outposts.	20 de julio de 2023
Soporte de prefijos IP para nodos de Windows	La asignación de prefijos IP a los nodos puede permitirle alojar un número significativamente mayor de pods en sus nodos que al asignar direcciones IP secundarias individuales a sus nodos.	6 de julio de 2023
Controlador de CSI de Amazon FSx para OpenZFS	Ahora puede instalar el controlador CSI de Amazon FSx para OpenZFS en los clústeres de Amazon EKS.	30 de junio de 2023

Los nodos de pods en Linux en clústeres IPv4 ahora pueden comunicarse con puntos de conexión IPv6.	Después de asignar una dirección IPv6 a su nodo, la dirección IPv4 de los pods es la dirección de red traducida a la dirección IPv6 del nodo en el que se ejecuta.	19 de junio de 2023
Grupos de nodos administrados de Windows en las regiones de AWS GovCloud (EE. UU.)	En las regiones de AWS GovCloud (EE. UU.), los grupos de nodos administrados de Amazon EKS ahora pueden ejecutar contenedores de Windows.	30 de mayo de 2023
Versión de Kubernetes 1.27	Se ha agregado compatibilidad con la versión 1.27 de Kubernetes para nuevas actualizaciones de versión y clústeres.	24 de mayo de 2023
Versión de Kubernetes 1.26	Se ha agregado compatibilidad con la versión 1.26 de Kubernetes para nuevas actualizaciones de versión y clústeres.	11 de abril de 2023
gMSA sin dominio	Ahora puede usar gMSA sin dominio con pods de Windows.	27 de marzo de 2023
Ampliación de las regiones de AWS de Amazon EKS	Amazon EKS ya está disponible en la región de AWS de Asia-Pacífico (Melbourne) (ap-southeast-4).	10 de marzo de 2023

Controlador CSI de Amazon File Cache	Ahora puede instalar el controlador CSI de Amazon File Cache en los clústeres de Amazon EKS.	3 de marzo de 2023
La versión 1.25 de Kubernetes ya está disponible para clústeres locales en AWS Outposts	Ahora puede crear un clúster local de Amazon EKS en un Outpost mediante las versiones de Kubernetes de la `1.22` a la `1.25`.	1 de marzo de 2023
Versión de Kubernetes `1.25`	Se ha agregado compatibilidad con la versión `1.25` de Kubernetes para nuevas actualizaciones de versión y clústeres.	22 de febrero de 2023
AWS Actualizaciones de política administrada de : actualización de una política existente	Amazon EKS ha actualizado una política administrada existente de AWS.	7 de febrero de 2023
Ampliación de las regiones de AWS de Amazon EKS	Amazon EKS ya está disponible en las regiones de AWS de Asia Pacífico (Hyderabad) (`ap-south-2`), Europa (Zúrich) (`eu-central-2`) y Europa (España) (`eu-south-2`).	6 de febrero de 2023
Las versiones de la `1.21` a la `1.24` de Kubernetes ahora están disponibles para los clústeres locales en AWS Outposts.	Ahora puede crear un clúster local de Amazon EKS en un Outpost mediante las versiones de Kubernetes de la `1.21` a la `1.24`. Anteriormente, solo estaba disponible la versión `1.21`.	17 de enero de 2023

Amazon EKS ahora admite AWS PrivateLink	Puede utilizar AWS PrivateLink para crear una conexión privada entre la VPC y Amazon EKS.	16 de diciembre de 2022
Soporte de Windows para grupos de nodos administrados	Ahora puede usar Windows para grupos de nodos administrados por Amazon EKS.	15 de diciembre de 2022
Los complementos de Amazon EKS de proveedores de software independientes ya están disponibles en AWS Marketplace	Ahora puede buscar y suscribirse a los complementos de Amazon EKS de proveedores de software independientes a través de AWS Marketplace.	28 de noviembre de 2022
AWS Actualizaciones de política administrada de : actualización de una política existente	Amazon EKS ha actualizado una política administrada existente de AWS.	17 de noviembre de 2022
Versión de Kubernetes `1.24`	Se ha agregado compatibilidad con la versión `1.24` de Kubernetes para nuevas actualizaciones de versión y clústeres.	15 de noviembre de 2022
Ampliación de las regiones de AWS de Amazon EKS	Amazon EKS ya está disponible en la región de AWS de Medio Oriente (EAU) (`me-central-1`).	3 de noviembre de 2022
AWS Actualizaciones de política administrada de : actualización de una política existente	Amazon EKS ha actualizado una política administrada existente de AWS.	24 de octubre de 2022

AWS Actualizaciones de política administrada de : actualización de una política existente	Amazon EKS ha actualiza do una política administrada existente de AWS.	20 de octubre de 2022
Ya están disponibles los clústeres locales en AWS Outposts	Ahora puede crear un clúster local de Amazon EKS en un Outpost.	19 de septiembre de 2022
Cuotas basadas en vCPU de Fargate	Fargate pasa de cuotas basadas en pods a cuotas basadas en vCPU.	8 de septiembre de 2022
AWS Actualizaciones de política administrada de : actualización de una política existente	Amazon EKS ha actualiza do una política administrada existente de AWS.	31 de agosto de 2022
Supervisión de costos	Amazon EKS ya admite Kubecost, que le permite supervisar los costos desglosados por los recursos de Kubernetes que incluyen pods, nodos, espacios de nombres y etiquetas.	24 de agosto de 2022
AWS Actualizaciones de políticas administradas de : nueva política	Amazon EKS ha agregado una nueva política administr ada de AWS.	24 de agosto de 2022
AWS Actualizaciones de políticas administradas de : nueva política	Amazon EKS ha agregado una nueva política administr ada de AWS.	23 de agosto de 2022

Recursos de etiquetas para facturación	Se agregó el soporte de la etiqueta de asignación de costos generada por `aws:eks:cluster-name` para todos los clústeres.	16 de agosto de 2022
Comodines de perfil de Fargate	Se agregó soporte para los comodines de perfil de Fargate en los criterios del selector para los espacios de nombres, las claves de las etiquetas y los valores de las etiquetas.	16 de agosto de 2022
Versión de Kubernetes `1.23`	Se ha agregado compatibilidad con la versión `1.23` de Kubernetes para nuevas actualizaciones de versión y clústeres.	11 de agosto de 2022
Visualización de los recursos de Kubernetes en la Consola de administración de AWS	Ahora puede ver la información sobre los recursos de Kubernetes desplegados en el clúster mediante la Consola de administración de AWS.	3 de mayo de 2022
Ampliación de las regiones de AWS de Amazon EKS	Amazon EKS ya está disponible en la región de AWS de Asia-Pacífico (Yakarta) (`ap-southeast-3`).	2 de mayo de 2022
Compatibilidad con la página de observabilidad y el complemento ADOT	Página Observabilidad añadida y AWS Distro for OpenTelemetry (ADOT).	21 de abril de 2022

Versión de Kubernetes `1.22`	Se ha agregado compatibilidad con la versión `1.22` de Kubernetes para nuevas actualizaciones de versión y clústeres.	4 de abril de 2022
AWS Actualizaciones de políticas administradas de : nueva política	Amazon EKS ha agregado una nueva política administrada de AWS.	4 de abril de 2022
Se agregaron detalles sobre la creación de parches de pod de Fargate	Al actualizar los pods de Fargate, Amazon EKS intenta desalojar los pods en expulsión de los presupuestos de interrupción de los pods. Puede crear reglas de eventos para reaccionar ante las expulsiones fallidas antes de que se eliminen los pods.	1 de abril de 2022
Lanzamiento completo: compatibilidad del complemento para el controlador de CSI de Amazon EBS	A partir de ahora, puede utilizar la Consola de administración de AWS, la CLI de AWS y la API para administrar el controlador de CSI de Amazon EBS.	31 de marzo de 2022
AWS Actualización del contenido de Outposts	Instrucciones para implementar un clúster de Amazon EKS en AWS Outposts.	22 de marzo de 2022
AWS Actualizaciones de política administrada de : actualización de una política existente	Amazon EKS ha actualizado una política administrada existente de AWS.	21 de marzo de 2022

Compatibilidad con Windows `containerd`	Ahora puede seleccionar el tiempo de funcionamiento `containerd` para nodos de Windows.	14 de marzo de 2022
Se agregaron consideraciones sobre Amazon EKS Connector a la documentación de seguridad	Describe el modelo de responsabilidad compartida en lo que se refiere a los clústeres conectados.	25 de febrero de 2022
Asignación de direcciones `IPv6` a los pods y los servicios	Ahora, puede crear un clúster `1.21` o posterior que asigne direcciones `IPv6` a sus pods y servicios.	6 de enero de 2022
AWS Actualizaciones de política administrada de : actualización de una política existente	Amazon EKS ha actualizado una política administrada existente de AWS.	13 de diciembre de 2021
Lanzamiento previa: compatibilidad del complemento para el controlador de CSI de Amazon EBS	A partir de ahora, puede realizar una vista previa de la Consola de administración de AWS, la CLI de AWS y la API para administrar el controlador de CSI de Amazon EBS.	9 de diciembre de 2021
Compatibilidad con el escalador automático de Karpenter	Ahora puede utilizar el proyecto de código abierto Karpenter para escalar los nodos de forma automática.	29 de noviembre de 2021
Compatibilidad con filtros de Fluent Bit de Kubernetes en el registro de Fargate	Ahora puede utilizar el filtro de Fluent Bit de Kubernetes con el registro de Fargate.	10 de noviembre de 2021

Compatibilidad con Windows disponible en el plano de control	La compatibilidad con Windows ahora se encuentra disponible en el plano de control. Ya no tendrá que habilitarla en su plano de datos.	9 de noviembre de 2021
Se ha agregado Bottlerocket como tipo de AMI para grupos de nodos administrados	Anteriormente, Bottlerocket solo estaba disponible como opción de nodo autoadmin istrado. Ahora se puede configurar como un grupo de nodos administrados, lo que reduce el esfuerzo necesario para cumplir los requisitos de conformidad de nodos.	28 de octubre de 2021
Compatibilidad con controlad ores DL1	Las AMI personalizadas de Amazon Linux ahora admiten cargas de trabajo de aprendizaje profundo para Linux 2 de Amazon. Esta habilitación permite una configuración genérica de base de referencia en las instalaciones o en la nube.	25 de octubre de 2021
Soporte de video VT1	Las AMI personalizadas de Amazon Linux ahora admiten VT1 para algunas distribuciones. Esta habilitac ión anuncia dispositivos Xilinx U30 en su clúster de Amazon EKS.	13 de septiembre de 2021

Amazon EKS Connector ya está disponible	Puede utilizar Amazon EKS Connector a fin de registrar y conectar cualquier clúster de Kubernetes conforme a AWS y visualizarlo en la consola de Amazon EKS.	8 de septiembre de 2021
Amazon EKS Anywhere ya está disponible	Amazon EKS Anywhere es una nueva opción de implementación para Amazon EKS que puede usar para crear y operar fácilmente clústeres de Kubernetes en las instalaciones.	8 de septiembre de 2021
Controlador de CSI de Amazon FSx para ONTAP de NetApp	Se ha agregado un tema que resume el controlador de CSI de Amazon FSx para ONTAP de NetApp y proporciona enlaces a otras referencias.	2 de septiembre de 2021
Los grupos de nodos administrados calculan automáticamente el número máximo de pods recomendados por Amazon EKS para los nodos	Los grupos de nodos administrados ahora calculan automáticamente el número máximo de pods de Amazon EKS para los nodos que implementa sin una plantilla de lanzamiento o con una plantilla de lanzamiento en la que no ha especificado un ID de AMI.	30 de agosto de 2021
Eliminación de la administración de Amazon EKS de la configuración del complemento sin eliminar el software del complemento de Amazon EKS	Ahora puede eliminar un complemento de Amazon EKS sin quitar el software del complemento del clúster.	20 de agosto de 2021

Creación de pods de múltiples anfitriones mediante Multus	Ahora puede agregar varias interfaces de red a un pod mediante Multus.	2 de agosto de 2021
Agregar más direcciones IP a los nodos Linux de Amazon EC2	Ahora puede agregar una cantidad significativa de direcciones IP más a los nodos de Amazon EC2 de Linux. Esto significa que puede ejecutar una mayor densidad de pods en cada nodo. Ahora puede agregar una cantidad significativa de direcciones IP más a los nodos de Amazon EC2 de Linux. Esto significa que puede ejecutar una mayor densidad de pods en cada nodo.	27 de julio de 2021
`containerd` Arranque en tiempo de ejecución de	La Imagen de máquina de Amazon (AMI) acelerada , optimizada para Amazon EKS de Amazon Linux, ahora contiene un indicador de arranque para habilitar opcionalmente el tiempo de ejecución de `containerd` en las AMI de Bottlerocket y optimizadas para Amazon EKS. Este indicador está disponible en todas las versiones de Kubernetes compatibles de la AMI.	19 de julio de 2021
Versión de Kubernetes `1.21`	La versión `1.21` de Kubernetes ahora es compatible.	19 de julio de 2021

Se ha agregado un tema de políticas administradas	Una lista de todas las políticas administradas por IAM de Amazon EKS y los cambios realizados en ellas desde el 17 de junio de 2021.	17 de junio de 2021
Uso de grupos de seguridad para pods con Fargate	Ahora puede utilizar grupos de seguridad para pods con Fargate, además de utilizarlos con nodos de Amazon EC2.	1 de junio de 2021
Se han agregado complementos de CoreDNS y `kube-proxy` de Amazon EKS	Amazon EKS ahora puede ayudarlo a administrar los complementos de CoreDNS y `kube-proxy` de Amazon EKS para su clúster.	19 de mayo de 2021
Versión de Kubernetes `1.20`	Se ha agregado compatibilidad con la versión `1.20` de Kubernetes para nuevas actualizaciones de versión y clústeres.	18 de mayo de 2021
Lanzamiento del Controlador del equilibrador de carga de AWS `2.2.0`	A partir de ahora, puede utilizar el controlador de balanceador de carga de AWS para crear Elastic Load Balancers mediante instancias o destinos IP.	14 de mayo de 2021
Taints de nodos para grupos de nodos administrados	Amazon EKS ahora admite la agregación de taints de notas a los grupos de nodos administrados.	11 de mayo de 2021
Cifrado de secretos para clústeres existentes	Amazon EKS ahora admite la adición de cifrado de secretos a los clústeres existentes.	26 de febrero de 2021

Versión de Kubernetes 1.19	Se ha agregado compatibilidad con la versión 1.19 de Kubernetes para nuevas actualizaciones de versión y clústeres.	16 de febrero de 2021
Amazon EKS ahora admite la utilización de proveedores de identidad OpenID Connect (OIDC) como método para autenticar usuarios de la versión 1.16 o posterior.	Los proveedores de identidad de OIDC se pueden utilizar con AWS Identity and Access Management (IAM) o como una alternativa de IAM.	12 de febrero de 2021
Visualización de los recursos de nodo y de la carga de trabajo en la Consola de administración de AWS	Ahora puede ver detalles sobre los nodos administrados, autoadministrados y de Fargate, así como las cargas de trabajo de Kubernetes implementadas en la Consola de administración de AWS.	1 de diciembre de 2020
Implemente tipos de instancias de spot en un grupo de nodos administrado	Ahora puede implementar varios tipos de instancias de spot o instancias bajo demanda en un grupo de nodos administrado.	1 de diciembre de 2020
Amazon EKS ahora puede administrar complementos específicos para su clúster	Puede administrar los complementos o dejar que Amazon EKS controle el lanzamiento y la versión de un complemento a través de la API de Amazon EKS.	1 de diciembre de 2020

Comparta un ALB en varias entradas	A partir de ahora, puede compartir un Application Load Balancer (ALB) de AWS entre múltiples entradas de Kubernetes. En el pasado, tenía que implementar un ALB independiente para cada entrada.	23 de octubre de 2020
Compatibilidad con destino IP NLB	Ahora puede implementar un equilibrador de carga de red con destinos IP. Esto significa que puede utilizar un NLB para equilibrar la carga del tráfico de red a los pods de Fargate y directamente a los pods que se ejecutan en nodos de Amazon EC2.	23 de octubre de 2020
Versión de Kubernetes `1.18`	Se ha agregado compatibilidad con la versión `1.18` de Kubernetes para nuevas actualizaciones de versión y clústeres.	13 de octubre de 2020
Especifique un bloque de CIDR personalizado para la asignación de direcciones IP del servicio Kubernetes.	Ahora puede especificar un bloque de CIDR personalizado desde el que Kubernetes asigna direcciones IP de servicio.	29 de septiembre de 2020
Asignación de los grupos de seguridad a pods individuales	Ahora puede asociar diferentes grupos de seguridad a algunos de los pods individuales que se ejecutan en muchos tipos de instancias de Amazon EC2.	9 de septiembre de 2020

Implementación de Bottlerocket en sus nodos	Ahora puede implementar nodos que ejecuten Bottlerocket.	31 de agosto de 2020
La capacidad de lanzar nodos de Arm normalmente está disponible	Ahora puede lanzar nodos de Arm en grupos de nodos administrados y autoadministrados.	17 de agosto de 2020
Plantillas de lanzamiento de grupos de nodos administrados y AMI personalizadas	Ahora puede implementar un grupo de nodos administrado mediante una plantilla de lanzamiento de Amazon EC2. Si lo desea, la plantilla de lanzamiento puede especificar una AMI personalizada.	17 de agosto de 2020
Compatibilidad de EFS con AWS Fargate	Ahora puede utilizar Amazon EFS con AWS Fargate.	17 de agosto de 2020
Actualización de la versión de la plataforma de Amazon EKS	Se trata de una nueva versión de la plataforma con mejoras y correcciones de seguridad. Esto incluye soporte UDP para servicios de tipo `LoadBalancer` cuando se utilizan equilibradores de carga de red con la versión `1.15` de Kubernetes o posterior. Para obtener más información, consulte Permitir UDP para el equilibrador de carga de red de AWS en GitHub.	12 de agosto de 2020

Ampliación de las regiones de AWS de Amazon EKS	Amazon EKS ya está disponible en las regiones de AWS de África (Ciudad del Cabo) (`af-south-1`) y Europa (Milán) (`eu-south-1`).	6 de agosto de 2020
Métricas de uso de Fargate	AWS Fargate incorpora métricas de uso de CloudWatch que proporcionan visibilidad del uso de recursos bajo demanda de Fargate de su cuenta.	3 de agosto de 2020
Versión de Kubernetes `1.17`	Se ha agregado compatibilidad con la versión `1.17` de Kubernetes para nuevas actualizaciones de versión y clústeres.	10 de julio de 2020
Crear y administrar recursos de App Mesh desde Kubernetes con el controlador de App Mesh para Kubernetes	Puede crear y administrar recursos de App Mesh desde Kubernetes. El controlador también inyecta automáticamente el proxy de Envoy y los contenedores init en los pods que se implementan.	18 de junio de 2020
Amazon EKS ahora admite nodos Inf1 de Amazon EC2	Puede agregar nodos Inf1 de Amazon EC2 al clúster.	4 de junio de 2020
Ampliación de las regiones de AWS de Amazon EKS	Amazon EKS ya está disponible en las regiones de AWS de AWS GovCloud (EE. UU. Este) (`us-gov-east-1`) y AWS GovCloud (EE. UU. Oeste) (`us-gov-west-1`).	13 de mayo de 2020

La versión 1.12 de Kubernetes ya no se admite en Amazon EKS	La versión 1.12 de Kubernetes ya no se admite en Amazon EKS. Actualice todos los clústeres 1.12 a la versión 1.13 o posterior para evitar la interrupción del servicio.	12 de mayo de 2020
Versión de Kubernetes 1.16	Se ha agregado compatibilidad con la versión 1.16 de Kubernetes para nuevas actualizaciones de versión y clústeres.	30 de abril de 2020
Se ha agregado el rol vinculado a servicios AWSServiceRoleForAmazonEKS	Se ha agregado el rol vinculado al servicio de AWSServiceRoleForAmazonEKS.	16 de abril de 2020
Versión de Kubernetes 1.15	Se ha agregado compatibilidad con la versión 1.15 de Kubernetes para nuevas actualizaciones de versión y clústeres.	10 de marzo de 2020
Ampliación de las regiones de AWS de Amazon EKS	Amazon EKS ahora está disponible en las regiones de AWS de Beijing (cn-north-1) y Ningxia (cn-northwest-1).	26 de febrero de 2020
Controlador de CSI de FSx para Lustre	Se ha agregado el tema para instalar el controlador CSI de FSx para Lustre en clústeres de Amazon EKS de Kubernetes 1.14.	23 de diciembre de 2019

Restrinja el acceso a la red al punto de conexión de acceso público de un clúster	Con esta actualización, puede utilizar Amazon EKS para restringir los rangos de CIDR que pueden comunicarse con el punto de enlace de acceso público del servidor de API de Kubernetes.	20 de diciembre de 2019
Resuelva la dirección de punto de conexión de acceso privado para un clúster desde fuera de una VPC	Con esta actualización, puede utilizar Amazon EKS para resolver el punto de enlace de acceso privado del servidor de API de Kubernetes desde fuera de una VPC.	13 de diciembre de 2019
(Beta) Nodos de instancia de Amazon EC2 de Amazon EC2 A1	Lanzamiento de nodos de instancia de Amazon EC2 de Amazon EC2 A1 que se registran con su clúster de Amazon EKS.	4 de diciembre de 2019
Creación de un clúster en Outposts de AWS	Amazon EKS ahora admite la creación de clústeres en AWS Outposts.	3 de diciembre de 2019
AWS Fargate en Amazon EKS	Los clústeres de Kubernete s de Amazon EKS ahora admiten la ejecución de pods en Fargate.	3 de diciembre de 2019
Ampliación de las regiones de AWS de Amazon EKS	Amazon EKS ya está disponibl e en la región de Canadá (centro) (`ca-central-1`) Región de AWS.	21 de noviembre de 2019

Grupos de nodos administrados	Los grupos de nodos administrados por Amazon EKS automatizan el aprovisionamiento y la gestión del ciclo de vida de nodos (instancias de Amazon EC2) para clústeres de Kubernetes de Amazon EKS.	18 de noviembre de 2019
Actualización de la versión de la plataforma de Amazon EKS	Nuevas versiones de la plataforma como respuesta a CVE-2019-11253.	6 de noviembre de 2019
La versión `1.11` de Kubernetes ya no se admite en Amazon EKS	La versión `1.11` de Kubernetes ya no se admite en Amazon EKS. Actualice todos los clústeres `1.11` a la versión `1.12` o posterior para evitar la interrupción del servicio.	4 de noviembre de 2019
Ampliación de las regiones de AWS de Amazon EKS	Amazon EKS ya está disponible en la región de AWS de América del Sur (São Paulo) (`sa-east-1`).	16 de octubre de 2019
Compatibilidad con Windows	Los clústeres de Amazon EKS que ejecutan la versión `1.14` de Kubernetes ya admiten las cargas de trabajo de Windows.	7 de octubre de 2019
Escalado automático	Se ha agregado un capítulo para tratar algunos de los diferentes tipos de escalado automático de Kubernetes compatibles con los clústeres de Amazon EKS.	30 de septiembre de 2019

Actualización del panel de Kubernetes	Se ha actualizado el tema para instalar el panel de Kubernetes en clústeres de Amazon EKS para utilizar la versión beta `2.0`.	28 de septiembre de 2019
Controlador CSI de Amazon EFS	Se ha agregado el tema para instalar el controlador CSI de Amazon EFS en clústeres de Amazon EKS de Kubernetes `1.14`.	19 de septiembre de 2019
Parámetro de Amazon EC2 Systems Manager para el ID de AMI optimizada para Amazon EKS	Se ha agregado un tema para recuperar el ID de AMI optimizada para Amazon EKS mediante un parámetro de Amazon EC2 Systems Manager. El parámetro elimina la necesidad de buscar los ID de AMI.	18 de septiembre de 2019
Etiquetado de recursos de Amazon EKS	Puede administrar el etiquetado de los clústeres de Amazon EKS.	16 de septiembre de 2019
Controlador de CSI de Amazon EBS	Se ha agregado el tema para instalar el controlador CSI de Amazon EBS en clústeres de Amazon EKS de Kubernetes `1.14`.	9 de septiembre de 2019
Nueva AMI optimizada para Amazon EKS con parches para `CVE-2019-9512` y `CVE-2019-9514`	Amazon EKS ha actualizado la AMI optimizada para Amazon EKS para abordar CVE-2019-9512 y CVE-2019-9514.	6 de septiembre de 2019

Anuncio de retirada del servicio de Kubernetes `1.11` en Amazon EKS	Amazon EKS retiró el soporte para la versión `1.11` de Kubernetes el 4 de noviembre de 2019.	4 de septiembre de 2019
Versión de Kubernetes `1.14`	Se ha agregado compatibilidad con la versión `1.14` de Kubernetes para nuevas actualizaciones de versión y clústeres.	3 de septiembre de 2019
Roles de IAM para cuentas de servicio	Con los roles de IAM de las cuentas de servicio en clústeres de Amazon EKS, puede asociar un rol de IAM a una cuenta de servicio de Kubernetes. Con esta característica, ya no es necesario proporcionar permisos extendidos al rol de IAM del nodo. De esta forma, los pods de ese nodo pueden llamar a las API de AWS.	3 de septiembre de 2019
Ampliación de las regiones de AWS de Amazon EKS	Amazon EKS ya está disponible en la región de AWS de Medio Oriente (Baréin) (`me-south-1`).	29 de agosto de 2019
Actualización de la versión de la plataforma de Amazon EKS	Nuevas versiones de la plataforma como respuesta a CVE-2019-9512 y CVE-2019-9514.	28 de agosto de 2019

Actualización de la versión de la plataforma de Amazon EKS	Nuevas versiones de la plataforma para como respuesta a CVE-2019-11247 y CVE-2019-11249.	5 de agosto de 2019
Ampliación de las regiones de Amazon EKS	Amazon EKS ya está disponible en la región de AWS de Asia-Pacífico (Hong Kong) (`ap-east-1`).	31 de julio de 2019
La versión `1.10` de Kubernetes ya no se admite en Amazon EKS	La versión `1.10` de Kubernetes ya no se admite en Amazon EKS. Actualice cualquier clúster `1.10` a la versión `1.11` o superior para evitar la interrupción del servicio.	30 de julio de 2019
Se ha agregado un tema sobre el controlador de entrada de ALB	El controlador de entrada de ALB de AWS para Kubernetes desencadena la creación de un ALB cuando se crean los recursos de entrada.	11 de julio de 2019
Nueva AMI optimizada para Amazon EKS	Eliminación de `kubectl` binarios de las AMI.	3 de julio de 2019
Versión de Kubernetes `1.13`	Se ha agregado compatibilidad con la versión `1.13` de Kubernetes para nuevas actualizaciones de versión y clústeres.	18 de junio de 2019

Nueva AMI optimizada para Amazon EKS con parches para `AWS-2019-005`	Amazon EKS ha actualizado la AMI optimizada para Amazon EKS a fin de resolver los problemas de vulnerabilidad descritos en link:security/security-bulletins/AWS-2019-005/[AWS-2019-005,type="marketing"].	17 de junio de 2019
Anuncio de retirada del soporte de Kubernetes `1.10` en Amazon EKS	Amazon EKS dejó de admitir la versión `1.10` de Kubernetes el 22 de julio de 2019.	21 de mayo de 2019
Actualización de la versión de la plataforma de Amazon EKS	Nueva versión de la plataforma para los clústeres de Kubernetes `1.11` y `1.10` a fin de admitir nombres de DNS personalizados en el certificado de `kubelet` y mejorar el rendimiento de `etcd`.	21 de mayo de 2019
Introducción a `eksctl`	En esta guía de introducción se describe cómo puede instalar todos los recursos necesarios para comenzar a utilizar Amazon EKS con `eksctl`. Se trata de una utilidad sencilla de línea de comandos para crear y administrar clústeres de Kubernetes en Amazon EKS.	10 de mayo de 2019

AWS Comando `get-token` de la CLI de	El comando `aws eks get-token` se ha agregado a la CLI de AWS. Ya no necesita instalar el autenticador de AWS IAM para Kubernetes para crear tokens de seguridad del cliente para la comunicación del servidor de API del clúster. Actualice la instalación de la CLI de AWS a la versión más reciente para utilizar esta nueva funcionalidad. Para obtener más información, consulte Instalación de la Interfaz de la línea de comandos de AWS en la Guía del usuario de la interfaz de la línea de comandos de AWS.	10 de mayo de 2019
Actualización de la versión de la plataforma de Amazon EKS	Nueva versión de la plataforma para los clústeres de Kubernetes `1.12` a fin de admitir nombres de DNS personalizados en el certificado de `kubelet` y mejorar el rendimiento de `etcd`. Esto corrige un error que provocaba que los daemons de `kubelet` de nodos solicitaran un certificado nuevo cada pocos segundos.	8 de mayo de 2019
Tutorial de Prometheus	Se ha agregado un tema para implementar Prometheus en su clúster de Amazon EKS.	5 de abril de 2019

Registro de plano de control de Amazon EKS	Con esta actualización, puede obtener registros de auditoría y diagnóstico directamente desde el panel de control de Amazon EKS. Puede utilizar estos CloudWatch Logs en su cuenta como referencia para proteger y ejecutar clústeres.	4 de abril de 2019
Versión de Kubernetes `1.12`	Se ha agregado compatibilidad con la versión `1.12` de Kubernetes para nuevas actualizaciones de versión y clústeres.	28 de marzo de 2019
Se ha agregado la guía de introducción a App Mesh	Se ha agregado documentación de introducción a App Mesh y Kubernetes.	27 de marzo de 2019
Acceso privado al punto de conexión del servidor de la API de Amazon EKS	Se ha agregado documentación para deshabilitar el acceso público al punto de conexión del servidor de la API de Kubernetes del clúster de Amazon EKS.	19 de marzo de 2019
Se agregó un tema para instalar el servidor de métricas de Kubernetes	El servidor de métricas de Kubernetes es un agregador de datos de uso de recursos en el clúster.	18 de marzo de 2019

Se ha agregado la lista de proyectos de código abierto relacionados	Estos proyectos de código abierto extienden la funcionalidad de los clústeres de Kubernetes que se ejecutan en AWS, incluidos los clústeres que administra Amazon EKS.	15 de marzo de 2019
Se ha agregado un tema para instalar Helm localmente	El administrador de paquetes `helm` para Kubernetes lo ayuda a instalar y administrar aplicaciones en su clúster de Kubernetes. En este tema se muestra cómo instalar y ejecutar los `helm` y `tiller`binarios localmente. De esta forma, puede instalar y administrar gráficos mediante la CLI de Helm en su sistema local.	11 de marzo de 2019
Actualización de la versión de la plataforma de Amazon EKS	Nueva versión de la plataforma que actualiza los clústeres de Kubernetes `1.11` de Amazon EKS al nivel de parche `1.11.8` para solucionar el problema CVE-2019-1002100.	8 de marzo de 2019
Se ha aumentado el límite del clúster	Amazon EKS ha aumentado el número de clústeres que puede crear en una región de AWS de 3 a 50.	13 de febrero de 2019

Ampliación de las regiones de AWS de Amazon EKS	Amazon EKS ya está disponible en las regiones de AWS de Europa (Londres) (`eu-west-2`), Europa (París) (`eu-west-3`) y Asia-Pacífico (Bombay) (`ap-south-1`).	13 de febrero de 2019
Nueva AMI optimizada para Amazon EKS con parches para `ALAS-2019-1156`	Amazon EKS ha actualizado la AMI optimizada para Amazon EKS a fin de resolver los problemas de vulnerabilidad descritos en ALAS-2019-1156.	11 de febrero de 2019
Nueva AMI optimizada para Amazon EKS con parches para `ALAS2-2019-1141`	Amazon EKS ha actualizado la AMI optimizada para Amazon EKS a fin de abordar las CVE a las que se hace referencia en ALAS2-2019-1141.	9 de enero de 2019
Ampliación de las regiones de AWS de Amazon EKS	Amazon EKS ya está disponible en la región de AWS de Asia-Pacífico (Seúl) (`ap-northeast-2`).	9 de enero de 2019
Ampliación de las regiones de Amazon EKS	Amazon EKS ya está disponible en las siguientes regiones de AWS adicionales: Europa (Fráncfort) (`eu-central-1`), Asia-Pacífico (Tokio) (`ap-northeast-1`), Asia-Pacífico (Singapur) (`ap-southeast-1`) y Asia-Pacífico (Sídney) (`ap-southeast-2`).	19 de diciembre de 2018

Actualizaciones de clúster de Amazon EKS	Se agregó documentación para las actualizaciones de la versión de Kubernetes del clúster y sustitución de nodos de Amazon EKS.	12 de diciembre de 2018
Ampliación de las regiones de AWS de Amazon EKS	Amazon EKS ya está disponible en la región de AWS de Europa (Estocolmo) (`eu-north-1`).	11 de diciembre de 2018
Actualización de la versión de la plataforma de Amazon EKS	La nueva versión de la plataforma actualiza Kubernetes al nivel de parche `1.10.11` para abordar link:security/security-bulletins/AWS-2018-020/[CVE-2018-1002105,type="marketing"].	4 de diciembre de 2018
Se ha agregado compatibilidad con la versión `1.0.0` del controlador de entrada de ALB	El controlador de entrada de ALB publica la versión `1.0.0` con compatibilidad formal de AWS.	20 de noviembre de 2018
Se ha agregado compatibilidad con la configuración de red de CNI	La versión `1.2.1` del complemento CNI de Amazon VPC para Kubernetes ahora es compatible con la configuración de red personalizada para las interfaces de red de pod secundarias.	16 de octubre de 2018

Se ha agregado compatibilidad con `MutatingAdmissionWebhook` y `ValidatingAdmissionWebhook`	La versión `1.10-eks.2` de la plataforma de Amazon EKS ahora admite los controladores de admisión `MutatingAdmissionWebhook` y `ValidatingAdmissionWebhook`.	10 de octubre de 2018
Se ha agregado información de las AMI de socios	Canonical se ha asociado con Amazon EKS para crear AMI de nodo que puede utilizar en sus clústeres.	3 de octubre de 2018
Se han agregado instrucciones para el comando `update-kubeconfig` de la CLI de AWS	Amazon EKS ha agregado `update-kubeconfig` a la CLI de AWS para simplificar el proceso de creación de un archivo `kubeconfig` para obtener acceso al clúster.	21 de septiembre de 2018
Nuevas AMI optimizadas para Amazon EKS	Amazon EKS ha actualizado las AMI optimizadas para Amazon EKS (con y sin compatibilidad con GPU) para proporcionar varias correcciones de seguridad y optimizaciones de las AMI.	13 de septiembre de 2018
Ampliación de las regiones de AWS de Amazon EKS	Amazon EKS ahora está disponible en la región de Europa (Irlanda) (`eu-west-1`).	5 de septiembre de 2018

Actualización de la versión de la plataforma de Amazon EKS	Nueva versión de la plataforma compatible con la capa de agregación y Horizontal Pod Autoscaler (HPA) de Kubernetes.	31 de agosto de 2018
Nuevas AMI optimizadas para Amazon EKS y compatibilidad con GPU	Amazon EKS ha actualizado la AMI optimizada para Amazon EKS a fin de utilizar una plantilla de nodos de AWS CloudFormation y un script de arranque nuevos. Además, hay disponible una nueva AMI optimizada para Amazon EKS compatible con GPU.	22 de agosto de 2018
Nueva AMI optimizada para Amazon EKS con parches para `ALAS2-2018-1058`	Amazon EKS ha actualizado la AMI optimizada para Amazon EKS a fin de abordar las CVE a las que se hace referencia en ALAS2-2018-1058.	14 de agosto de 2018
Scripts de compilación de la AMI optimizada para Amazon EKS	Amazon EKS ha establecido en código abierto los scripts de compilación que se utilizan para crear la AMI optimizada para Amazon EKS. Estos scripts de compilación están ahora disponibles en GitHub.	10 de julio de 2018
Versión inicial de Amazon EKS	Documentación inicial para el lanzamiento del servicio	5 de junio de 2018

Cómo contribuir a la Guía de usuario de EKS

AWS ha lanzado una experiencia de contribución mejorada para la Guía del usuario de EKS.

Ahora puede editar directamente la fuente de la Guía del usuario de EKS en GitHub.

La documentación ahora utiliza AsciiDoc, un potente lenguaje de creación similar a Markdown. AsciiDoc combina una sintaxis simple con características empresariales de documentación, como formato avanzado, referencias cruzadas y controles de seguridad.

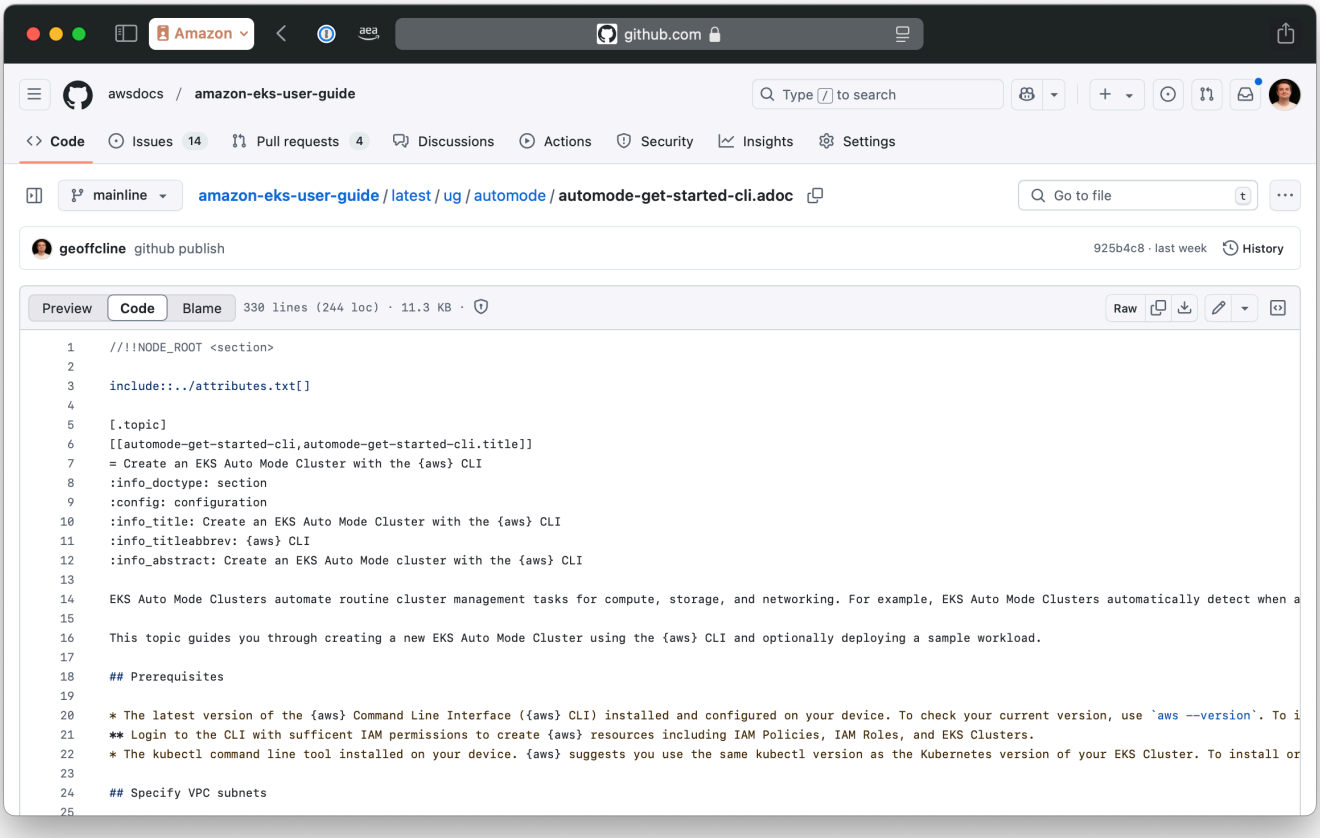
Ahora puede editar la documentación de EKS directamente en GitHub. Nuestro proceso optimizado incluye:

- Un procesamiento más rápido de solicitudes de extracción
- Menos pasos manuales
- Comprobaciones automatizadas de calidad del contenido

Esperamos contar con sus contribuciones.

Cómo editar una sola página desde un navegador web

Puede editar fácilmente una sola página de la Guía del usuario de EKS directamente desde el navegador web.



Si desea editar varias páginas desde el navegador web, consulte the section called “Edición de archivos con GitHub”.

Requisitos previos

- La página de la documentación que se va a cambiar se abre en el navegador web.
- Sesión iniciada en GitHub.

Procedimiento

1. Vaya a la página que desea editar de la Guía del usuario de Amazon EKS.
2. En el panel de la derecha, elija el enlace Edit this page on GitHub.
3. Una vez en GitHub, abra el editor de una de las siguientes maneras:
   - Pulse la tecla e del teclado.
   - Haga clic en el icono del lápiz y seleccione Editar in situ en el menú desplegable.

- Si no ve la opción de editar, debe iniciar sesión en GitHub. La cuenta de GitHub no necesita permisos especiales para sugerir cambios. Sin embargo, los colaboradores internos de Amazon deben vincular su perfil de GitHub.

4. Haga los cambios necesarios en el contenido desde el editor de GitHub.
   - El editor permite resaltar la sintaxis y obtener una vista previa de las capacidades.
   - Puede utilizar el marcado AsciiDoc para dar formato a los cambios.
   - Puede utilizar `ctrl-f` para abrir una interfaz de búsqueda/reemplazo.
5. (Opcional) Obtenga una vista previa de los cambios.
   - Utilice la pestaña `preview` para obtener una vista previa de los cambios con un formato enriquecido.
   - Utilice la opción `show diff` para resaltar las secciones modificadas. Las secciones eliminadas tienen un indicador rojo en el margen izquierdo. Las secciones nuevas tienen un indicador verde en el margen izquierdo.
6. Cuando termine de editar, haga clic en el botón Confirmar cambios… en la parte superior del editor
7. En el cuadro de diálogo de confirmación:
   - Verifique que la dirección de correo electrónico sea correcta.
   - Agregue un mensaje de confirmación breve, pero descriptivo, que explique los cambios.
   - Si es necesario, puede agregar una descripción más larga.
   - Seleccione la opción para crear una nueva ramificación y una solicitud de extracción.

Ha creado una solicitud de extracción que incluye los cambios propuestos.

Información general de la solicitud de extracción

Al crear una solicitud de extracción:

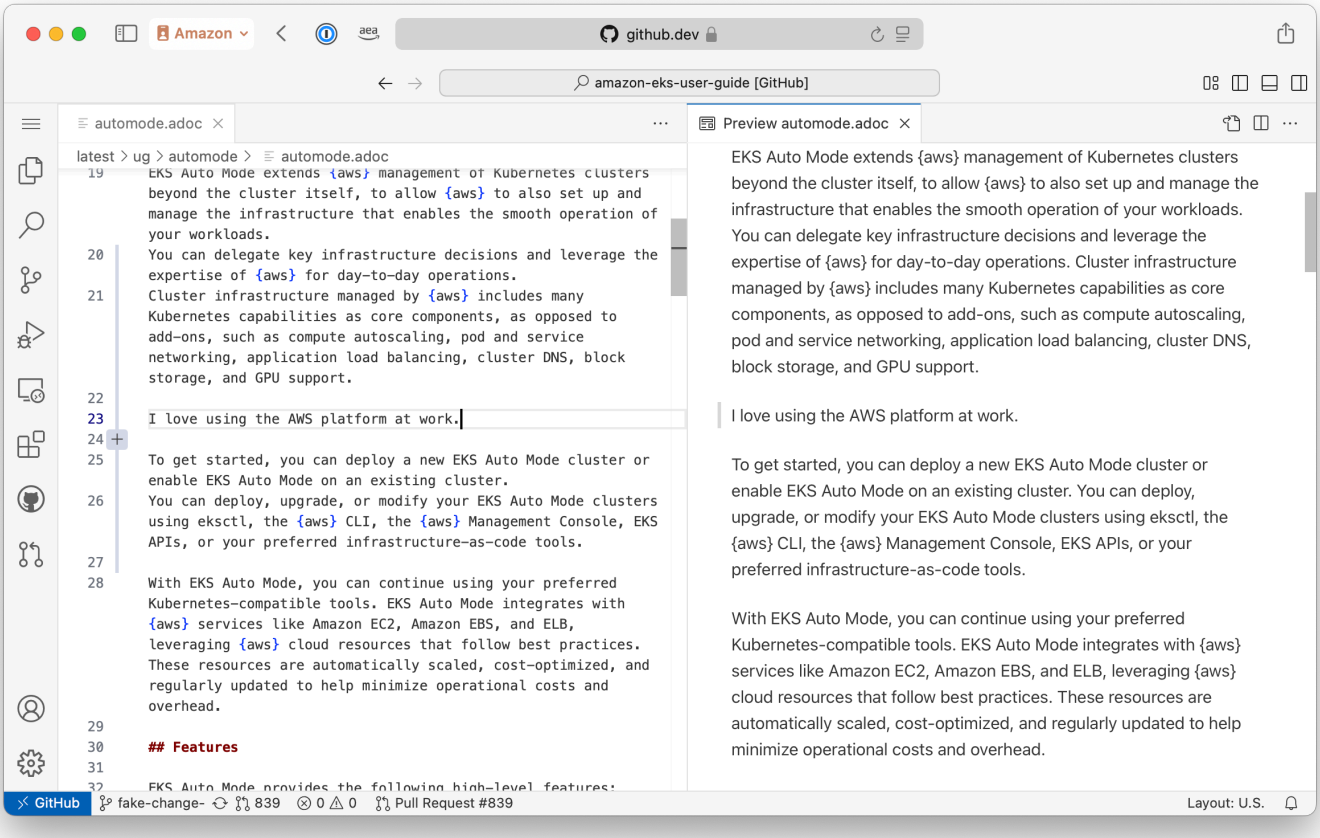
- Los cambios se someten a la revisión de quienes mantienen el repositorio.
- Los revisores pueden comentar los cambios y solicitar modificaciones.
- Se pueden ejecutar pruebas automatizadas para validar los cambios.
- Una vez aprobados, los cambios se pueden fusionar en el repositorio principal.

Las solicitudes de extracción ayudan a garantizar la calidad y permiten discutir los cambios antes de integrarlos.

Descubra cómo se revisan y aprueban las solicitudes de extracción en la documentación de GitHub.

Cómo editar varios archivos desde un navegador web con el Editor web de GitHub

Si desea proponer cambios en varias páginas o crear una nueva página de documentación, utilice el editor web GitHub.dev. Este editor web se basa en el conocido editor de texto Visual Studio Code.



Requisitos previos

- Sesión iniciada en GitHub.
- Estar familiarizado con el editor Visual Studio Code
- Estar familiarizado con Git

Procedimiento

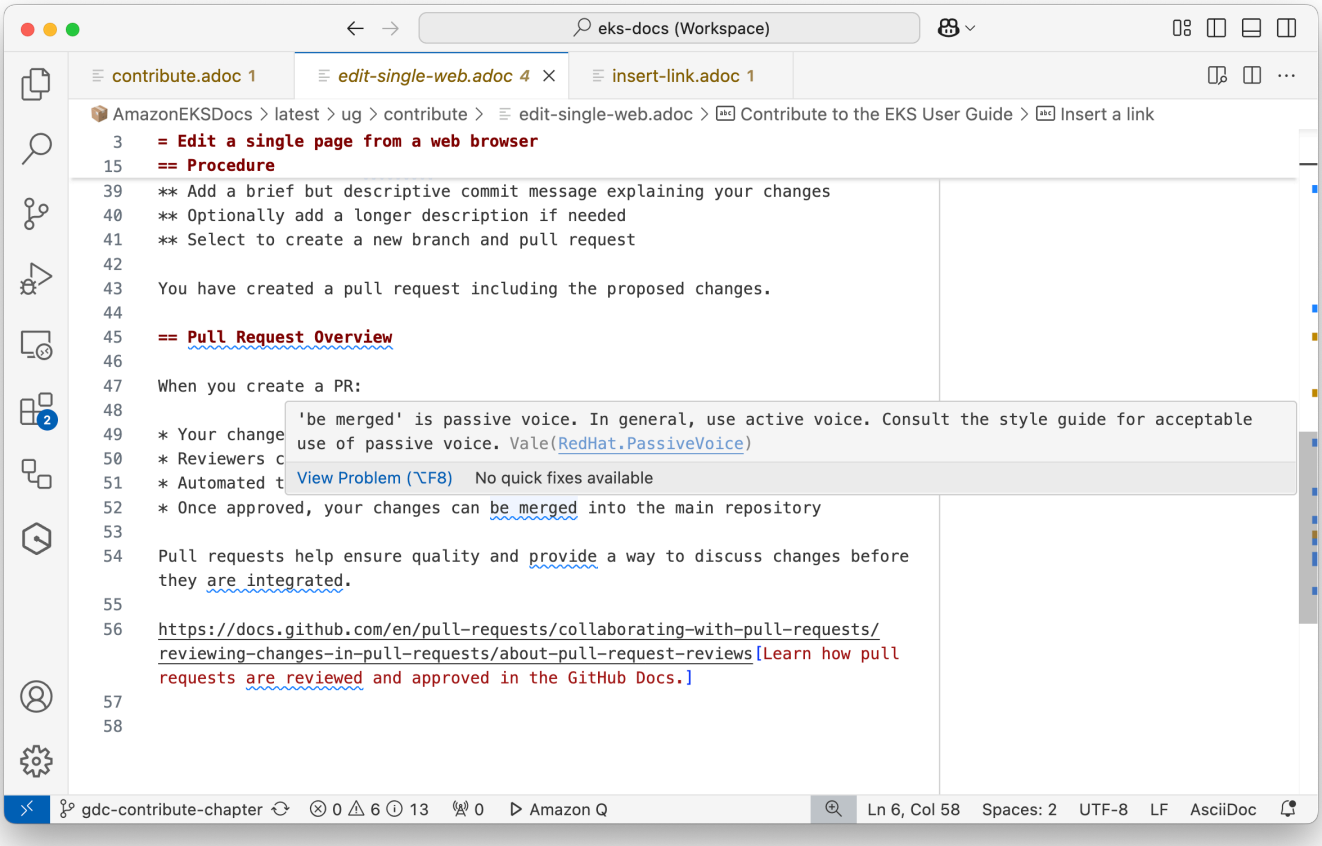
> **Note**
>
> El equipo de documentación de EKS ha creado un archivo de espacio de trabajo que incluye configuraciones sugeridas para el editor, como el ajuste de texto y el resaltado de sintaxis AsciiDoc. Sugerimos que cargue este archivo de espacio de trabajo.

1. Abra el espacio de trabajo en GitHub.dev.
   - Puede marcar la URL `https://github.dev/awsdocs/amazon-eks-user-guide/blob/mainline/eks-docs.code-workspace?workspace=true` como favorita
2. (Solo la configuración inicial) Es posible que se le pida que cree una bifurcación del repositorio en su propia cuenta de GitHub. Acepte esta petición. Para obtener más información, consulte Acerca de las bifurcaciones en la documentación de GitHub.
3. (Solo la configuración inicial) Acepte la petición que aparece en la parte inferior derecha para instalar la extensión AsciiDoc.
4. Vaya al contenido de la documentación en `latest/ug`.
   - Los archivos de la documentación se organizan según su sección de nivel superior. Por ejemplo, las páginas del capítulo “Seguridad” tienen archivos fuente en el directorio “security/”.
5. Para obtener una vista previa de una página de documentación, utilice el botón Abrir vista previa al lado, situado en la parte superior derecha. El icono incluye una pequeña lupa.
6. Usa la pestaña Control de código fuente a la izquierda para confirmar los cambios. Para obtener más información, consulte la documentación de Visual Studio Code:
   - Confirmación de cambios
   - Crear una solicitud de extracción

Después de crear una solicitud de extracción, el equipo de documentación la revisará.

Visualización de comentarios de estilo al escribir mediante la instalación local de Vale

Puede ver los comentarios de estilo a medida que escribe. Esto ayuda a identificar escritura inadecuada y errores tipográficos.



Información general:

- La CLI de Vale carga las guías de estilo y las ejecuta respecto a los archivos fuente.
- El repositorio de documentación de EKS incluye un archivo de configuración de Vale que carga guías de estilo y reglas locales.
- La extensión de Vale para Visual Studio (VS) Code muestra los comentarios de Vale en el editor.

Instalación de Vale

Siga las instrucciones que aparecen en la documentación de la CLI de Vale para la Instalación de Vale mediante un Administrador de paquetes.

Instalación de la extensión de Vale de VS Code

1. Abra VS Code.
2. Haga clic en el icono Extensiones situado en la barra de actividades (o pulse Ctrl + Shift + X).

3. Busque “Vale”.
4. Haga clic en Instalar en la extensión “Vale VSCode” proporcionada por Chris Chinchilla.
5. Vuelva a cargar VS Code cuando se le solicite.

Sincronización de Vale

Vale utiliza el archivo de configuración de `.vale.ini` en la raíz del proyecto para determinar qué reglas de estilo aplicar.

1. Abra VS Code.
2. Haga clic en Ver > Terminal (o pulse Ctrl + `).
3. Vaya al directorio raíz del proyecto si es necesario.
4. Ejecute el comando:

```
vale sync
```

5. Espere a que Vale termine de descargar y sincronizar las reglas de estilo

Visualización de comentarios de estilo en VS Code

1. Abra un archivo Markdown o AsciiDoc en VS Code.
2. La extensión Vale revisará automáticamente el texto según las reglas de estilo.
3. Los problemas de estilo se subrayarán en el editor.
4. Pase el cursor sobre el texto subrayado para ver la sugerencia de estilo específica.
5. Para solucionar los problemas, siga las sugerencias o consulte la guía de estilo.

Cómo crear una nueva página

Aprenda a crear una nueva página de documentación. En este tema se incluyen instrucciones para crear los metadatos de la página inicial y agregar la página al índice de la guía.

Cómo crear la página

1. Vaya al directorio de capítulos. Por ejemplo, si quiere crear una nueva página en la sección “Seguridad”, vaya al directorio `latest/ug/security`.

2. Determine el ID de la página. Según la convención, el identificador de la página está todo en minúsculas y segmentado con `-`. El identificador de esta página es `create-page`.
3. Cree un archivo nuevo con el ID de la página y la extensión `adoc`. Por ejemplo, `create-page.adoc`.
4. Inserte los metadatos de la página con esta plantilla:

```
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.
```

Cómo agregar la página a la navegación

1. Vaya a la página principal. La página principal de las secciones de nivel superior es `book.adoc`.
2. En la parte inferior de la página principal, incluya la página secundaria.

   ```
   include::${filename}[leveloffset=+1]
   ```

   Por ejemplo:

   ```
   include::create-page.adoc[leveloffset=+1]
   ```

Cómo insertar un enlace

AsciiDoc admite varios tipos de enlaces. Es importante utilizar el tipo de enlace correcto para que funcione correctamente en diferentes entornos.

Enlazar a una página o sección de la Guía del usuario de EKS

Utilice referencias cruzadas (xref) para enlazar páginas o secciones del mismo sitio de documentación, como la Guía del usuario de EKS. Se actualizarán automáticamente si la sección de destino se traslada o cambia de nombre.

Cómo utilizar el título de la página como texto del enlace

Para la mayoría de los casos, si vincula a otro ID en esta guía del usuario, utilice el siguiente método para que el texto del enlace se actualice automáticamente al título más reciente según sea necesario.

```
For more information, see <<page-or-section-id>>.
```

Cómo definir texto de enlace personalizado

En los casos en los que deba tener un texto de enlace personalizado, utilice el siguiente formato.

```
Here's an example of a <<page-or-section-id,link with custom text>>.
```

Enlace a otra guía de la documentación de AWS

1. Busque el enlace a la página de la documentación de AWS.
2. Elimine el prefijo `https://docs.aws.amazon.com/` y conserve solo la ruta. La ruta debe comenzar con una letra.
3. Cree un enlace como se muestra a continuación:

```
link:AmazonS3/latest/userguide/create-bucket-overview.html[Create a bucket,
 type="documentation"]
```

Enlace a una página web externa

Este formato crea un enlace estándar a una página no alojada por Amazon. Por ejemplo, use esto para los enlaces de GitHub.

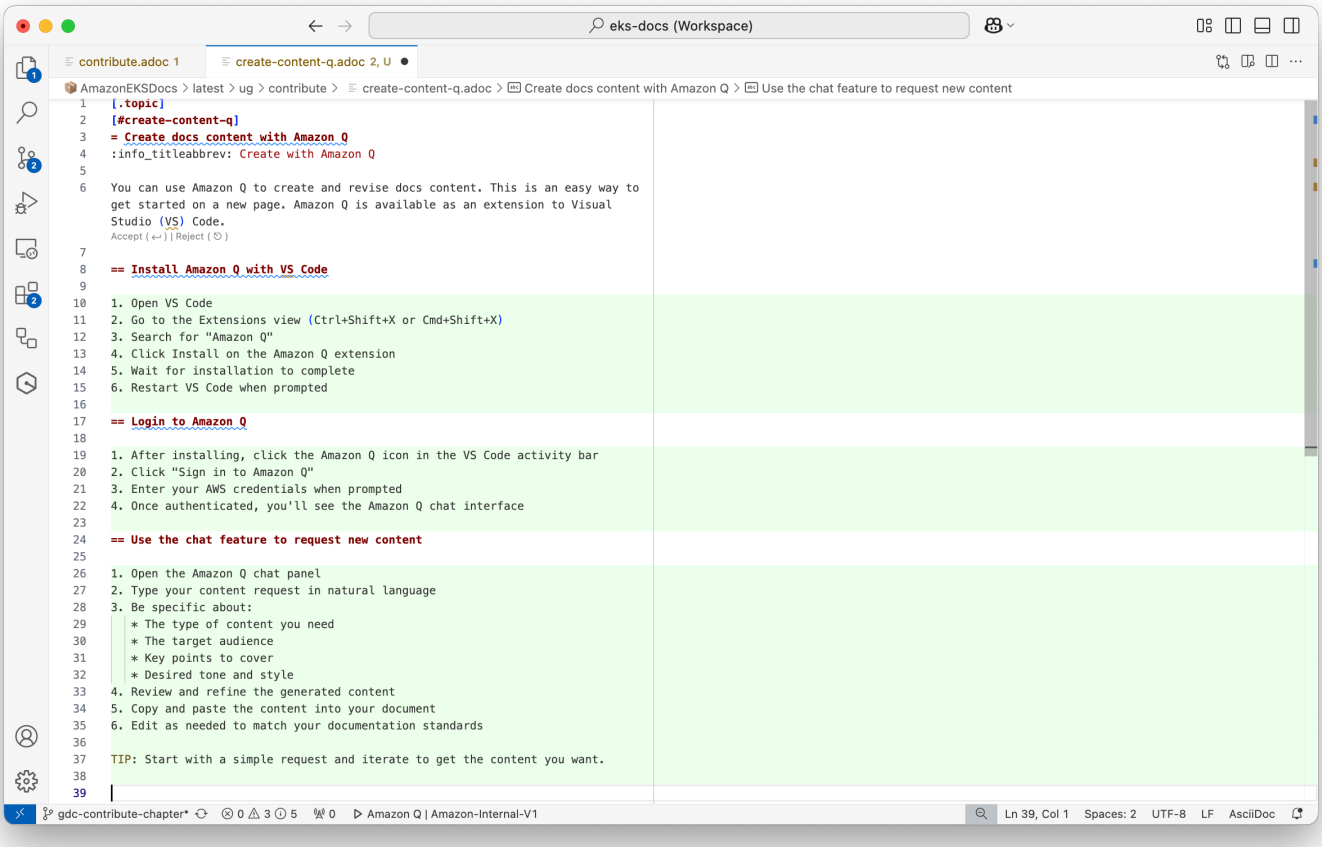
```
https://example.com[Link text]
```

> **Note**
>
> Tenemos una lista de dominios externos permitidos. La lista de permitidos está en `vale/styles/EksDocs/ExternalDomains.yml`

Crea contenido para la documentación con Amazon Q

Puede utilizar Amazon Q para crear y revisar el contenido de la documentación. Se trata de una forma sencilla de comenzar a trabajar en una nueva página. Amazon Q se encuentra disponible como extensión de Visual Studio Code.

En la siguiente imagen, Amazon Q generó las líneas marcadas en verde.



Instale Amazon Q con VS Code

1. Abra VS Code

2. Vaya a la vista Extensiones (Ctrl+Shift+X o Cmd+Shift+X)
3. Busque “Amazon Q”
4. Haga clic en Instalar en la extensión de Amazon Q
5. Espere a que se complete la instalación
6. Reinicie VS Code cuando se le solicite

Inicie sesión en Amazon Q

1. Después de la instalación, elija el icono de Amazon Q en la barra de actividad de VS Code.
2. Elija Iniciar sesión en Amazon Q.
3. Ingrese las credenciales de AWS cuando se le pida.
4. Una vez autenticado, verá la interfaz de chat de Amazon Q.

Cómo utilizar Amazon Q para crear contenido

1. Abra el archivo que desea editar en VS Code.
2. Seleccione el texto que desea revisar o la ubicación del contenido nuevo.
3. Pulse Ctrl + I o Cmd + I.
4. En la petición, especifique lo siguiente:
   - El tipo de contenido que necesita.
   - El público objetivo.
   - Los puntos clave que se van a abarcar.
   - Tono y estilo deseados.
5. Revise el contenido generado en la vista previa en línea.
6. Presione Intro para aceptar los cambios o Esc para rechazarlos.
7. Continúe con la edición según sea necesario.

Consejos

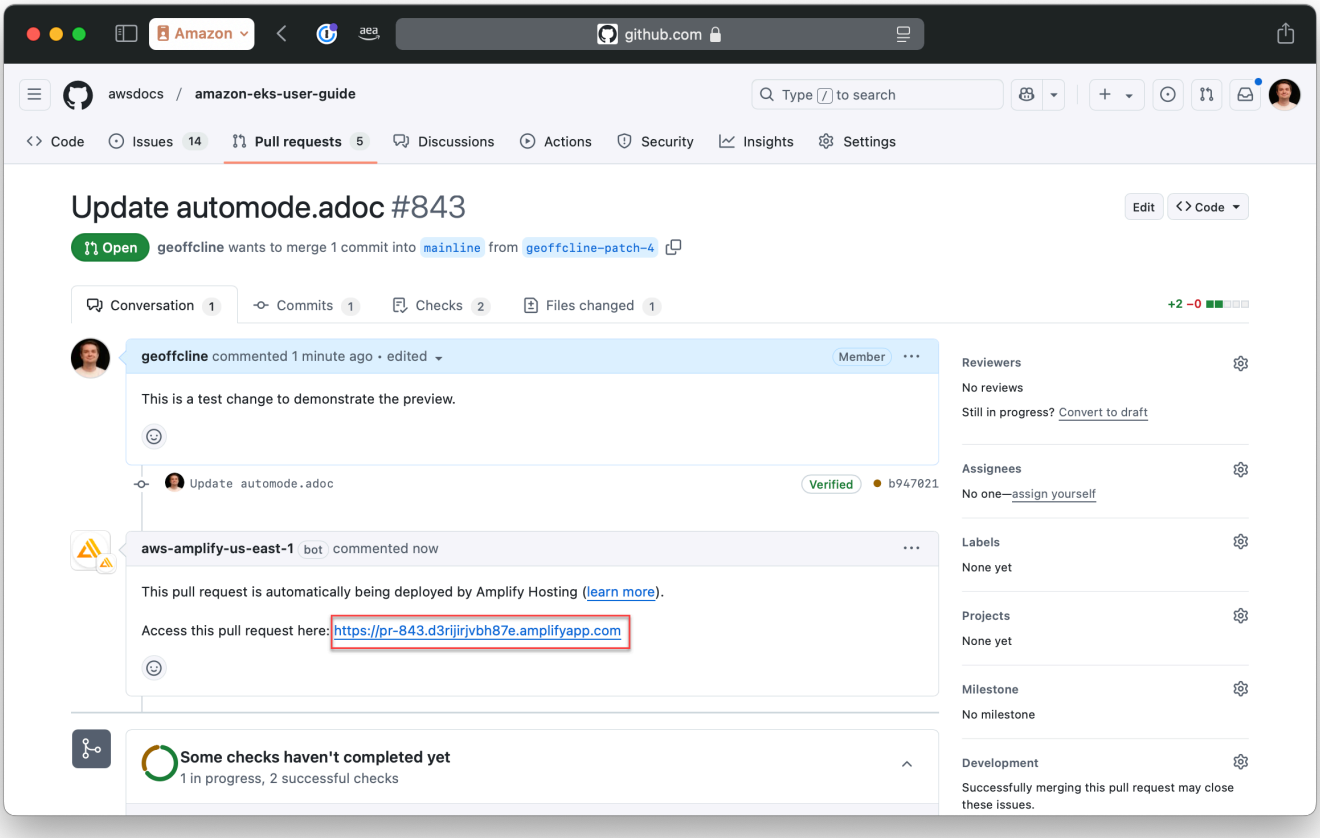
- Comience con una simple solicitud e itere para obtener el contenido que desea.
- Cree un primer borrador de los encabezados de página y pida a Q que los complete.

- Es posible que Amazon Q genere Markdown. Está bien. La herramienta AsciiDoc puede comprender la mayor parte de la sintaxis markdown.

Para obtener más información sobre Amazon Q Developer, consulte Cómo utilizar Amazon Q Developer en el IDE.

Cómo obtener una vista previa del contenido de la solicitud de extracción

La Guía del usuario de Amazon EKS de GitHub está configurada para crear y generar una vista previa del sitio de la documentación. Esta vista previa no tiene el tema completo de AWS, pero verifica que el contenido se genere correctamente y que los enlaces funcionen.



Esta vista previa está alojada en una URL temporal por AWS Amplify.

Visualización de una vista previa

Cuando envía una solicitud de extracción, AWS Amplify intenta compilar e implementar una vista previa del contenido.

Si la compilación se lleva a cabo correctamente, aws-amplify-us-east-1 comenta el enlace de vista previa en la solicitud de extracción. Elija el enlace situado a la derecha de “Access this pull request here” (como se indica en la captura de pantalla con un contorno rojo).

Si se produce un error en la compilación, los administradores del repositorio podrán ver los registros y enviar comentarios.

> **Note**
>
> Si no ha contribuido antes, es posible que un responsable del proyecto tenga que aprobar la ejecución de la compilación.

Limitaciones de la versión preliminar

La vista previa se compila como un único archivo HTML grande. Aparecerá en varias páginas cuando se publique.

Qué funciona:

- Referencias cruzadas (`xref`)
- Enlaces a Internet
- Imágenes
- Contenido alojado desde `samples/`

Qué no funciona:

- Enlaces a otro contenido de AWS con `type="documentation"`. Esto se debe a que el contenido no existe en el entorno de vista previa.
- El atributo `{aws}` no aparecerá correctamente. El valor de esto cambia en función del entorno.

Referencia de sintaxis de AsciiDoc

AsciiDoc es el lenguaje de marcado preferido para la documentación de AWS. Si bien las herramientas tienen compatibilidad parcial para la sintaxis de Markdown (como encabezados y listas), recomendamos usar la sintaxis AsciiDoc para todo el contenido a fin de garantizar la coherencia y la representación adecuada.

En esta guía se describen los elementos de sintaxis más comunes que necesitará para contribuir a la documentación de Amazon EKS. Para obtener una sintaxis más avanzada e información detallada, consulte la documentación de AsciiDoc.

Nueva página

Cada documento nuevo de AsciiDoc debe comenzar con la estructura definida en the section called "Cómo crear la página".

Encabezados

Use encabezados para organizar su contenido jerárquicamente:

```
= Page/topic title (level 1)
== Section title (level 2)
=== Level 3 heading
==== Level 4 heading
===== Level 5 heading
====== Level 6 heading
```

Nota: Siempre use la primera letra de la frase en mayúsculas en los encabezados de la documentación de AWS.

Formato de texto

Formatee el texto para resaltar la información importante:

```
Use *bold text* for UI labels.
Use _italic text_ for introducing terms or light emphasis.
Use `monospace text` for code, file names, and commands.
```

Listas

Listas sin ordenar

Cree listas con viñetas para los elementos que no tengan una secuencia específica:

```
* First item
* Second item
** Nested item
** Another nested item
* Third item
```

Listas ordenadas

Cree listas numeradas para los pasos secuenciales o los elementos priorizados:

```
. First step
. Second step
.. Substep 1
.. Substep 2
. Third step
```

Enlaces

Consulte the section called “Cómo insertar el enlace” para obtener más información sobre cómo dar formato a los enlaces correctamente. No se admiten los enlaces de estilo Markdown.

Ejemplos de código

Código en línea

Use acentos graves para el código en línea:

```
Use the `kubectl get pods` command to list all pods.
```

Bloques de código

Cree bloques de código con resaltado de sintaxis y compatibilidad con atributos (similar a las entidades):

```
[source,python,subs="verbatim,attributes"]
----
def hello_world():
    print("Hello, World!")
----
```

Imágenes

Inserte imágenes con texto alternativo para facilitar la accesibilidad:

```
image::images/image-file.png[Alt text description]
```

Tablas

Cree tablas para organizar la información:

```
[%header,cols="2"]
|===
|Header 1
|Header 2

|Cell 1,1
|Cell 1,2

|Cell 2,1
|Cell 2,2
|===
```

Para tablas más complejas, consulte la documentación de la tabla de AsciiDoc.

Avisos

Use anotaciones para resaltar la información importante y las advertencias:

```
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.
```

Vista previa:

> **Note**
>
> Esta es una anotación.

Inclusión de otros archivos

Incluya contenido de otros archivos:

```
include::filename.adoc[]
```

Atributos (similares a las entidades)

Utilice atributos predefinidos para mantener la coherencia. En particular, DEBE utilizar atributos para AWS y `arn:aws:` .

```
{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
----
```

Para obtener una lista de atributos, consulte el archivo `../attributes.txt`.

Procedimientos

Formatee procedimientos paso a paso:

```
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*.
```

```
...
```