aws

Benutzer-Leitfaden

Application Auto Scaling

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

Application Auto Scaling: Benutzer-Leitfaden

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

Die Handelsmarken und Handelsaufmachung von Amazon dürfen nicht in einer Weise in Verbindung mit nicht von Amazon stammenden Produkten oder Services verwendet werden, auf eine Art und Weise, dass Kunden irrgeführt werden könnten oder Amazon schlecht gemacht oder diskreditiert werden könnte. Alle anderen Handelsmarken, die nicht Eigentum von Amazon sind, gehören den jeweiligen Besitzern, die möglicherweise zu Amazon gehören oder nicht, mit Amazon verbunden sind oder von Amazon gesponsert werden.

Table of Contents

Was ist Application Auto Scaling?

Application Auto Scaling ist ein Webservice für Entwickler und Systemadministratoren, die eine Lösung zur automatischen Skalierung ihrer skalierbaren Ressourcen für einzelne AWS Dienste benötigen, die über Amazon EC2 Auto Scaling hinausgehen. Mit Application Auto Scaling können Sie die automatische Skalierung für die folgenden Ressourcen konfigurieren:

- WorkSpaces Flotten von Anwendungen
- Aurora-Replikate
- Amazon Comprehend-Dokumentklassifizierungs- und Entitätserkennungs-Endpunkte
- DynamoDB-Tabellen und globale sekundäre Indizes
- Amazon-ECS-Dienstleistungen
- ElastiCache Replikationsgruppen (Redis OSS und Valkey) und Memcached-Cluster
- Amazon EMR-Cluster
- Amazon Keyspaces-Tabellen (für Apache Cassandra)
- Lambda-Funktion bereitgestellte Gleichzeitigkeit
- Amazon Managed Streaming for Apache Kafka (MSK) Broker-Speicher
- Amazon Neptune-Cluster
- SageMaker Varianten von KI-Endpunkten
- SageMaker Komponenten der KI-Inferenz
- SageMaker Serverlos bereitgestellte Parallelität mit KI
- Spot-Flottenanforderungen
- Pool von Amazon WorkSpaces
- Von Ihren eigenen Anwendungen und Services bereitgestellte benutzerdefinierte Ressourcen. Weitere Informationen finden Sie im GitHubRepository.

Die regionale Verfügbarkeit der oben aufgeführten AWS Dienste finden Sie in der Regionstabelle „“.

Informationen zur Skalierung Ihrer Flotte von EC2 Amazon-Instances mithilfe von Auto Scaling Scaling-Gruppen finden Sie im Amazon EC2 Auto Scaling Scaling-Benutzerhandbuch.

Merkmale von Application Auto Scaling

Application Auto Scaling ermöglicht Ihnen die automatische Skalierung Ihrer skalierbaren Ressourcen entsprechend den von Ihnen definierten Bedingungen.

- Skalierung der Zielverfolgung — Skalieren Sie eine Ressource auf der Grundlage eines Zielwerts für eine bestimmte CloudWatch Metrik.
- Schrittweise Skalierung – Skaliert eine Ressource auf der Grundlage einer Reihe von Skalierungsanpassungen, die je nach Ausmaß der Alarmüberschreitung variieren.
- Geplante Skalierung – Skalieren Sie eine Ressource nur einmalig oder nach einem wiederkehrenden Zeitplan.
- Prädiktive Skalierung — Skalieren Sie eine Ressource proaktiv, um sie auf der Grundlage historischer Daten an die erwartete Auslastung anzupassen.

Arbeiten Sie mit Application Auto Scaling

Sie können die Skalierung mit den folgenden Schnittstellen konfigurieren, abhängig von der Ressource, die Sie skalieren:

- AWS-Managementkonsole – Stellt eine Weboberfläche bereit, mit der Sie die Skalierung konfigurieren können. Eröffnen Sie ein AWS Konto und melden Sie sich bei der an. AWS-ManagementkonsoleÖffnen Sie dann die Service-Konsole für eine der in der Einführung aufgeführten Ressourcen. Um beispielsweise eine Lambda-Funktion zu skalieren, öffnen Sie die AWS Lambda console. Stellen Sie sicher, dass Sie die Konsole in derselben Weise öffnen AWS-Region wie die Ressource, mit der Sie arbeiten möchten.

  > **Note**
  >
  > Der Konsolenzugriff ist nicht für alle Ressourcen verfügbar. Weitere Informationen finden Sie unter AWS-Services die Sie mit Application Auto Scaling verwenden können.

- AWS Command Line Interface (AWS CLI) — Stellt Befehle für eine Vielzahl von AWS-Services Befehlen bereit und wird unter Windows, MacOS und Linux unterstützt. Um zu beginnen, sehen Sie sich AWS Command Line Interface an. Eine Liste der Befehle finden Sie unter application-autoscaling in der AWS CLI Befehlsreferenz.

- AWS Tools for Windows PowerShell— Stellt Befehle für eine breite Palette von AWS Produkten für Benutzer bereit, die in der Umgebung Skripts erstellen. PowerShell Informationen zu den ersten Schritten finden Sie im AWS -Tools für PowerShell -Benutzerhandbuch. Weitere Informationen finden Sie in der AWS -Tools für PowerShell Cmdlet-Referenz.
- AWS SDKs— Stellt sprachspezifische API-Operationen bereit und kümmert sich um viele Verbindungsdetails, wie z. B. die Berechnung von Signaturen, die Behandlung von Wiederholungsversuchen von Anfragen und die Behandlung von Fehlern. Weitere Informationen finden Sie unter Tools, auf denen Sie aufbauen können. AWS
- HTTPS-API – Bietet API-Aktionen auf niedriger Ebene, die Sie mithilfe von HTTPS-Anforderungen aufrufen. Weitere Informationen finden Sie unter Aktionen in der Application Auto Scaling API-Referenz.
- CloudFormation— Unterstützt die Konfiguration der Skalierung mithilfe einer CloudFormation Vorlage. Weitere Informationen finden Sie unter Konfigurieren Sie die Auto Scaling-Ressourcen für Anwendungen mithilfe von AWS CloudFormation.

Um programmgesteuert eine Verbindung zu einem herzustellen AWS-Service, verwenden Sie einen Endpunkt. Informationen zu Endpunkten für Aufrufe von Application Auto Scaling finden Sie unter Application Auto Scaling Scaling-Endpunkte und Kontingente in den Allgemeine AWS-Referenz

Konzepte für das Application Auto Scaling

In diesem Thema werden die wichtigsten Konzepte erläutert, die Ihnen helfen, Application Auto Scaling kennenzulernen und zu nutzen.

Skalierbares Ziel

> Eine Entität, die Sie erstellen, um die Ressource anzugeben, die Sie skalieren möchten. Jedes skalierbare Ziel wird eindeutig durch einen Service-Namespace, eine Ressourcen-ID und eine skalierbare Dimension identifiziert, die eine Kapazitätsdimension des zugrunde liegenden Dienstes darstellt. Ein Amazon ECS-Service unterstützt beispielsweise die automatische Skalierung der Anzahl seiner Aufgaben, eine DynamoDB-Tabelle unterstützt das Auto Scaling der Lese- und Schreibkapazität der Tabelle und ihrer globalen sekundären Indizes, und ein Aurora-Cluster unterstützt die Skalierung der Anzahl seiner Replikate.

> **Tip**
>
> Jedes skalierbare Ziel hat auch eine minimale und maximale Kapazität. Die Skalierungsrichtlinien gehen nie über oder unter den Minimal-/Maximalbereich. Sie können direkt an der zugrunde liegenden Ressource out-of-band Änderungen vornehmen, die außerhalb dieses Bereichs liegen, von denen Application Auto Scaling nichts weiß. Wenn jedoch eine Skalierungsrichtlinie aufgerufen oder die `RegisterScalableTarget`-API aufgerufen wird, ruft Application Auto Scaling die aktuelle Kapazität ab und vergleicht sie mit der minimalen und maximalen Kapazität. Liegt sie außerhalb des Minimal- und Maximalbereichs, wird die Kapazität so aktualisiert, dass sie dem festgelegten Minimum und Maximum entspricht.

Skalieren in

Wenn Application Auto Scaling die Kapazität für ein skalierbares Ziel automatisch verringert, skaliert das skalierbare Ziel nach innen. Wenn Skalierungsrichtlinien festgelegt sind, kann das skalierbare Ziel nicht unter seiner Mindestkapazität abskaliert werden.

Horizontale Skalierung

Wenn Application Auto Scaling automatisch die Kapazität für ein skalierbares Ziel erhöht, skaliert das skalierbare Ziel nach aussen. Wenn Skalierungsrichtlinien festgelegt sind, kann das skalierbare Ziel nicht über seine maximale Kapazität aufskaliert werden.

Skalierungsrichtlinie

Eine Skalierungsrichtlinie weist Application Auto Scaling an, eine bestimmte CloudWatch Metrik zu verfolgen. Anschließend wird festgelegt, welche Skalierungsmaßnahme zu ergreifen ist, wenn die Metrik einen bestimmten Schwellenwert über- oder unterschreitet. Sie könnten beispielsweise eine Skalierung vornehmen, wenn die CPU-Auslastung in Ihrem Cluster zu steigen beginnt, und eine Skalierung vornehmen, wenn sie wieder sinkt.

Die Metriken, die für Auto Scaling verwendet werden, werden vom Ziel-Service veröffentlicht, aber Sie können auch Ihre eigene Metrik veröffentlichen CloudWatch und sie dann mit einer Skalierungsrichtlinie verwenden.

Ein Abkühlungszeitraum zwischen den Skalierungsaktivitäten ermöglicht es der Ressource, sich zu stabilisieren, bevor eine weitere Skalierungsaktivität beginnt. Application Auto Scaling wertet die Metriken während der Abkühlungsphase weiter aus. Nach Ablauf des Abkühlungszeitraums

leitet die Skalierungsrichtlinie bei Bedarf eine weitere Skalierungsaktivität ein. Wenn während des Abkühlungszeitraums aufgrund des aktuellen Metrikwerts eine größere Skalierung erforderlich ist, nimmt die Skalierungsrichtlinie sofort eine Skalierung vor.

Geplante Aktion

Geplante Aktionen skalieren Ressourcen automatisch zu einem bestimmten Datum und einer bestimmten Uhrzeit. Sie funktionieren, indem sie die minimale und maximale Kapazität für ein skalierbares Ziel ändern, und können daher verwendet werden, um nach einem Zeitplan zu skalieren, indem die minimale Kapazität hoch oder die maximale Kapazität niedrig eingestellt wird. Sie können zum Beispiel geplante Aktionen verwenden, um eine Anwendung zu skalieren, die an Wochenenden keine Ressourcen verbraucht, indem Sie die Kapazität am Freitag verringern und am darauffolgenden Montag erhöhen.

Sie können auch geplante Aktionen verwenden, um die minimalen und maximalen Werte im Laufe der Zeit zu optimieren, um sich an Situationen anzupassen, in denen ein höherer Datenverkehr als normal erwartet wird, z. B. bei Marketingkampagnen oder saisonalen Schwankungen. Auf diese Weise können Sie die Leistung in Zeiten verbessern, in denen Sie aufgrund der zunehmenden Nutzung eine höhere Skalierung vornehmen müssen, und die Kosten in Zeiten senken, in denen Sie weniger Ressourcen benötigen.

Weitere Informationen

AWS-Services die Sie mit Application Auto Scaling verwenden können — In diesem Abschnitt werden die Dienste vorgestellt, die Sie skalieren können, und Sie können das Auto Scaling einrichten, indem Sie ein skalierbares Ziel registrieren. Außerdem wird jede der mit dem IAM-Dienst verknüpften Rollen beschrieben, die Application Auto Scaling für den Zugriff auf Ressourcen im Zieldienst erstellt.

Zielverfolgungs-Skalierungsrichtlinien für Application Auto Scaling — Eine der wichtigsten Funktionen von Application Auto Scaling ist die Nachverfolgung von Skalierungsrichtlinien für das Ziel. Erfahren Sie, wie Zielverfolgungsrichtlinien automatisch die gewünschte Kapazität anpassen, um die Auslastung auf der Grundlage Ihrer konfigurierten Metrik- und Zielwerte konstant zu halten. So können Sie beispielsweise die Zielverfolgung so konfigurieren, dass die durchschnittliche CPU-Auslastung für Ihre Spot-Flotte bei 50 Prozent bleibt. Application Auto Scaling startet oder beendet dann EC2 Instances nach Bedarf, um die aggregierte CPU-Auslastung auf allen Servern bei 50 Prozent zu halten.

AWS-Services die Sie mit Application Auto Scaling verwenden können

Application Auto Scaling lässt sich in andere AWS Dienste integrieren, sodass Sie Skalierungsfunktionen hinzufügen können, um den Anforderungen Ihrer Anwendung gerecht zu werden. Die automatische Skalierung ist eine optionale Funktion des Services, die standardmäßig in fast allen Fällen deaktiviert ist.

In der folgenden Tabelle sind die AWS Dienste aufgeführt, die Sie mit Application Auto Scaling verwenden können, einschließlich Informationen zu unterstützten Methoden für die Konfiguration von Auto Scaling. Sie können Application Auto Scaling auch mit benutzerdefinierten Ressourcen verwenden.

- Konsolenzugriff - Sie können einen kompatiblen AWS Dienst zum Starten der automatischen Skalierung konfigurieren, indem Sie eine Skalierungsrichtlinie in der Konsole des Zieldienstes konfigurieren.
- CLI-Zugriff - Sie können einen kompatiblen AWS -Dienst so konfigurieren, dass die automatische Skalierung über die AWS CLI.
- SDK-Zugriff — Sie können einen kompatiblen AWS Dienst so konfigurieren, dass er Auto Scaling mit dem startet AWS SDKs.
- CloudFormation Zugriff — Sie können einen kompatiblen AWS Dienst so konfigurieren, dass er Auto Scaling mithilfe einer CloudFormation Stack-Vorlage startet. Weitere Informationen finden Sie unter Konfigurieren Sie die Auto Scaling-Ressourcen für Anwendungen mithilfe von AWS CloudFormation.

AWS Dienst	Konsolenzugriff[1]	CLI-Zugang	SDK-Zugang	CloudFormation Zugriff
WorkSpaces Anwendungen	Ja	Ja	Ja	Ja

AWS Dienst	Konsolenzugriff[1]	CLI-Zugang	SDK-Zugang	CloudFormation Zugriff
Aurora	Ja	Ja	Ja	Ja
Amazon Comprehend	Nein	Ja	Ja	Ja
Amazon-DynamoDB	Ja	Ja	Ja	Ja
Amazon ECS	Ja	Ja	Ja	Ja
Amazon ElastiCache	Ja	Ja	Ja	Ja
Amazon EMR	Ja	Ja	Ja	Ja
Amazon Keyspaces	Ja	Ja	Ja	Ja
Lambda	Nein	Ja	Ja	Ja

AWS Dienst	Konsolenzugriff[1]	CLI-Zugang	SDK-Zugang	CloudFormation Zugriff
Amazon MSK	Ja	Ja	Ja	Ja
Amazon Neptune	Nein	Ja	Ja	Ja
SageMaker AI	Ja	Ja	Ja	Ja
Spot Fleet	Ja	Ja	Ja	Ja
WorkSpaces	Ja	Ja	Ja	Ja
Benutzerdefinierte Ressourcen	Nein	Ja	Ja	Ja

[1] Konsolenzugriff zur Konfiguration von Skalierungsrichtlinien. Die meisten Dienste unterstützen die Konfiguration der geplanten Skalierung über die Konsole nicht. Derzeit bieten nur Amazon WorkSpaces Applications und Spot Fleet Konsolenzugriff für die geplante Skalierung. ElastiCache

WorkSpaces Amazon-Anwendungen und Application Auto Scaling

Sie können WorkSpaces Anwendungsflotten mithilfe von Skalierungsrichtlinien zur Zielverfolgung, schrittweisen Skalierungsrichtlinien und geplanten Skalierung skalieren.

Verwenden Sie die folgenden Informationen, um Sie bei der Integration von WorkSpaces Anwendungen in Application Auto Scaling zu unterstützen.

Für WorkSpaces Anwendungen wurde eine dienstbezogene Rolle erstellt

Die folgende dienstverknüpfte Rolle wird automatisch in Ihrem erstellt, AWS-Konto wenn Sie WorkSpaces Anwendungsressourcen als skalierbare Ziele mit Application Auto Scaling registrieren. Mit dieser Rolle kann Application Auto Scaling unterstützte Operationen innerhalb Ihres Kontos durchführen. Weitere Informationen finden Sie unter Servicegebundene Rollen für Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_AppStreamFleet`

Von der dienstgebundenen Rolle verwendeter Hauptdienst

Die im vorigen Abschnitt beschriebene dienstgebundene Rolle kann nur vom Hauptdienst übernommen werden, der durch die für die Rolle definierten vertrauenswürdigen Beziehungen autorisiert ist. Die von Application Auto Scaling verwendete dienstgebundene Rolle gewährt Zugriff auf den folgenden Hauptdienst:

- `appstream.application-autoscaling.amazonaws.com`

Registrierung von WorkSpaces Anwendungsflotten als skalierbare Ziele mit Application Auto Scaling

Application Auto Scaling erfordert ein skalierbares Ziel, bevor Sie Skalierungsrichtlinien oder geplante Aktionen für eine WorkSpaces Anwendungsflotte erstellen können. Ein skalierbares Ziel ist eine Ressource, die dank Application Auto Scaling auf- und abskaliert werden kann. Skalierbare Ziele werden eindeutig durch die Kombination von Ressourcen-ID, skalierbarer Dimension und Namespace identifiziert.

Wenn Sie Auto Scaling über die WorkSpaces Anwendungskonsole konfigurieren, registriert WorkSpaces Applications automatisch ein skalierbares Ziel für Sie.

Wenn Sie Auto Scaling über die AWS CLI oder eine der folgenden Optionen konfigurieren möchten AWS SDKs, können Sie die folgenden Optionen verwenden:

- AWS CLI:

Rufen Sie den register-scalable-targetBefehl für eine WorkSpaces Anwendungsflotte auf. Im folgenden Beispiel wird die gewünschte Kapazität einer Flotte mit dem Namen `sample-fleet` registriert, mit einer Mindestkapazität von einer Flotten-Instance und einer Höchstkapazität von fünf Flotten-Instances.

```
aws application-autoscaling register-scalable-target \
   --service-namespace appstream \
   --scalable-dimension appstream:fleet:DesiredCapacity \
   --resource-id fleet/sample-fleet \
   --min-capacity 1 \
   --max-capacity 5
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Rufen Sie den Vorgang RegisterScalableTarget auf und geben Sie `ResourceId`, `ScalableDimension`, `ServiceNamespace`, `MinCapacity`, und `MaxCapacity` als Parameter an.

Zugehörige Ressourcen

Weitere Informationen finden Sie unter Fleet Auto Scaling for Amazon WorkSpaces Applications im Amazon WorkSpaces Applications Administration Guide.

Amazon Aurora und Application Auto Scaling

Sie können Aurora-DB-Cluster mithilfe von Zielverfolgungs-Skalierungsrichtlinien, Stufenskalierungsrichtlinien und geplanter Skalierung skalieren.

Verwenden Sie die folgenden Informationen, um Aurora mit Application Auto Scaling zu integrieren.

Service-verknüpfte Rolle für Aurora erstellt

Die folgende serviceverknüpfte Rolle wird automatisch in Ihrem erstellt, AWS-Konto wenn Sie Aurora-Ressourcen als skalierbare Ziele mit Application Auto Scaling registrieren. Mit dieser Rolle kann Application Auto Scaling unterstützte Operationen innerhalb Ihres Kontos durchführen. Weitere Informationen finden Sie unter Servicegebundene Rollen für Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_RDSCluster`

Von der dienstgebundenen Rolle verwendeter Hauptdienst

Die im vorigen Abschnitt beschriebene dienstgebundene Rolle kann nur vom Hauptdienst übernommen werden, der durch die für die Rolle definierten vertrauenswürdigen Beziehungen autorisiert ist. Die von Application Auto Scaling verwendete dienstgebundene Rolle gewährt Zugriff auf den folgenden Hauptdienst:

- `rds.application-autoscaling.amazonaws.com`

Registrierung von Aurora DB-Clustern als skalierbare Ziele mit Application Auto Scaling

Application Auto Scaling erfordert ein skalierbares Ziel, bevor Sie Skalierungsrichtlinien oder geplante Aktionen für einen Aurora-Cluster erstellen können. Ein skalierbares Ziel ist eine Ressource, die Application Auto Scaling aus- und einskalieren kann. Skalierbare Ziele werden eindeutig durch die Kombination von Ressourcen-ID, skalierbarer Dimension und Namespace identifiziert.

Wenn Sie die automatische Skalierung über die Aurora-Konsole konfigurieren, registriert Aurora automatisch ein skalierbares Ziel für Sie.

Wenn Sie Auto Scaling über die AWS CLI oder eine der folgenden Optionen konfigurieren möchten AWS SDKs, können Sie die folgenden Optionen verwenden:

- AWS CLI:

  Rufen Sie den Befehl register-scalable-target für einen Aurora-Cluster auf. Im folgenden Beispiel wird die Anzahl der Aurora-Replikate in einem Cluster mit dem Namen `my-db-cluster` registriert, mit einer Mindestkapazität von einem Aurora-Replikat und einer Höchstkapazität von acht Aurora-Replikaten.

```
aws application-autoscaling register-scalable-target \
  --service-namespace rds \
  --scalable-dimension rds:cluster:ReadReplicaCount \
  --resource-id cluster:my-db-cluster \
  --min-capacity 1 \
  --max-capacity 8
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Rufen Sie den Vorgang RegisterScalableTarget auf und geben Sie `ResourceId`, `ScalableDimension`, `ServiceNamespace`, `MinCapacity`, und `MaxCapacity` als Parameter an.

Zugehörige Ressourcen

Weitere Informationen finden Sie unter Amazon Aurora Auto Scaling with Aurora Replicas im Amazon RDS-Benutzerhandbuch für Aurora.

Amazon Comprehend und Application Auto Scaling

Sie können Amazon Comprehend Dokumentenklassifizierung und Entity Recognizer Endpunkte mit Hilfe von Zielverfolgungs-Skalierungsrichtlinien und geplanter Skalierung skalieren.

Die folgenden Informationen helfen Ihnen bei der Integration von Amazon Comprehend mit Application Auto Scaling.

Service-verknüpfte Rolle für Amazon Comprehend erstellt

Die folgende serviceverknüpfte Rolle wird automatisch in Ihrem erstellt, AWS-Konto wenn Sie Amazon Comprehend Comprehend-Ressourcen als skalierbare Ziele mit Application Auto Scaling registrieren. Mit dieser Rolle kann Application Auto Scaling unterstützte Operationen innerhalb Ihres

Kontos durchführen. Weitere Informationen finden Sie unter Servicegebundene Rollen für Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_ComprehendEndpoint`

Von der dienstgebundenen Rolle verwendeter Hauptdienst

Die im vorigen Abschnitt beschriebene dienstgebundene Rolle kann nur vom Hauptdienst übernommen werden, der durch die für die Rolle definierten vertrauenswürdigen Beziehungen autorisiert ist. Die von Application Auto Scaling verwendete dienstgebundene Rolle gewährt Zugriff auf den folgenden Hauptdienst:

- `comprehend.application-autoscaling.amazonaws.com`

Registrierung von Amazon Comprehend Ressourcen als skalierbare Ziele mit Application Auto Scaling

Application Auto Scaling erfordert ein skalierbares Ziel, bevor Sie Skalierungsrichtlinien oder geplante Aktionen für einen Amazon Comprehend Document Classification oder Entity Recognizer Endpunkt erstellen können. Ein skalierbares Ziel ist eine Ressource, die dank Application Auto Scaling auf- und abskaliert werden kann. Skalierbare Ziele werden eindeutig durch die Kombination von Ressourcen-ID, skalierbarer Dimension und Namespace identifiziert.

Um Auto Scaling mit der AWS CLI oder einer der folgenden zu konfigurieren AWS SDKs, können Sie die folgenden Optionen verwenden:

- AWS CLI:

  Rufen Sie den Befehl register-scalable-target für einen Endpunkt der Dokumentenklassifizierung auf. Das folgende Beispiel registriert die gewünschte Anzahl von Inferenzeinheiten, die vom Modell für einen Dokumentenklassifikator-Endpunkt verwendet werden sollen, unter Verwendung des ARN des Endpunkts, mit einer Mindestkapazität von einer Inferenzeinheit und einer Höchstkapazität von drei Inferenzeinheiten.

```
aws application-autoscaling register-scalable-target \
  --service-namespace comprehend \
  --scalable-dimension comprehend:document-classifier-endpoint:DesiredInferenceUnits
 \
```

```
  --resource-id arn:aws:comprehend:us-west-2:123456789012:document-classifier-
endpoint/EXAMPLE \
  --min-capacity 1 \
  --max-capacity 3
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Rufen Sie den Befehl register-scalable-target für einen Entity Recognizer Endpunkt auf. Das folgende Beispiel registriert die gewünschte Anzahl von Inferenzeinheiten, die vom Modell für einen Entity Recognizer unter Verwendung der ARN des Endpunkts verwendet werden sollen, mit einer Mindestkapazität von einer Inferenzeinheit und einer Höchstkapazität von drei Inferenzeinheiten.

```
aws application-autoscaling register-scalable-target \
  --service-namespace comprehend \
  --scalable-dimension comprehend:entity-recognizer-endpoint:DesiredInferenceUnits \
  --resource-id arn:aws:comprehend:us-west-2:123456789012:entity-recognizer-
endpoint/EXAMPLE \
  --min-capacity 1 \
  --max-capacity 3
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Rufen Sie den Vorgang RegisterScalableTarget auf und geben Sie `ResourceId`, `ScalableDimension`, `ServiceNamespace`, `MinCapacity`, und `MaxCapacity` als Parameter an.

Zugehörige Ressourcen

Weitere Informationen finden Sie unter Auto Scaling with Endpoints im Amazon Comprehend Developer Guide.

Amazon DynamoDB und Application Auto Scaling

Sie können DynamoDB-Tabellen und globale sekundäre Indizes mithilfe von Zielverfolgungs-Skalierungsrichtlinien und geplanter Skalierung skalieren.

Die folgenden Informationen helfen Ihnen bei der Integration von DynamoDB mit Application Auto Scaling.

Service-verknüpfte Rolle für DynamoDB erstellt

Die folgende serviceverknüpfte Rolle wird automatisch in Ihrem erstellt, AWS-Konto wenn Sie DynamoDB-Ressourcen als skalierbare Ziele mit Application Auto Scaling registrieren. Mit dieser Rolle kann Application Auto Scaling unterstützte Operationen innerhalb Ihres Kontos durchführen. Weitere Informationen finden Sie unter Servicegebundene Rollen für Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_DynamoDBTable`

Von der dienstgebundenen Rolle verwendeter Hauptdienst

Die im vorigen Abschnitt beschriebene dienstgebundene Rolle kann nur vom Hauptdienst übernommen werden, der durch die für die Rolle definierten vertrauenswürdigen Beziehungen autorisiert ist. Die von Application Auto Scaling verwendete dienstgebundene Rolle gewährt Zugriff auf den folgenden Hauptdienst:

- `dynamodb.application-autoscaling.amazonaws.com`

Registrieren von DynamoDB-Ressourcen als skalierbare Ziele mit Application Auto Scaling

Application Auto Scaling erfordert ein skalierbares Ziel, bevor Sie Skalierungsrichtlinien oder geplante Aktionen für eine DynamoDB-Tabelle oder einen globalen sekundären Index erstellen können. Ein skalierbares Ziel ist eine Ressource, die dank Application Auto Scaling auf- und abskaliert werden

kann. Skalierbare Ziele werden eindeutig durch die Kombination von Ressourcen-ID, skalierbarer Dimension und Namespace identifiziert.

Wenn Sie die automatische Skalierung über die DynamoDB-Konsole konfigurieren, registriert DynamoDB automatisch ein skalierbares Ziel für Sie.

Wenn Sie Auto Scaling über die AWS CLI oder eine der folgenden Optionen konfigurieren möchten AWS SDKs, können Sie die folgenden Optionen verwenden:

- AWS CLI:

  Rufen Sie den register-scalable-targetBefehl für die Schreibkapazität einer Tabelle auf. Im folgenden Beispiel wird die bereitgestellte Schreibkapazität einer `my-table` aufgerufenen Tabelle mit einer Mindestkapazität von fünf Schreibkapazitätseinheiten und einer maximalen Kapazität von 10 Schreibkapazitätseinheiten registriert:

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace dynamodb \
    --scalable-dimension dynamodb:table:WriteCapacityUnits \
    --resource-id table/my-table \
    --min-capacity 5 \
    --max-capacity 10
  ```

  Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück:

  ```
  {
      "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
  target/1234abcd56ab78cd901ef1234567890ab123"
  }
  ```

  Rufen Sie den register-scalable-targetBefehl für die Lesekapazität einer Tabelle auf. Im folgenden Beispiel wird die bereitgestellte Lesekapazität einer `my-table` aufgerufenen Tabelle mit einer Mindestkapazität von fünf Lesekapazitätseinheiten und einer maximalen Kapazität von 10 Leseeinheiten registriert:

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace dynamodb \
    --scalable-dimension dynamodb:table:ReadCapacityUnits \
    --resource-id table/my-table \
    --min-capacity 5 \
  ```

```
  --max-capacity 10
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück:

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Rufen Sie den register-scalable-targetBefehl für die Schreibkapazität eines globalen sekundären Indexes auf. Das folgende Beispiel registriert die bereitgestellte Schreibkapazität eines globalen sekundären Indexes namens`my-table-index`, mit einer Mindestkapazität von fünf Schreibkapazitätseinheiten und einer maximalen Kapazität von 10 Schreibkapazitätseinheiten:

```
aws application-autoscaling register-scalable-target \
  --service-namespace dynamodb \
  --scalable-dimension dynamodb:index:WriteCapacityUnits \
  --resource-id table/my-table/index/my-table-index \
  --min-capacity 5 \
  --max-capacity 10
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück:

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Rufen Sie den register-scalable-targetBefehl für die Lesekapazität eines globalen sekundären Indexes auf. Das folgende Beispiel registriert die bereitgestellte Lesekapazität eines globalen sekundären Indexes namens`my-table-index`, mit einer Mindestkapazität von fünf Lesekapazitätseinheiten und einer maximalen Kapazität von 10 Lesekapazitätseinheiten:

```
aws application-autoscaling register-scalable-target \
  --service-namespace dynamodb \
  --scalable-dimension dynamodb:index:ReadCapacityUnits \
  --resource-id table/my-table/index/my-table-index \
  --min-capacity 5 \
  --max-capacity 10
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück:

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Rufen Sie den Vorgang RegisterScalableTarget auf und geben Sie `ResourceId`, `ScalableDimension`, `ServiceNamespace`, `MinCapacity`, und `MaxCapacity` als Parameter an.

Zugehörige Ressourcen

Wenn Sie gerade erst mit Application Auto Scaling beginnen, finden Sie in der folgenden Dokumentation weitere nützliche Informationen zur Skalierung Ihrer DynamoDB-Ressourcen:

- Verwaltung der Durchsatzkapazität mit DynamoDB Auto Scaling im Amazon DynamoDB Developer Leitfaden
- Evaluieren Sie die Auto Scaling-Einstellungen Ihrer Tabelle im Amazon DynamoDB Developer Guide
- So konfigurieren Sie CloudFormation Auto Scaling für DynamoDB-Tabellen und -Indizes im Blog AWS

Amazon ECS und Application Auto Scaling

Sie können ECS-Services mithilfe von Zielverfolgungs-Skalierungsrichtlinien, prädiktiven Skalierungsrichtlinien, schrittweisen Skalierungsrichtlinien und geplanten Skalierungsrichtlinien skalieren.

Die folgenden Informationen helfen Ihnen bei der Integration von Amazon ECS mit Application Auto Scaling.

Serviceverknüpfte Rolle für Amazon ECS erstellt

Die folgende serviceverknüpfte Rolle wird automatisch in Ihrem erstellt, AWS-Konto wenn Sie Amazon ECS-Ressourcen als skalierbare Ziele mit Application Auto Scaling registrieren. Mit dieser

Rolle kann Application Auto Scaling unterstützte Operationen innerhalb Ihres Kontos durchführen. Weitere Informationen finden Sie unter Servicegebundene Rollen für Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_ECSService`

Von der dienstgebundenen Rolle verwendeter Hauptdienst

Die im vorigen Abschnitt beschriebene dienstgebundene Rolle kann nur vom Hauptdienst übernommen werden, der durch die für die Rolle definierten vertrauenswürdigen Beziehungen autorisiert ist. Die von Application Auto Scaling verwendete dienstgebundene Rolle gewährt Zugriff auf den folgenden Hauptdienst:

- `ecs.application-autoscaling.amazonaws.com`

Registrierung von ECS-Diensten als skalierbare Ziele mit Application Auto Scaling

Application Auto Scaling erfordert ein skalierbares Ziel, bevor Sie Skalierungsrichtlinien oder geplante Aktionen für einen Amazon ECS-Service erstellen können. Ein skalierbares Ziel ist eine Ressource, die dank Application Auto Scaling auf- und abskaliert werden kann. Skalierbare Ziele werden eindeutig durch die Kombination von Ressourcen-ID, skalierbarer Dimension und Namespace identifiziert.

Wenn Sie die automatische Skalierung über die Amazon ECS-Konsole konfigurieren, dann registriert Amazon ECS automatisch ein skalierbares Ziel für Sie.

Wenn Sie Auto Scaling über die AWS CLI oder eine der folgenden Optionen konfigurieren möchten AWS SDKs, können Sie die folgenden Optionen verwenden:

- AWS CLI:

  Rufen Sie den Befehl register-scalable-target für einen Amazon ECS-Service auf. Das folgende Beispiel registriert ein skalierbares Ziel für einen Service namens `sample-app-service`, der auf dem `default` Cluster läuft, mit einer minimalen Taskanzahl von einem Task und einer maximalen Taskanzahl von 10 Tasks.

```
aws application-autoscaling register-scalable-target \
  --service-namespace ecs \
```

```
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/default/sample-app-service \
  --min-capacity 1 \
  --max-capacity 10
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Rufen Sie den Vorgang RegisterScalableTarget auf und geben Sie `ResourceId`, `ScalableDimension`, `ServiceNamespace`, `MinCapacity`, und `MaxCapacity` als Parameter an.

Zugehörige Ressourcen

Wenn Sie gerade erst mit Application Auto Scaling beginnen, finden Sie in der folgenden Dokumentation weitere nützliche Informationen zur Skalierung Ihrer Amazon ECS-Ressourcen:

- Service Auto Scaling im Amazon Elastic Container Service Developer Guide
- Optimieren Sie die auto Skalierung von Amazon ECS-Service im Amazon Elastic Container Service Developer Guide

> **Note**
>
> Anweisungen zum Aussetzen von Scale-Out-Prozessen während laufender Amazon ECS-Bereitstellungen finden Sie in der folgenden Dokumentation:
> auto Skalierung und Bereitstellungen von Services im Amazon Elastic Container Service Developer Guide

ElastiCache und Application Auto Scaling

Sie können ElastiCache Amazon-Replikationsgruppen (Redis OSS und Valkey) und selbst entworfene Memcached-Cluster mithilfe von Zielverfolgungs-Skalierungsrichtlinien und geplanter Skalierung horizontal skalieren.

Verwenden Sie für die Integration ElastiCache mit Application Auto Scaling die folgenden Informationen.

Eine mit dem Dienst verknüpfte Rolle wurde erstellt für ElastiCache

Die folgende serviceverknüpfte Rolle wird automatisch in Ihrem erstellt, AWS-Konto wenn Sie ElastiCache Ressourcen als skalierbare Ziele mit Application Auto Scaling registrieren. Mit dieser Rolle kann Application Auto Scaling unterstützte Operationen innerhalb Ihres Kontos durchführen. Weitere Informationen finden Sie unter Servicegebundene Rollen für Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_ElastiCacheRG`

Von der dienstgebundenen Rolle verwendeter Hauptdienst

Die im vorigen Abschnitt beschriebene dienstgebundene Rolle kann nur vom Hauptdienst übernommen werden, der durch die für die Rolle definierten vertrauenswürdigen Beziehungen autorisiert ist. Die von Application Auto Scaling verwendete dienstgebundene Rolle gewährt Zugriff auf den folgenden Hauptdienst:

- `elasticache.application-autoscaling.amazonaws.com`

Registrierung von ElastiCache Ressourcen als skalierbare Ziele mit Application Auto Scaling

Application Auto Scaling erfordert ein skalierbares Ziel, bevor Sie Skalierungsrichtlinien oder geplante Aktionen für eine ElastiCache Replikationsgruppe, einen Cluster oder einen Knoten erstellen können. Ein skalierbares Ziel ist eine Ressource, die dank Application Auto Scaling auf- und abskaliert werden kann. Skalierbare Ziele werden eindeutig durch die Kombination von Ressourcen-ID, skalierbarer Dimension und Namespace identifiziert.

Wenn Sie Auto Scaling über die ElastiCache Konsole konfigurieren, wird ElastiCache automatisch ein skalierbares Ziel für Sie registriert.

Wenn Sie Auto Scaling über die AWS CLI oder eine der folgenden Optionen konfigurieren möchten AWS SDKs, können Sie die folgenden Optionen verwenden:

- AWS CLI:

  Rufen Sie den register-scalable-targetBefehl für eine ElastiCache Replikationsgruppe auf. Im folgenden Beispiel wird die gewünschte Anzahl von Knotengruppen für eine Replikationsgruppe mit dem Namen `mycluster1` registriert, mit einer Mindestkapazität von einem und einer Höchstkapazität von fünf.

```
aws application-autoscaling register-scalable-target \
   --service-namespace elasticache \
   --scalable-dimension elasticache:replication-group:NodeGroups \
   --resource-id replication-group/mycluster1 \
   --min-capacity 1 \
   --max-capacity 5
```

  Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

  Im folgenden Beispiel wird die gewünschte Anzahl von Replikaten pro Knotengruppe für eine `mycluster2` aufgerufene Replikationsgruppe mit einer Mindestkapazität von einem und einer Höchstkapazität von fünf registriert.

```
aws application-autoscaling register-scalable-target \
   --service-namespace elasticache \
   --scalable-dimension elasticache:replication-group:Replicas \
   --resource-id replication-group/mycluster2 \
   --min-capacity 1 \
   --max-capacity 5
```

  Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/234abcd56ab78cd901ef1234567890ab1234"
```

```
}
```

Im folgenden Beispiel wird die gewünschte Anzahl von Knoten für einen aufgerufenen `mynode1` Cluster mit einer Mindestkapazität von 20 und einer Höchstkapazität von 50 registriert.

```
aws application-autoscaling register-scalable-target \
   --service-namespace elasticache \
   --scalable-dimension elasticache:cache-cluster:Nodes \
   --resource-id cache-cluster/mynode1 \
   --min-capacity 20 \
   --max-capacity 50
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/01234abcd56ab78cd901ef1234567890ab12"
}
```

- AWS SDK:

  Rufen Sie den Vorgang RegisterScalableTarget auf und geben Sie `ResourceId`, `ScalableDimension`, `ServiceNamespace`, `MinCapacity`, und `MaxCapacity` als Parameter an.

Zugehörige Ressourcen

Weitere Informationen finden Sie unter Auto Scaling Valkey- und Redis OSS-Cluster und Scaling-Cluster für Memcached im Amazon-Benutzerhandbuch. ElastiCache

Amazon Keyspaces (für Apache Cassandra) und Application Auto Scaling

Sie können Amazon Keyspaces-Tabellen mithilfe von Zielverfolgungs-Skalierungsrichtlinien und geplanter Skalierung skalieren.

Die folgenden Informationen helfen Ihnen bei der Integration von Amazon Keyspaces mit Application Auto Scaling.

Service-verknüpfte Rolle für Amazon Keyspaces erstellt

Die folgende serviceverknüpfte Rolle wird automatisch in Ihrem erstellt, AWS-Konto wenn Sie Amazon Keyspaces-Ressourcen als skalierbare Ziele mit Application Auto Scaling registrieren. Mit dieser Rolle kann Application Auto Scaling unterstützte Operationen innerhalb Ihres Kontos durchführen. Weitere Informationen finden Sie unter Servicegebundene Rollen für Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_CassandraTable`

Von der dienstgebundenen Rolle verwendeter Hauptdienst

Die im vorigen Abschnitt beschriebene dienstgebundene Rolle kann nur vom Hauptdienst übernommen werden, der durch die für die Rolle definierten vertrauenswürdigen Beziehungen autorisiert ist. Die von Application Auto Scaling verwendete dienstgebundene Rolle gewährt Zugriff auf den folgenden Hauptdienst:

- `cassandra.application-autoscaling.amazonaws.com`

Registrierung von Amazon Keyspaces-Tabellen als skalierbare Ziele mit Application Auto Scaling

Application Auto Scaling erfordert ein skalierbares Ziel, bevor Sie Skalierungsrichtlinien oder geplante Aktionen für eine Amazon Keyspaces-Tabelle erstellen können. Ein skalierbares Ziel ist eine Ressource, die dank Application Auto Scaling auf- und abskaliert werden kann. Skalierbare Ziele werden eindeutig durch die Kombination von Ressourcen-ID, skalierbarer Dimension und Namespace identifiziert.

Wenn Sie die automatische Skalierung über die Amazon Keyspaces-Konsole konfigurieren, dann registriert Amazon Keyspaces automatisch ein skalierbares Ziel für Sie.

Wenn Sie Auto Scaling über die AWS CLI oder eine der folgenden Optionen konfigurieren möchten AWS SDKs, können Sie die folgenden Optionen verwenden:

- AWS CLI:

  Rufen Sie den register-scalable-target Befehl für eine Amazon Keyspaces-Tabelle auf. Das folgende Beispiel registriert die bereitgestellte Schreibkapazität einer Tabelle mit dem

Namen `mytable`, mit einer Mindestkapazität von fünf Schreibkapazitätseinheiten und einer Höchstkapazität von 10 Schreibkapazitätseinheiten.

```
aws application-autoscaling register-scalable-target \
  --service-namespace cassandra \
  --scalable-dimension cassandra:table:WriteCapacityUnits \
  --resource-id keyspace/mykeyspace/table/mytable \
  --min-capacity 5 \
  --max-capacity 10
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Das folgende Beispiel registriert die bereitgestellte Lesekapazität einer Tabelle mit dem Namen `mytable`, mit einer Mindestkapazität von fünf Lesekapazitätseinheiten und einer Höchstkapazität von 10 Lesekapazitätseinheiten.

```
aws application-autoscaling register-scalable-target \
  --service-namespace cassandra \
  --scalable-dimension cassandra:table:ReadCapacityUnits \
  --resource-id keyspace/mykeyspace/table/mytable \
  --min-capacity 5 \
  --max-capacity 10
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Rufen Sie den Vorgang RegisterScalableTarget auf und geben Sie `ResourceId`, `ScalableDimension`, `ServiceNamespace`, `MinCapacity`, und `MaxCapacity` als Parameter an.

Zugehörige Ressourcen

Weitere Informationen finden Sie unter Automatisches Verwalten der Durchsatzkapazität mit Amazon Keyspaces Auto Scaling im Amazon Keyspaces Developer Guide.

AWS Lambda und Application Auto Scaling

Sie können die AWS Lambda bereitgestellte Parallelität mithilfe von Skalierungsrichtlinien für die Zielverfolgung und der geplanten Skalierung skalieren.

Die folgenden Informationen helfen Ihnen bei der Integration von Lambda mit Application Auto Scaling.

Dienstverknüpfte Rolle für Lambda erstellt

Die folgende serviceverknüpfte Rolle wird automatisch in Ihrem erstellt, AWS-Konto wenn Sie Lambda-Ressourcen als skalierbare Ziele mit Application Auto Scaling registrieren. Mit dieser Rolle kann Application Auto Scaling unterstützte Operationen innerhalb Ihres Kontos durchführen. Weitere Informationen finden Sie unter Servicegebundene Rollen für Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_LambdaConcurrency`

Von der dienstgebundenen Rolle verwendeter Hauptdienst

Die im vorigen Abschnitt beschriebene dienstgebundene Rolle kann nur vom Hauptdienst übernommen werden, der durch die für die Rolle definierten vertrauenswürdigen Beziehungen autorisiert ist. Die von Application Auto Scaling verwendete dienstgebundene Rolle gewährt Zugriff auf den folgenden Hauptdienst:

- `lambda.application-autoscaling.amazonaws.com`

Registrieren von Lambda-Funktionen als skalierbare Ziele mit Application Auto Scaling

Application Auto Scaling erfordert ein skalierbares Ziel, bevor Sie Skalierungsrichtlinien oder geplante Aktionen für eine Lambda-Funktion erstellen können. Ein skalierbares Ziel ist eine Ressource, die

dank Application Auto Scaling auf- und abskaliert werden kann. Skalierbare Ziele werden eindeutig durch die Kombination von Ressourcen-ID, skalierbarer Dimension und Namespace identifiziert.

Um Auto Scaling mit der AWS CLI oder einer der folgenden zu konfigurieren AWS SDKs, können Sie die folgenden Optionen verwenden:

- AWS CLI:

  Rufen Sie den register-scalable-target-Befehl für eine Lambda-Funktion auf. Im folgenden Beispiel wird die bereitgestellte Gleichzeitigkeit für einen Alias mit dem Namen `BLUE` für eine Funktion mit dem Namen `my-function` mit einer Mindestkapazität von 0 und einer Höchstkapazität von 100 registriert.

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace lambda \
    --scalable-dimension lambda:function:ProvisionedConcurrency \
    --resource-id function:my-function:BLUE \
    --min-capacity 0 \
    --max-capacity 100
  ```

  Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

  ```
  {
      "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
  }
  ```

- AWS SDK:

  Rufen Sie den Vorgang RegisterScalableTarget auf und geben Sie `ResourceId`, `ScalableDimension`, `ServiceNamespace`, `MinCapacity`, und `MaxCapacity` als Parameter an.

Zugehörige Ressourcen

Wenn Sie gerade erst mit Application Auto Scaling beginnen, finden Sie in der folgenden Dokumentation weitere nützliche Informationen zur Skalierung Ihrer Lambda-Funktionen:

- Konfiguration der bereitgestellten Parallelität finden Sie im Entwicklerhandbuch AWS Lambda
- Planung von Lambda Provisioned Concurrency für wiederkehrende Spitzennutzung im Blog AWS

Amazon Managed Streaming for Apache Kafka (MSK) und Application Auto Scaling

Sie können den Amazon MSK-Clusterspeicher mithilfe von Zielverfolgungs-Skalierungsrichtlinien skalieren. Gibt an, ob die Herunterskalierung durch die Richtlinie für die Ziel-Nachverfolgung deaktiviert ist.

Die folgenden Informationen helfen Ihnen bei der Integration von Amazon MSK mit Application Auto Scaling.

Service-gebundene Rolle für Amazon MSK erstellt

Die folgende serviceverknüpfte Rolle wird automatisch in Ihrem erstellt, AWS-Konto wenn Sie Amazon MSK-Ressourcen als skalierbare Ziele mit Application Auto Scaling registrieren. Mit dieser Rolle kann Application Auto Scaling unterstützte Operationen innerhalb Ihres Kontos durchführen. Weitere Informationen finden Sie unter Servicegebundene Rollen für Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_KafkaCluster`

Von der dienstgebundenen Rolle verwendeter Hauptdienst

Die im vorigen Abschnitt beschriebene dienstgebundene Rolle kann nur vom Hauptdienst übernommen werden, der durch die für die Rolle definierten vertrauenswürdigen Beziehungen autorisiert ist. Die von Application Auto Scaling verwendete dienstgebundene Rolle gewährt Zugriff auf den folgenden Hauptdienst:

- `kafka.application-autoscaling.amazonaws.com`

Registrierung von Amazon MSK-Cluster-Speicher als skalierbare Ziele mit Application Auto Scaling

Application Auto Scaling erfordert ein skalierbares Ziel, bevor Sie eine Skalierungsrichtlinie für die Größe des Speichervolumens pro Broker eines Amazon MSK Clusters erstellen können. Ein skalierbares Ziel ist eine Ressource, die Application Auto Scaling aufskalieren oder abskalieren kann. Skalierbare Ziele werden eindeutig durch die Kombination von Ressourcen-ID, skalierbarer Dimension und Namespace identifiziert.

Wenn Sie die automatische Skalierung über die Amazon MSK-Konsole konfigurieren, dann registriert Amazon MSK automatisch ein skalierbares Ziel für Sie.

Wenn Sie Auto Scaling über die AWS CLI oder eine der folgenden Optionen konfigurieren möchten AWS SDKs, können Sie die folgenden Optionen verwenden:

- AWS CLI:

  Rufen Sie den Befehl register-scalable-target für einen Amazon MSK-Cluster auf. Das folgende Beispiel registriert die Größe des Speichervolumens pro Broker eines Amazon MSK Clusters mit einer Mindestkapazität von 100 GiB und einer Höchstkapazität von 800 GiB.

  ```
  aws application-autoscaling register-scalable-target \
     --service-namespace kafka \
     --scalable-dimension kafka:broker-storage:VolumeSize \
     --resource-id arn:aws:kafka:us-east-1:123456789012:cluster/demo-
  cluster-1/6357e0b2-0e6a-4b86-a0b4-70df934c2e31-5 \
     --min-capacity 100 \
     --max-capacity 800
  ```

  Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

  ```
  {
      "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
  target/1234abcd56ab78cd901ef1234567890ab123"
  }
  ```

- AWS SDK:

  Rufen Sie den Vorgang RegisterScalableTarget auf und geben Sie `ResourceId`, `ScalableDimension`, `ServiceNamespace`, `MinCapacity`, und `MaxCapacity` als Parameter an.

> **Note**
>
> Wenn ein Amazon MSK-Cluster das skalierbare Ziel ist, ist scale in deaktiviert und kann nicht aktiviert werden.

Zugehörige Ressourcen

Weitere Informationen finden Sie unter Automatische Skalierung für Amazon MSK-Cluster im Amazon Managed Streaming for Apache Kafka Developer Guide.

Amazon Neptune und Application Auto Scaling

Sie können Neptune-Funktionen mithilfe von Zielverfolgungs-Skalierungsrichtlinien und geplanter Skalierung skalieren.

Die folgenden Informationen helfen Ihnen bei der Integration von Neptune mit Application Auto Scaling.

Serviceverknüpfte Rolle für Neptune erstellt

Die folgende serviceverknüpfte Rolle wird automatisch in Ihrem erstellt, AWS-Konto wenn Sie Neptune-Ressourcen als skalierbare Ziele mit Application Auto Scaling registrieren. Mit dieser Rolle kann Application Auto Scaling unterstützte Operationen innerhalb Ihres Kontos durchführen. Weitere Informationen finden Sie unter Servicegebundene Rollen für Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_NeptuneCluster`

Von der dienstgebundenen Rolle verwendeter Hauptdienst

Die im vorigen Abschnitt beschriebene dienstgebundene Rolle kann nur vom Hauptdienst übernommen werden, der durch die für die Rolle definierten vertrauenswürdigen Beziehungen autorisiert ist. Die von Application Auto Scaling verwendete dienstgebundene Rolle gewährt Zugriff auf den folgenden Hauptdienst:

- `neptune.application-autoscaling.amazonaws.com`

Registrierung von Neptune-Clustern als skalierbare Ziele mit Application Auto Scaling

Application Auto Scaling erfordert ein skalierbares Ziel, bevor Sie Skalierungsrichtlinien oder geplante Aktionen für einen Neptune-Cluster erstellen können. Ein skalierbares Ziel ist eine Ressource, die dank Application Auto Scaling auf- und abskaliert werden kann. Skalierbare Ziele werden eindeutig durch die Kombination von Ressourcen-ID, skalierbarer Dimension und Namespace identifiziert.

Um Auto Scaling mit der AWS CLI oder einer der folgenden zu konfigurieren AWS SDKs, können Sie die folgenden Optionen verwenden:

- AWS CLI:

  Rufen Sie den register-scalable-targetBefehl für einen Neptun-Cluster auf. Im folgenden Beispiel wird die gewünschte Kapazität einer Clusters mit dem Namen `mycluster` registriert, mit einer Mindestkapazität von einer Flotten-Instance und einer Höchstkapazität von acht Flotten-Instances.

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace neptune \
    --scalable-dimension neptune:cluster:ReadReplicaCount \
    --resource-id cluster:mycluster \
    --min-capacity 1 \
    --max-capacity 8
  ```

  Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

  ```
  {
      "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
  }
  ```

- AWS SDK:

  Rufen Sie den Vorgang RegisterScalableTarget auf und geben Sie `ResourceId`, `ScalableDimension`, `ServiceNamespace`, `MinCapacity`, und `MaxCapacity` als Parameter an.

Zugehörige Ressourcen

Weitere Informationen finden Sie unter Automatische Skalierung der Anzahl von Replikaten in einem Amazon Neptune Neptune-DB-Cluster im Neptune-Benutzerhandbuch.

Amazon SageMaker AI und Application Auto Scaling

Sie können SageMaker KI-Endpunktvarianten, bereitgestellte Parallelität für serverlose Endpunkte und Inferenzkomponenten mithilfe von Skalierungsrichtlinien für Zielverfolgung, schrittweiser Skalierung und geplanter Skalierung skalieren.

Verwenden Sie die folgenden Informationen, um Sie bei der Integration von SageMaker KI in Application Auto Scaling zu unterstützen.

Für SageMaker KI wurde eine serviceverknüpfte Rolle erstellt

Die folgende serviceverknüpfte Rolle wird automatisch in Ihrem erstellt, AWS-Konto wenn Sie SageMaker KI-Ressourcen als skalierbare Ziele mit Application Auto Scaling registrieren. Mit dieser Rolle kann Application Auto Scaling unterstützte Operationen innerhalb Ihres Kontos durchführen. Weitere Informationen finden Sie unter Servicegebundene Rollen für Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_SageMakerEndpoint`

Von der dienstgebundenen Rolle verwendeter Hauptdienst

Die im vorigen Abschnitt beschriebene dienstgebundene Rolle kann nur vom Hauptdienst übernommen werden, der durch die für die Rolle definierten vertrauenswürdigen Beziehungen autorisiert ist. Die von Application Auto Scaling verwendete dienstgebundene Rolle gewährt Zugriff auf den folgenden Hauptdienst:

- `sagemaker.application-autoscaling.amazonaws.com`

Registrierung von SageMaker KI-Endpunktvarianten als skalierbare Ziele mit Application Auto Scaling

Application Auto Scaling erfordert ein skalierbares Ziel, bevor Sie Skalierungsrichtlinien oder geplante Aktionen für ein SageMaker KI-Modell (Variante) erstellen können. Ein skalierbares Ziel ist eine Ressource, die dank Application Auto Scaling auf- und abskaliert werden kann. Skalierbare Ziele werden eindeutig durch die Kombination von Ressourcen-ID, skalierbarer Dimension und Namespace identifiziert.

Wenn Sie Auto Scaling mithilfe der SageMaker KI-Konsole konfigurieren, registriert SageMaker KI automatisch ein skalierbares Ziel für Sie.

Wenn Sie Auto Scaling über die AWS CLI oder eine der folgenden Optionen konfigurieren möchten AWS SDKs, können Sie die folgenden Optionen verwenden:

- AWS CLI:

Rufen Sie den register-scalable-targetBefehl für eine Produktvariante auf. Das folgende Beispiel registriert die gewünschte Anzahl von Instances für eine Produktvariante namens `my-variant`, die auf dem Endpunkt `my-endpoint` ausgeführt wird, mit einer Mindestkapazität von einer Instance und einer Höchstkapazität von acht Instances.

```
aws application-autoscaling register-scalable-target \
  --service-namespace sagemaker \
  --scalable-dimension sagemaker:variant:DesiredInstanceCount \
  --resource-id endpoint/my-endpoint/variant/my-variant \
  --min-capacity 1 \
  --max-capacity 8
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Rufen Sie den Vorgang RegisterScalableTarget auf und geben Sie `ResourceId`, `ScalableDimension`, `ServiceNamespace`, `MinCapacity`, und `MaxCapacity` als Parameter an.

Registrieren der bereitgestellten Gleichzeitigkeit von Serverless-Endpunkten als skalierbare Ziele mit Application Auto Scaling

Application Auto Scaling erfordert auch ein skalierbares Ziel, bevor Sie Skalierungsrichtlinien oder geplante Aktionen für die bereitgestellte Gleichzeitigkeit von Serverless-Endpunkten erstellen können.

Wenn Sie Auto Scaling mithilfe der SageMaker KI-Konsole konfigurieren, registriert SageMaker KI automatisch ein skalierbares Ziel für Sie.

Verwenden Sie andernfalls eine der folgenden Methoden, um das skalierbare Ziel zu registrieren:

- AWS CLI:

Rufen Sie den register-scalable-targetBefehl für eine Produktvariante auf. Das folgende Beispiel registriert die bereitgestellte Gleichzeitigkeit für eine Produktvariante namens `my-variant`, die auf dem Endpunkt `my-endpoint` ausgeführt wird, mit einer Mindestkapazität von eins und einer Höchstkapazität von zehn.

```
aws application-autoscaling register-scalable-target \
  --service-namespace sagemaker \
  --scalable-dimension sagemaker:variant:DesiredProvisionedConcurrency \
  --resource-id endpoint/my-endpoint/variant/my-variant \
  --min-capacity 1 \
  --max-capacity 10
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Rufen Sie den Vorgang RegisterScalableTarget auf und geben Sie `ResourceId`, `ScalableDimension`, `ServiceNamespace`, `MinCapacity`, und `MaxCapacity` als Parameter an.

Registrieren von Inferenzkomponenten als skalierbare Ziele mit Application Auto Scaling

Application Auto Scaling erfordert ein skalierbares Ziel, bevor Sie Skalierungsrichtlinien oder geplante Aktionen für Inferenzkomponenten erstellen können.

- AWS CLI:

  Rufen Sie den register-scalable-targetBefehl für eine Inferenzkomponente auf. Im folgenden Beispiel wird die gewünschte Kopienanzahl für eine Inferenzkomponente namens `my-inference-component` registriert, mit einer Mindestkapazität von null Kopien und einer Höchstkapazität von drei Kopien.

```
aws application-autoscaling register-scalable-target \
  --service-namespace sagemaker \
  --scalable-dimension sagemaker:inference-component:DesiredCopyCount \
  --resource-id inference-component/my-inference-component \
  --min-capacity 0 \
  --max-capacity 3
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Rufen Sie den Vorgang RegisterScalableTarget auf und geben Sie `ResourceId`, `ScalableDimension`, `ServiceNamespace`, `MinCapacity`, und `MaxCapacity` als Parameter an.

Zugehörige Ressourcen

Wenn Sie gerade erst mit Application Auto Scaling beginnen, finden Sie weitere nützliche Informationen zur Skalierung Ihrer SageMaker KI-Ressourcen im Amazon SageMaker AI Developer Guide:

- Automatisches Skalieren von Amazon SageMaker AI-Modellen
- Automatisches Skalieren von Provisioned Concurrency für einen serverlosen Endpunkt
- Legen Sie Richtlinien für die auto Skalierung für Endpunktbereitstellungen mit mehreren Modellen fest
- Automatische Skalierung eines asynchronen Endpunkts

> **Note**
>
> Im Jahr 2023 führte SageMaker KI neue Inferenzfunktionen ein, die auf Echtzeit-Inferenzendpunkten basieren. Sie erstellen einen SageMaker KI-Endpunkt mit einer Endpunktkonfiguration, die den Instanztyp und die anfängliche Anzahl der Instanzen für

den Endpunkt definiert. Erstellen Sie anschließend eine Inferenzkomponente, bei der es sich um ein SageMaker KI-Hosting-Objekt handelt, mit dem Sie ein Modell auf einem Endpunkt bereitstellen können. Informationen zur Skalierung von Inferenzkomponenten finden Sie im Blog unter Amazon SageMaker AI fügt neue Inferenzfunktionen hinzu, um die Bereitstellungskosten und die Latenz von Basismodellen zu reduzieren und die Kosten für die Modellbereitstellung mithilfe der neuesten Funktionen von Amazon SageMaker AI um durchschnittlich 50% zu reduzieren. AWS

Amazon EC2 Spot-Flotte und Application Auto Scaling

Sie können Spot Fleets mithilfe von Zielverfolgungs-Skalierungsrichtlinien, Stufenskalierungsrichtlinien und geplanter Skalierung skalieren.

Die folgenden Informationen helfen Ihnen bei der Integration von Spot Fleet mit Application Auto Scaling.

Serviceverknüpfte Rolle für Spot Fleet erstellt

Die folgende serviceverknüpfte Rolle wird automatisch in Ihrem erstellt, AWS-Konto wenn Sie Spot-Flottenressourcen als skalierbare Ziele mit Application Auto Scaling registrieren. Mit dieser Rolle kann Application Auto Scaling unterstützte Operationen innerhalb Ihres Kontos durchführen. Weitere Informationen finden Sie unter Servicegebundene Rollen für Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest`

Von der dienstgebundenen Rolle verwendeter Hauptdienst

Die im vorigen Abschnitt beschriebene dienstgebundene Rolle kann nur vom Hauptdienst übernommen werden, der durch die für die Rolle definierten vertrauenswürdigen Beziehungen autorisiert ist. Die von Application Auto Scaling verwendete dienstgebundene Rolle gewährt Zugriff auf den folgenden Hauptdienst:

- `ec2.application-autoscaling.amazonaws.com`

Registrierung von Spot Fleets als skalierbare Ziele mit Application Auto Scaling

Application Auto Scaling erfordert ein skalierbares Ziel, bevor Sie Skalierungsrichtlinien oder geplante Aktionen für ein Spot Fleet erstellen können. Ein skalierbares Ziel ist eine Ressource, die dank Application Auto Scaling auf- und abskaliert werden kann. Skalierbare Ziele werden eindeutig durch die Kombination von Ressourcen-ID, skalierbarer Dimension und Namespace identifiziert.

Wenn Sie die automatische Skalierung über die Spot Fleet-Konsole konfigurieren, registriert Spot Fleet automatisch ein skalierbares Ziel für Sie.

Wenn Sie Auto Scaling über die AWS CLI oder eine der folgenden Optionen konfigurieren möchten AWS SDKs, können Sie die folgenden Optionen verwenden:

- AWS CLI:

  Rufen Sie den Befehl register-scalable-target für eine Spot-Flotte auf. Das folgende Beispiel registriert die Zielkapazität einer Spot-Flotte anhand ihrer Anfrage-ID, mit einer Mindestkapazität von zwei Instanzen und einer Höchstkapazität von 10 Instanzen.

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace ec2 \
    --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
    --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --min-capacity 2 \
    --max-capacity 10
  ```

  Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

  ```
  {
      "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
  target/1234abcd56ab78cd901ef1234567890ab123"
  }
  ```

- AWS SDK:

  Rufen Sie den Vorgang RegisterScalableTarget auf und geben Sie `ResourceId`, `ScalableDimension`, `ServiceNamespace`, `MinCapacity`, und `MaxCapacity` als Parameter an.

Zugehörige Ressourcen

Weitere Informationen finden Sie unter Grundlegendes zur automatischen Skalierung für Spot Fleet im Amazon EC2 EC2-Benutzerhandbuch.

Amazon WorkSpaces und Application Auto Scaling

Sie können einen Pool von Skalierungsrichtlinien WorkSpaces mit Zielverfolgung, schrittweiser Skalierung und geplanter Skalierung skalieren.

Verwenden Sie die folgenden Informationen, um Sie bei der Integration WorkSpaces mit Application Auto Scaling zu unterstützen.

Eine mit dem Dienst verknüpfte Rolle wurde erstellt für WorkSpaces

Application Auto Scaling erstellt automatisch die serviceverknüpfte Rolle, die AWSServiceRoleForApplicationAutoScaling_WorkSpacesPool in Ihrem benannt ist AWS-Konto , wenn Sie WorkSpaces Ressourcen als skalierbare Ziele bei Application Auto Scaling registrieren. Weitere Informationen finden Sie unter Servicegebundene Rollen für Application Auto Scaling.

Diese dienstverknüpfte Rolle verwendet die verwaltete Richtlinie. AWSApplication AutoscalingWorkSpacesPoolPolicy Diese Richtlinie gewährt Application Auto Scaling die Erlaubnis, Amazon in WorkSpaces Ihrem Namen anzurufen. Weitere Informationen finden Sie AWSApplicationAutoscalingWorkSpacesPoolPolicyin der Referenz zu AWS verwalteten Richtlinien.

Von der dienstgebundenen Rolle verwendeter Hauptdienst

Die dienstbezogene Rolle vertraut darauf, dass der folgende Dienstprinzipal die Rolle übernimmt:

- `workspaces.application-autoscaling.amazonaws.com`

Registrierung von WorkSpaces Pools als skalierbare Ziele mit Application Auto Scaling

Application Auto Scaling erfordert ein skalierbares Ziel, bevor Sie Skalierungsrichtlinien oder geplante Aktionen für erstellen können WorkSpaces. Ein skalierbares Ziel ist eine Ressource, die dank Application Auto Scaling auf- und abskaliert werden kann. Skalierbare Ziele werden eindeutig durch die Kombination von Ressourcen-ID, skalierbarer Dimension und Namespace identifiziert.

Wenn Sie Auto Scaling über die WorkSpaces Konsole konfigurieren, wird WorkSpaces automatisch ein skalierbares Ziel für Sie registriert.

Wenn Sie Auto Scaling über die AWS CLI oder eine der folgenden Optionen konfigurieren möchten AWS SDKs, können Sie die folgenden Optionen verwenden:

- AWS CLI:

  Rufen Sie den register-scalable-targetBefehl für einen Pool von auf WorkSpaces. Im folgenden Beispiel wird die Zielkapazität eines Pools WorkSpaces anhand seiner Anforderungs-ID registriert. Die Mindestkapazität beträgt zwei virtuelle Desktops und die maximale Kapazität zehn virtuelle Desktops.

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace workspaces \
    --resource-id workspacespool/wspool-abcdef012 \
    --scalable-dimension workspaces:workspacespool:DesiredUserSessions \
    --min-capacity 2 \
    --max-capacity 10
  ```

  Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

  ```
  {
      "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
  }
  ```

- AWS SDK:

  Rufen Sie den Vorgang RegisterScalableTarget auf und geben Sie `ResourceId`, `ScalableDimension`, `ServiceNamespace`, `MinCapacity`, und `MaxCapacity` als Parameter an.

Zugehörige Ressourcen

Weitere Informationen finden Sie unter Auto Scaling for WorkSpaces Pools im Amazon WorkSpaces Administration Guide.

Benutzerdefinierte Ressourcen und Application Auto Scaling

Sie können benutzerdefinierte Ressourcen mithilfe von Zielverfolgungs-Skalierungsrichtlinien, Stufenskalierungsrichtlinien und geplanter Skalierung skalieren.

Die folgenden Informationen helfen Ihnen bei der Integration benutzerdefinierter Ressourcen in Application Auto Scaling.

Für benutzerdefinierte Ressourcen erstellte serviceverknüpfte Rolle

Die folgende serviceverknüpfte Rolle wird automatisch in Ihrem erstellt, AWS-Konto wenn Sie benutzerdefinierte Ressourcen als skalierbare Ziele mit Application Auto Scaling registrieren. Mit dieser Rolle kann Application Auto Scaling unterstützte Operationen innerhalb Ihres Kontos durchführen. Weitere Informationen finden Sie unter Servicegebundene Rollen für Application Auto Scaling.

- `AWSServiceRoleForApplicationAutoScaling_CustomResource`

Von der dienstgebundenen Rolle verwendeter Hauptdienst

Die im vorigen Abschnitt beschriebene dienstgebundene Rolle kann nur vom Hauptdienst übernommen werden, der durch die für die Rolle definierten vertrauenswürdigen Beziehungen autorisiert ist. Die von Application Auto Scaling verwendete dienstgebundene Rolle gewährt Zugriff auf den folgenden Hauptdienst:

- `custom-resource.application-autoscaling.amazonaws.com`

Registrierung von benutzerdefinierten Ressourcen als skalierbare Ziele mit Application Auto Scaling

Application Auto Scaling benötigt ein skalierbares Ziel, bevor Sie Skalierungsrichtlinien oder geplante Aktionen für eine benutzerdefinierte Ressource erstellen können. Ein skalierbares Ziel ist eine Ressource, die dank Application Auto Scaling auf- und abskaliert werden kann. Skalierbare Ziele werden eindeutig durch die Kombination von Ressourcen-ID, skalierbarer Dimension und Namespace identifiziert.

Um Auto Scaling mit der AWS CLI oder einer der folgenden zu konfigurieren AWS SDKs, können Sie die folgenden Optionen verwenden:

- AWS CLI:

  Rufen Sie den register-scalable-target-Befehl für eine benutzerdefinierte Ressource auf. Im folgenden Beispiel wird eine benutzerdefinierte Ressource als skalierbares Ziel registriert, mit einer gewünschten Mindestanzahl von einer Kapazitätseinheit und einer gewünschten Höchstanzahl von 10 Kapazitätseinheiten. Die Datei `custom-resource-id.txt` enthält eine Zeichenfolge, die die Ressourcen-ID identifiziert, die den Pfad zu der benutzerdefinierten Ressource über Ihren Amazon API Gateway-Endpunkt darstellt.

```
aws application-autoscaling register-scalable-target \
  --service-namespace custom-resource \
  --scalable-dimension custom-resource:ResourceType:Property \
  --resource-id file://~/custom-resource-id.txt \
  --min-capacity 1 \
  --max-capacity 10
```

  Inhalt von `custom-resource-id.txt`:

```
https://example.execute-api.us-west-2.amazonaws.com/prod/
scalableTargetDimensions/1-23456789
```

  Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK:

  Rufen Sie den Vorgang RegisterScalableTarget auf und geben Sie `ResourceId`, `ScalableDimension`, `ServiceNamespace`, `MinCapacity`, und `MaxCapacity` als Parameter an.

Zugehörige Ressourcen

Wenn Sie gerade erst mit Application Auto Scaling beginnen, finden Sie in der folgenden Dokumentation weitere nützliche Informationen zur Skalierung Ihrer benutzerdefinierten Ressourcen:

GitHubRepository

Konfigurieren Sie die Auto Scaling-Ressourcen für Anwendungen mithilfe von AWS CloudFormation

Application Auto Scaling ist in einen Service integriert AWS CloudFormation, der Sie bei der Modellierung und Einrichtung Ihrer AWS Ressourcen unterstützt, sodass Sie weniger Zeit mit der Erstellung und Verwaltung Ihrer Ressourcen und Infrastruktur verbringen müssen. Sie erstellen eine Vorlage, die alle AWS benötigten Ressourcen beschreibt und diese Ressourcen für Sie CloudFormation bereitstellt und konfiguriert.

Wenn Sie sie verwenden CloudFormation, können Sie Ihre Vorlage wiederverwenden, um Ihre Application Auto Scaling Scaling-Ressourcen konsistent und wiederholt einzurichten. Beschreiben Sie Ihre Ressourcen einmal und stellen Sie dann dieselben Ressourcen immer wieder in mehreren AWS-Konten Regionen bereit.

Application Auto Scaling und CloudFormation Vorlagen

Um Ressourcen für Application Auto Scaling und damit verbundene Dienste bereitzustellen und zu konfigurieren, müssen Sie CloudFormation Templates verstehen. Vorlagen sind formatierte Textdateien in JSON oder YAML. Diese Vorlagen beschreiben die Ressourcen, die Sie in Ihren CloudFormation Stacks bereitstellen möchten. Wenn Sie mit JSON oder YAML nicht vertraut sind, können Sie CloudFormation Designer verwenden, um Ihnen die ersten Schritte mit Vorlagen zu erleichtern. CloudFormation Weitere Informationen finden Sie unter Was ist CloudFormation - Designer? im AWS CloudFormation -Benutzerhandbuch.

Wenn Sie eine Stack-Vorlage für Application Auto Scaling-Ressourcen erstellen, müssen Sie die folgenden Angaben machen:

- Einen Namespace für den Zieldienst (z. B. **`appstream`**). Informationen zum Abrufen von AWS::ApplicationAutoScaling::ScalableTargetDienst-Namespaces finden Sie in der Referenz.
- Eine skalierbare Dimension, die mit der Zielressource verbunden ist (z. B. **`appstream:fleet:DesiredCapacity`**). Informationen zu skalierbaren Dimensionen finden Sie in der AWS::ApplicationAutoScaling::ScalableTargetReferenz.
- Eine Ressourcen-ID für die Zielressource (z. B. **`fleet/sample-fleet`**). In der AWS::ApplicationAutoScaling::ScalableTargetReferenz finden Sie Informationen zur Syntax und Beispiele für bestimmte Ressourcen IDs.

- Eine mit einem Service verknüpfte Rolle für die Zielressource (z. B. **`arn:aws:iam::012345678910:role/aws-service-role/appstream.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_AppStreamFleet`**). Informationen zur Rolle finden Sie in der ARN-Referenz für serviceverknüpfte Rollen Tabelle ARNs.

Weitere Informationen zu Application-Auto-Scaling-Ressourcen finden Sie in der Application Auto Scaling-Referenz im AWS CloudFormation -Benutzerhandbuch.

Beispielvorlagen-Snippets

In den folgenden Abschnitten des AWS CloudFormation Benutzerhandbuchs finden Sie Beispielausschnitte, die Sie in CloudFormation Vorlagen aufnehmen können:

- Beispiele für Skalierungsrichtlinien und geplante Aktionen finden Sie unter Konfigurieren von Application Auto Scaling Scaling-Ressourcen mit AWS CloudFormation.
- Weitere Beispiele für Skalierungsrichtlinien finden Sie unter AWS::ApplicationAutoScaling::ScalingPolicy.

Erfahren Sie mehr über CloudFormation

Weitere Informationen CloudFormation dazu finden Sie in den folgenden Ressourcen:

- AWS CloudFormation
- AWS CloudFormation Benutzerhandbuch
- CloudFormation API Reference
- AWS CloudFormation -Benutzerhandbuch für die Befehlszeilenschnittstelle

Geplante Skalierung von Application Auto Scaling

Mit der geplanten Skalierung können Sie eine automatische Skalierung für Ihre Anwendung auf der Grundlage vorhersehbarer Laständerungen einrichten, indem Sie geplante Aktionen erstellen, mit denen die Kapazität zu bestimmten Zeiten erhöht oder verringert wird. So können Sie Ihre Anwendung proaktiv skalieren, um sie an vorhersehbare Laständerungen anzupassen.

Beispiel: Angenommen, Sie haben ein regelmäßiges wöchentliches Datenverkehrsmuster, bei dem die Last in der Wochenmitte steigt und gegen Ende der Woche sinkt. Sie können in Application Auto Scaling einen Skalierungsplan konfigurieren, der sich an diesem Muster orientiert:

- Am Mittwochmorgen erhöht eine geplante Aktion die Kapazität, indem die zuvor festgelegte Mindestkapazität des skalierbaren Ziels erhöht wird.
- Am Freitagabend verringert eine weitere geplante Aktion die Kapazität, indem die zuvor festgelegte maximale Kapazität des skalierbaren Ziels verringert wird.

Mit diesen geplanten Skalierungsaktionen können Sie Kosten und Leistung optimieren. Ihre Anwendung verfügt über ausreichend Kapazität, um die Hauptverkehrsspitzen unter der Woche zu bewältigen, ohne dass zu anderen Zeiten unnötige Kapazitäten bereitgestellt werden.

Sie können geplante Skalierung und Skalierungsrichtlinien zusammen verwenden, um die Vorteile von proaktiven und reaktiven Skalierungsansätzen zu nutzen. Nachdem eine geplante Skalierungsaktion ausgeführt wurde, kann die Skalierungsrichtlinie weiterhin Entscheidungen darüber treffen, ob die Kapazität weiter skaliert werden soll. So können Sie sicherstellen, dass Sie über eine ausreichende Kapazität verfügen, um die Last für Ihre Anwendung zu bewältigen. Während sich Ihre Anwendung an die Nachfrage anpasst, muss die aktuelle Kapazität innerhalb der minimalen und maximalen Kapazität liegen, die durch Ihre geplante Aktion festgelegt wurde.

Inhalt

So funktioniert die geplante Skalierung für Application Auto Scaling

In diesem Thema wird beschrieben, wie die geplante Skalierung funktioniert, und es werden die wichtigsten Überlegungen vorgestellt, die Sie verstehen müssen, um sie effektiv nutzen zu können.

Inhalt

Funktionsweise

Um die geplante Skalierung zu nutzen, erstellen Sie geplante Aktionen, die Application Auto Scaling anweisen, zu bestimmten Zeiten Skalierungsaktivitäten durchzuführen. Wenn Sie eine geplante Aktion erstellen, geben Sie das skalierbare Ziel, den Zeitpunkt der Skalierungsaktivität sowie eine Mindest- und eine Maximalkapazität an. Sie können geplante Aktionen erstellen, die nur einmal oder nach einem wiederkehrenden Zeitplan skaliert werden.

Zum festgelegten Zeitpunkt skaliert Application Auto Scaling auf der Grundlage der neuen Kapazitätswerte, indem die aktuelle Kapazität mit der festgelegten Mindest- und Höchstkapazität verglichen wird.

- Wenn die aktuelle Kapazität geringer ist als die angegebene Mindestkapazität, wird Application Auto Scaling auf die angegebene Mindestkapazität erweitert (erhöht).
- Wenn die aktuelle Kapazität größer als die angegebene Maximalkapazität ist, skaliert Application Auto Scaling auf die angegebene Maximalkapazität hinein (verringert die Kapazität).

Überlegungen

Beachten Sie bei der Erstellung einer geplanten Aktion Folgendes:

- Eine geplante Aktion setzt die `MinCapacity` und `MaxCapacity` zum angegebenen Zeitpunkt und Datum auf den Wert, der durch die geplante Aktion angegeben ist. Die Anfrage kann optional

nur eine dieser Größen enthalten. Sie können beispielsweise eine geplante Aktion erstellen, bei der nur die minimale Kapazität angegeben ist. In einigen Fällen müssen Sie jedoch beide Größen einbeziehen, um sicherzustellen, dass die neue Mindestkapazität nicht größer als die maximale Kapazität ist oder die neue maximale Kapazität nicht unter der Mindestkapazität liegt.

- Standardmäßig befinden sich die wiederkehrenden Zeitpläne in UTC (Coordinated Universal Time). Sie können die Zeitzone ändern, wenn sie Ihrer örtlichen Zeitzone oder einer Zeitzone in einem anderen Teil Ihres Netzwerks entsprechen soll. Wenn Sie eine Zeitzone angeben, in der die Sommerzeit gilt, wird die Aktion automatisch an die Sommerzeit angepasst. Weitere Informationen finden Sie unter Planen Sie wiederkehrende Skalierungsaktionen mit Application Auto Scaling.
- Sie können die geplante Skalierung für ein skalierbares Ziel vorübergehend deaktivieren. Dadurch können Sie verhindern, dass geplante Aktionen aktiv sind, ohne sie löschen zu müssen. Sie können die geplante Skalierung dann fortsetzen, wenn Sie sie erneut verwenden möchten. Weitere Informationen finden Sie unter Die Skalierung von Application Auto Scaling unterbrechen und wiederaufnehmen.
- Die Reihenfolge, in der geplante Aktionen ausgeführt werden, ist für dasselbe skalierbare Ziel garantiert, jedoch nicht für geplante Aktionen über skalierbare Ziele hinweg.
- Um eine geplante Aktion erfolgreich abzuschließen, muss sich die angegebene Ressource im Zielservice in einem skalierbaren Zustand befinden. Ist dies nicht der Fall, schlägt die Anforderung fehl und gibt eine Fehlermeldung zurück, z. B. `Resource Id [ActualResourceId] is not scalable. Reason: The status of all DB instances must be 'available' or 'incompatible-parameters'`.
- Aufgrund der verteilten Natur von Application Auto Scaling und den Zieldiensten kann die Verzögerung zwischen dem Zeitpunkt der Auslösung der geplanten Aktion und dem Zeitpunkt, zu dem der Zieldienst die Skalierungsaktion honoriert, einige Sekunden betragen. Da geplante Aktionen in der Reihenfolge ausgeführt werden, in der sie angegeben wurden, kann die Ausführung von geplanten Aktionen mit nahe beieinander liegenden Startzeiten länger dauern.

Häufig verwendete Befehle zur Erstellung, Verwaltung und Löschung von geplanten Aktionen

Zu den häufig verwendeten Befehlen für die Arbeit mit der Zeitplan-Skalierung gehören:

- register-scalable-targetum Ressourcen als skalierbare Ziele zu registrieren AWS oder anzupassen (eine Ressource, die Application Auto Scaling skalieren kann) und die Skalierung auszusetzen und wieder aufzunehmen.

- put-scheduled-actionum geplante Aktionen für ein vorhandenes skalierbares Ziel hinzuzufügen oder zu ändern.
- describe-scaling-activitiesum Informationen über Skalierungsaktivitäten in einer AWS Region zurückzugeben.
- describe-scheduled-actionsum Informationen über geplante Aktionen in einer AWS Region zurückzugeben.
- delete-scheduled-actionum eine geplante Aktion zu löschen.

Zugehörige Ressourcen

Ein detailliertes Beispiel für die Verwendung der geplanten Skalierung finden Sie im Blogbeitrag Scheduling AWS Lambda Provisioned Concurrency for recurriring peak usage auf dem AWS Compute-Blog.

Informationen zum Erstellen von geplanten Aktionen für Auto Scaling finden Sie unter Geplante Skalierung für Amazon EC2 Auto Scaling im Benutzerhandbuch für Amazon EC2 Auto Scaling.

Einschränkungen

Die folgenden Einschränkungen gelten für die Verwendung der geplanten Skalierung:

- Die Namen der geplanten Aktionen müssen pro skalierbarem Ziel eindeutig sein.
- Application Auto Scaling bietet keine Präzision zweiter Ebene in Zeitplanausdrücken. Die feinste Zeitauflösung bei Verwendung eines Cron-Ausdrucks ist 1 Minute.
- Das skalierbare Ziel kann nicht ein Amazon MSK-Cluster sein. Geplante Skalierung wird für Amazon MSK nicht unterstützt.
- Der Konsolenzugriff zum Anzeigen, Hinzufügen, Aktualisieren oder Entfernen von geplanten Aktionen für skalierbare Ressourcen hängt von der verwendeten Ressource ab. Weitere Informationen finden Sie unter AWS-Services die Sie mit Application Auto Scaling verwenden können.

Erstellen Sie geplante Aktionen für Application Auto Scaling mit dem AWS CLI

Die folgenden Beispiele zeigen, wie Sie mithilfe des AWS CLI put-scheduled-actionBefehls geplante Aktionen erstellen. Wenn Sie die neue Kapazität angeben, können Sie eine Mindestkapazität, eine Maximalkapazität oder beides angeben.

In diesen Beispielen werden skalierbare Ziele für einige der Dienste verwendet, die in Application Auto Scaling integriert sind. Um ein anderes skalierbares Ziel zu verwenden, geben Sie seinen Namespace in`--service-namespace`, seine skalierbare Dimension in `--scalable-dimension` und seine Ressourcen-ID in `--resource-id` an.

Denken Sie bei der Verwendung von daran AWS CLI, dass Ihre Befehle in der für Ihr Profil AWS-Region konfigurierten Version ausgeführt werden. Wenn Sie die Befehle in einer anderen Region ausführen möchten, ändern Sie entweder die Standardregion für Ihr Profil, oder verwenden Sie den `--region`-Parameter mit dem Befehl.

Beispiele

- Erstellen einer geplanten Aktion, die nur einmal ausgeführt wird
- Erstellen einer geplanten Aktion, die in einem wiederkehrenden Intervall ausgeführt wird
- Erstellen einer geplanten Aktion, die nach einem wiederkehrenden Zeitplan ausgeführt wird
- Erstellen einer einmaligen geplanten Aktion, die eine Zeitzone angibt
- Erstellen einer wiederkehrenden geplanten Aktion, die eine Zeitzone angibt

Erstellen einer geplanten Aktion, die nur einmal ausgeführt wird

Um Ihr skalierbares Ziel nur einmal zu einem bestimmten Datum und einer bestimmten Uhrzeit automatisch zu skalieren, verwenden Sie die Option `--schedule "at(`*`yyyy-mm-ddThh:mm:ss`*`)"`.

Example Beispiel: Einmalige horizontale Skalierung nach oben

Nachfolgend ein Beispiel für die Erstellung einer geplanten Aktion zum Abbau von Kapazitäten zu einem bestimmten Datum und einer bestimmten Uhrzeit.

Wenn der für `MinCapacity` angegebene Wert zu dem für `--schedule` angegebenen Datum und Zeitpunkt (22:00 Uhr UTC am 31. März 2021) über der aktuellen Kapazität liegt, skaliert Application Auto Scaling auf `MinCapacity`.

Linux, macOS oder Unix

```
aws application-autoscaling put-scheduled-action --service-namespace custom-resource \
  --scalable-dimension custom-resource:ResourceType:Property \
  --resource-id file://~/custom-resource-id.txt \
  --scheduled-action-name scale-out \
  --schedule "at(2021-03-31T22:00:00)" \
  --scalable-target-action MinCapacity=3
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace custom-resource ^
  --scalable-dimension custom-resource:ResourceType:Property ^
  --resource-id file://~/custom-resource-id.txt ^
  --scheduled-action-name scale-out ^
  --schedule "at(2021-03-31T22:00:00)" ^
  --scalable-target-action MinCapacity=3
```

Wenn diese eingeplante Aktion ausgeführt wird und die maximale Kapazität unter dem für die minimale Kapazität angegebenen Wert liegt, müssen Sie eine neue minimale und maximale Kapazität angeben und nicht nur die minimale Kapazität.

Example Beispiel: Einmalige horizontale Skalierung nach unten

Nachfolgend ein Beispiel für die Erstellung einer geplanten Aktion zur Kapazitätsanpassung zu einem bestimmten Datum und einer bestimmten Uhrzeit.

Wenn der für `MaxCapacity` angegebene Wert zu dem für `--schedule` angegebenen Datum und Zeitpunkt (22:30 Uhr UTC am 31. März 2021) unter der aktuellen Kapazität liegt, skaliert Application Auto Scaling automatisch ab auf `MaxCapacity`.

Linux, macOS oder Unix

```
aws application-autoscaling put-scheduled-action --service-namespace custom-resource \
  --scalable-dimension custom-resource:ResourceType:Property \
  --resource-id file://~/custom-resource-id.txt \
  --scheduled-action-name scale-in \
```

```
  --schedule "at(2021-03-31T22:30:00)" \
  --scalable-target-action MinCapacity=0,MaxCapacity=0
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace custom-resource ^
  --scalable-dimension custom-resource:ResourceType:Property ^
  --resource-id file://~/custom-resource-id.txt ^
  --scheduled-action-name scale-in ^
  --schedule "at(2021-03-31T22:30:00)" ^
  --scalable-target-action MinCapacity=0,MaxCapacity=0
```

Erstellen einer geplanten Aktion, die in einem wiederkehrenden Intervall ausgeführt wird

Um eine Skalierung in einem wiederkehrenden Intervall zu planen, verwenden Sie die Option `--schedule "rate(value unit)"`. Der Wert muss eine positive ganze Zahl sein. Die Einheit kann `minute`, `minutes`, `hour`, `hours`, `day` oder `days` sein. Weitere Informationen finden Sie unter Bewertungsausdrücke im EventBridge Amazon-Benutzerhandbuch.

Im Folgenden finden Sie ein Beispiel für eine geplante Aktion, die einen Ratenausdruck verwendet.

Wenn der für `MinCapacity` angegebene Wert im angegebenen Zeitplan (alle 5 Stunden, beginnend am 30. Januar 2021 um 12:00 Uhr UTC und endend am 31. Januar 2021 um 22:00 Uhr UTC) über der aktuellen Kapazität liegt, skaliert Application Auto Scaling automatisch auf `MinCapacity`. Wenn der für `MaxCapacity` angegebene Wert unter der aktuellen Kapazität liegt, skaliert Application Auto Scaling auf `MaxCapacity`.

Linux, macOS oder Unix

```
aws application-autoscaling put-scheduled-action --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service \
  --scheduled-action-name my-recurring-action \
  --schedule "rate(5 hours)" \
  --start-time 2021-01-30T12:00:00 \
  --end-time 2021-01-31T22:00:00 \
  --scalable-target-action MinCapacity=3,MaxCapacity=10
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace ecs ^
  --scalable-dimension ecs:service:DesiredCount ^
  --resource-id service/my-cluster/my-service ^
  --scheduled-action-name my-recurring-action ^
  --schedule "rate(5 hours)" ^
  --start-time 2021-01-30T12:00:00 ^
  --end-time 2021-01-31T22:00:00 ^
  --scalable-target-action MinCapacity=3,MaxCapacity=10
```

Erstellen einer geplanten Aktion, die nach einem wiederkehrenden Zeitplan ausgeführt wird

Planen Sie eine Skalierung im Rahmen eines sich wiederholenden Zeitplans unter Verwendung der `--schedule "cron(fields)"`-Option. Weitere Informationen finden Sie unter Planen Sie wiederkehrende Skalierungsaktionen mit Application Auto Scaling.

Im Folgenden finden Sie ein Beispiel für eine geplante Aktion, die einen Cron-Ausdruck verwendet.

Wenn der für `MinCapacity` angegebene Wert im angegebenen Zeitplan (täglich um 9:00 Uhr UTC) über der aktuellen Kapazität liegt, skaliert Application Auto Scaling auf `MinCapacity`. Wenn der für `MaxCapacity` angegebene Wert unter der aktuellen Kapazität liegt, skaliert Application Auto Scaling auf `MaxCapacity`.

Linux, macOS oder Unix

```
aws application-autoscaling put-scheduled-action --service-namespace appstream \
  --scalable-dimension appstream:fleet:DesiredCapacity \
  --resource-id fleet/sample-fleet \
  --scheduled-action-name my-recurring-action \
  --schedule "cron(0 9 * * ? *)" \
  --scalable-target-action MinCapacity=10,MaxCapacity=50
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace appstream ^
  --scalable-dimension appstream:fleet:DesiredCapacity ^
  --resource-id fleet/sample-fleet ^
  --scheduled-action-name my-recurring-action ^
  --schedule "cron(0 9 * * ? *)" ^
  --scalable-target-action MinCapacity=10,MaxCapacity=50
```

Erstellen einer einmaligen geplanten Aktion, die eine Zeitzone angibt

Geplante Aktionen sind standardmäßig auf die UTC-Zeitzone eingestellt. Um eine andere Zeitzone anzugeben, fügen Sie die Option `--timezone` ein und geben den kanonischen Namen für die Zeitzone an (z. B. `America/New_York`). Weitere Informationen finden Sie unter. https://www.joda.org/joda-time/timezones.htmlDort finden Sie Informationen zu den IANA-Zeitzonen, die bei Anrufen put-scheduled-actionunterstützt werden.

Im folgenden Beispiel wird die Option `--timezone` bei der Erstellung einer geplanten Aktion zur Skalierung der Kapazität zu einem bestimmten Datum und einer bestimmten Uhrzeit verwendet.

Wenn der für `MinCapacity` angegebene Wert zu dem für `--schedule` angegebenen Datum und Zeitpunkt (17:00 Uhr Ortszeit am 31. Januar 2021) über der aktuellen Kapazität liegt, skaliert Application Auto Scaling auf `MinCapacity` ab. Wenn der für `MaxCapacity` angegebene Wert unter der aktuellen Kapazität liegt, skaliert Application Auto Scaling auf `MaxCapacity`.

Linux, macOS oder Unix

```
aws application-autoscaling put-scheduled-action --service-namespace comprehend \
  --scalable-dimension comprehend:document-classifier-endpoint:DesiredInferenceUnits \
  --resource-id arn:aws:comprehend:us-west-2:123456789012:document-classifier-endpoint/
EXAMPLE \
  --scheduled-action-name  my-one-time-action \
  --schedule "at(2021-01-31T17:00:00)" --timezone "America/New_York" \
  --scalable-target-action MinCapacity=1,MaxCapacity=3
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace comprehend ^
  --scalable-dimension comprehend:document-classifier-endpoint:DesiredInferenceUnits ^
  --resource-id arn:aws:comprehend:us-west-2:123456789012:document-classifier-endpoint/
EXAMPLE ^
  --scheduled-action-name  my-one-time-action ^
  --schedule "at(2021-01-31T17:00:00)" --timezone "America/New_York" ^
  --scalable-target-action MinCapacity=1,MaxCapacity=3
```

Erstellen einer wiederkehrenden geplanten Aktion, die eine Zeitzone angibt

Es folgt ein Beispiel, bei dem die `--timezone`-Option verwendet wird, wenn Sie eine wiederkehrende geplante Aktion zum Skalieren der Kapazität erstellen. Weitere Informationen finden Sie unter Planen Sie wiederkehrende Skalierungsaktionen mit Application Auto Scaling.

Wenn der für `MinCapacity` angegebene Wert im angegebenen Zeitplan (jeden Montag bis Freitag um 18:00 Uhr Ortszeit) über der aktuellen Kapazität liegt, skaliert Application Auto Scaling auf `MinCapacity` ab. Wenn der für `MaxCapacity` angegebene Wert unter der aktuellen Kapazität liegt, skaliert Application Auto Scaling auf `MaxCapacity`.

Linux, macOS oder Unix

```
aws application-autoscaling put-scheduled-action --service-namespace lambda \
  --scalable-dimension lambda:function:ProvisionedConcurrency \
  --resource-id function:my-function:BLUE \
  --scheduled-action-name my-recurring-action \
  --schedule "cron(0 18 ? * MON-FRI *)" --timezone "Etc/GMT+9" \
  --scalable-target-action MinCapacity=10,MaxCapacity=50
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace lambda ^
  --scalable-dimension lambda:function:ProvisionedConcurrency ^
  --resource-id function:my-function:BLUE ^
  --scheduled-action-name my-recurring-action ^
  --schedule "cron(0 18 ? * MON-FRI *)" --timezone "Etc/GMT+9" ^
  --scalable-target-action MinCapacity=10,MaxCapacity=50
```

Beschreiben Sie die geplante Skalierung für Application Auto Scaling mit dem AWS CLI

Diese AWS CLI Beispielbefehle beschreiben Skalierungsaktivitäten und geplante Aktionen unter Verwendung von Ressourcen aus Diensten, die in Application Auto Scaling integriert sind. Geben Sie für ein anderes skalierbares Ziel seinen Namespace in`--service-namespace`, seine skalierbare Dimension in `--scalable-dimension` und seine Ressourcen-ID in `--resource-id` an.

Denken Sie bei der Verwendung von daran AWS CLI, dass Ihre Befehle in der für Ihr Profil AWS-Region konfigurierten Version ausgeführt werden. Wenn Sie die Befehle in einer anderen Region ausführen möchten, ändern Sie entweder die Standardregion für Ihr Profil, oder verwenden Sie den `--region`-Parameter mit dem Befehl.

Beispiele

- Beschreiben Sie die Skalierungsaktivitäten für einen Service
- Beschreiben Sie die geplanten Aktionen für einen Dienst

- Beschreiben Sie die geplanten Aktionen für ein skalierbares Ziel

Beschreiben Sie die Skalierungsaktivitäten für einen Service

Verwenden Sie den describe-scaling-activitiesBefehl, um die Skalierungsaktivitäten für alle skalierbaren Ziele in einem angegebenen Service-Namespace anzuzeigen.

Das folgende Beispiel ruft die Skalierungsaktivitäten ab, die mit dem Service-Namespace dynamodb verbunden sind.

Linux, macOS oder Unix

```
aws application-autoscaling describe-scaling-activities --service-namespace dynamodb
```

Windows

```
aws application-autoscaling describe-scaling-activities --service-namespace dynamodb
```

Ausgabe

Wenn der Befehl erfolgreich ist, gibt er eine Ausgabe zurück, die der folgenden ähnelt.

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 10.",
            "ResourceId": "table/my-table",
            "ActivityId": "4d1308c0-bbcf-4514-a673-b0220ae38547",
            "StartTime": 1561574415.086,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574449.51,
            "Cause": "maximum capacity was set to 10",
            "StatusMessage": "Successfully set write capacity units to 10. Change
 successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 5 and max capacity to 10",
            "ResourceId": "table/my-table",
```

```
            "ActivityId": "f2b7847b-721d-4e01-8ef0-0c8d3bacc1c7",
            "StartTime": 1561574414.644,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-second-scheduled-action was triggered",
            "StatusMessage": "Successfully set min capacity to 5 and max capacity to
 10",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 15.",
            "ResourceId": "table/my-table",
            "ActivityId": "d8ea4de6-9eaa-499f-b466-2cc5e681ba8b",
            "StartTime": 1561574108.904,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574140.255,
            "Cause": "minimum capacity was set to 15",
            "StatusMessage": "Successfully set write capacity units to 15. Change
 successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 15 and max capacity to 20",
            "ResourceId": "table/my-table",
            "ActivityId": "3250fd06-6940-4e8e-bb1f-d494db7554d2",
            "StartTime": 1561574108.512,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-first-scheduled-action was triggered",
            "StatusMessage": "Successfully set min capacity to 15 and max capacity to
 20",
            "StatusCode": "Successful"
        }
    ]
}
```

Um diesen Befehl so zu ändern, dass er nur die Skalierungsaktivitäten für eines Ihrer skalierbaren Ziele abruft, fügen Sie die Option `--resource-id` hinzu.

Beschreiben Sie die geplanten Aktionen für einen Dienst

Verwenden Sie den describe-scheduled-actionsBefehl, um die geplanten Aktionen für alle skalierbaren Ziele in einem angegebenen Dienst-Namespace zu beschreiben.

Das folgende Beispiel ruft die geplanten Aktionen ab, die mit dem Service-Namespace `ec2` verbunden sind.

Linux, macOS oder Unix

```
aws application-autoscaling describe-scheduled-actions --service-namespace ec2
```

Windows

```
aws application-autoscaling describe-scheduled-actions --service-namespace ec2
```

Ausgabe

Wenn der Befehl erfolgreich ist, gibt er eine Ausgabe zurück, die der folgenden ähnelt.

```
{
    "ScheduledActions": [
        {
            "ScheduledActionName": "my-one-time-action",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:493a6261-fbb9-432d-855d-3c302c14bdb9:resource/ec2/
spot-fleet-request/sfr-107dc873-0802-4402-a901-37294EXAMPLE:scheduledActionName/my-one-
time-action",
            "ServiceNamespace": "ec2",
            "Schedule": "at(2021-01-31T17:00:00)",
            "Timezone": "America/New_York",
            "ResourceId": "spot-fleet-request/sfr-107dc873-0802-4402-
a901-37294EXAMPLE",
            "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
            "ScalableTargetAction": {
                "MaxCapacity": 1
            },
            "CreationTime": 1607454792.331
        },
        {
            "ScheduledActionName": "my-recurring-action",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:493a6261-fbb9-432d-855d-3c302c14bdb9:resource/ec2/
spot-fleet-request/sfr-107dc873-0802-4402-a901-37294EXAMPLE:scheduledActionName/my-
recurring-action",
            "ServiceNamespace": "ec2",
            "Schedule": "rate(5 minutes)",
```

```
            "ResourceId": "spot-fleet-request/sfr-107dc873-0802-4402-
a901-37294EXAMPLE",
            "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
            "StartTime": 1604059200.0,
            "EndTime": 1612130400.0,
            "ScalableTargetAction": {
                "MinCapacity": 3,
                "MaxCapacity": 10
            },
            "CreationTime": 1607454949.719
        },
        {
            "ScheduledActionName": "my-one-time-action",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:4bce34c7-bb81-4ecf-b776-5c726efb1567:resource/ec2/
spot-fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE:scheduledActionName/my-one-
time-action",
            "ServiceNamespace": "ec2",
            "Schedule": "at(2020-12-08T9:36:00)",
            "Timezone": "America/New_York",
            "ResourceId": "spot-fleet-request/sfr-40edeb7b-9ae7-44be-
bef2-5c4c8EXAMPLE",
            "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
            "ScalableTargetAction": {
                "MinCapacity": 1,
                "MaxCapacity": 3
            },
            "CreationTime": 1607456031.391
        }
    ]
}
```

Beschreiben Sie die geplanten Aktionen für ein skalierbares Ziel

Um Informationen über die geplanten Aktionen für ein bestimmtes skalierbares Ziel abzurufen, fügen Sie die `--resource-id` Option hinzu, wenn Sie geplante Aktionen mit dem describe-scheduled-actionsBefehl beschreiben.

Wenn Sie die Option `--scheduled-action-names` hinzufügen und den Namen einer geplanten Aktion als Wert angeben, gibt der Befehl nur die geplante Aktion zurück, deren Name übereinstimmt, wie im folgenden Beispiel gezeigt.

Linux, macOS oder Unix

```
aws application-autoscaling describe-scheduled-actions --service-namespace ec2 \
  --resource-id spot-fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE \
  --scheduled-action-names my-one-time-action
```

Windows

```
aws application-autoscaling describe-scheduled-actions --service-namespace ec2 ^
  --resource-id spot-fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE ^
  --scheduled-action-names my-one-time-action
```

Ausgabe

Wenn der Befehl erfolgreich ist, gibt er eine Ausgabe zurück, die der folgenden ähnelt. Wenn Sie mehr als einen Wert für angegeben haben`--scheduled-action-names`, enthält die Ausgabe alle geplanten Aktionen, deren Namen übereinstimmen.

```
{
    "ScheduledActions": [
        {
            "ScheduledActionName": "my-one-time-action",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:4bce34c7-bb81-4ecf-b776-5c726efb1567:resource/ec2/
spot-fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE:scheduledActionName/my-one-
time-action",
            "ServiceNamespace": "ec2",
            "Schedule": "at(2020-12-08T9:36:00)",
            "Timezone": "America/New_York",
            "ResourceId": "spot-fleet-request/sfr-40edeb7b-9ae7-44be-
bef2-5c4c8EXAMPLE",
            "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
            "ScalableTargetAction": {
                "MinCapacity": 1,
                "MaxCapacity": 3
            },
            "CreationTime": 1607456031.391
        }
    ]
}
```

Planen Sie wiederkehrende Skalierungsaktionen mit Application Auto Scaling

> **Important**
>
> Für Hilfe zu Cron-Ausdrücken für Amazon EC2 Auto Scaling lesen Sie das Thema Regelmäßige Zeitpläne im Benutzerhandbuch zu Amazon EC2 Auto Scaling. Mit Amazon EC2 Auto Scaling verwenden Sie die herkömmliche Cron-Syntax anstelle der benutzerdefinierten Cron-Syntax, die Application Auto Scaling verwendet.

Mithilfe eines Cron-Ausdrucks können Sie geplante Aktionen erstellen, die nach einem wiederkehrenden Zeitplan ausgeführt werden.

Um einen wiederkehrenden Zeitplan zu erstellen, geben Sie einen Cron-Ausdruck und eine Zeitzone an, um zu beschreiben, wann diese geplante Aktion wiederholt werden soll. Die unterstützten Zeitzonenwerte sind die kanonischen Namen der von Joda-Time unterstützten IANA-Zeitzonen (z. B. `Etc/GMT+9` oder `Pacific/Tahiti`). Sie können optional ein Datum und eine Uhrzeit für die Startzeit, die Endzeit oder beides angeben. Ein Beispielbefehl, mit dem AWS CLI eine geplante Aktion erstellt wird, finden Sie unterErstellen einer wiederkehrenden geplanten Aktion, die eine Zeitzone angibt.

Der unterstützte Cron-Ausdruck besteht aus sechs Feldern, getrennt durch Leerzeichen: [Minute] [Stunde] [Tag_des_Monats] [Monat_des_Jahres] [Wochentag] [Jahr]. Beispielsweise konfiguriert der Cron-Ausdruck `30 6 ? * MON *` eine geplante Aktion, die jeden Montag um 6:30 Uhr wiederholt wird. Das Sternchen wird als Platzhalter verwendet, um alle Werte für ein Feld abzugleichen.

Weitere Informationen zur Cron-Syntax für geplante Aktionen von Application Auto Scaling finden Sie unter Referenz zu Cron-Ausdrücken im EventBridge Amazon-Benutzerhandbuch.

Wählen Sie bei der Erstellung eines wiederkehrenden Zeitplans Ihre Start- und Endzeiten sorgfältig aus. Beachten Sie Folgendes:

- Wenn Sie eine Startzeit angeben, führt Application Auto Scaling die Aktion zu dieser Zeit aus und führt die Aktion dann auf der Grundlage der angegebenen Wiederholung aus.
- Wenn Sie eine Endzeit angeben, wird die Aktion nach dieser Zeit nicht mehr wiederholt. Application Auto Scaling merkt sich keine früheren Werte und kehrt nach der Endzeit zu diesen früheren Werten zurück.

- Die Start- und Endzeit müssen in UTC festgelegt werden, wenn Sie das AWS CLI oder verwenden, um eine geplante AWS SDKs Aktion zu erstellen oder zu aktualisieren.

Beispiele

Sie können sich auf die folgende Tabelle beziehen, wenn Sie einen wiederkehrenden Zeitplan für ein skalierbares Application-Auto-Scaling-Ziel erstellen. Die folgenden Beispiele zeigen die korrekte Syntax für die Verwendung von Application Auto Scaling zum Erstellen oder Aktualisieren einer geplanten Aktion.

Minuten	Stunden	Tag des Monats	Monat	Wochentag	Jahr	Bedeutung
0	10	*	*	?	*	Ausführung jeden Tag um 10:00 Uhr (UTC)
15	12	*	*	?	*	Ausführung jeden Tag um 12:15 Uhr (UTC)
0	18	?	*	MO-FR	*	Ausführung jeden Montag bis Freitag um 18:00 Uhr (UTC)
0	8	1	*	?	*	Lauf um 8:00 Uhr (UTC) am 1. Tag

Minuten	Stunden	Tag des Monats	Monat	Wochentag	Jahr	Bedeutung
						eines jeden Monats
0/15	*	*	*	?	*	Ausführung alle 15 Minuten
0/10	*	?	*	MO-FR	*	Ausführung alle 10 Minuten von Montag bis Freitag
0/5	8-17	?	*	MO-FR	*	Ausführung alle 5 Minuten von Montag bis Freitag zwischen 08:00 Uhr und 17:55 Uhr (UTC)

Exception

Sie können auch einen Cron-Ausdruck mit einem Zeichenfolgenwert erstellen, der sieben Felder enthält. In diesem Fall können Sie die ersten drei Felder verwenden, um den Zeitpunkt anzugeben, zu dem eine geplante Aktion ausgeführt werden soll, einschließlich der Sekunden. Der vollständige Cron-Ausdruck hat die folgenden durch Leerzeichen getrennten Felder: [Sekunden] [Minuten] [Stunden] [Tag_des_Monats] [Monat] [Tag_des_Woche] [Jahr]. Dieser Ansatz garantiert jedoch nicht, dass die geplante Aktion genau in der von Ihnen angegebenen Sekunde ausgeführt wird.

Außerdem kann es sein, dass einige Service-Konsolen das Sekundenfeld in einem Cron-Ausdruck nicht unterstützen.

Ausschalten der geplanten Skalierung für ein skalierbares Ziel

Sie können die geplante Skalierung vorübergehend deaktivieren, ohne Ihre geplanten Aktionen zu löschen. Weitere Informationen finden Sie unter Die Skalierung von Application Auto Scaling unterbrechen und wiederaufnehmen.

Um die geplante Skalierung auszusetzen

Unterbrechen Sie die geplante Skalierung auf einem skalierbaren Ziel, indem Sie den register-scalable-targetBefehl mit der `--suspended-state` Option verwenden und den Wert des `ScheduledScalingSuspended` Attributs angeben`true`, wie im folgenden Beispiel gezeigt.

Linux, macOS oder Unix

```
aws application-autoscaling register-scalable-target --service-namespace rds \
  --scalable-dimension rds:cluster:ReadReplicaCount --resource-id cluster:my-db-cluster
 \
  --suspended-state '{"ScheduledScalingSuspended": true}'
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace rds ^
  --scalable-dimension rds:cluster:ReadReplicaCount --resource-id cluster:my-db-cluster
 ^
  --suspended-state "{\"ScheduledScalingSuspended\": true}"
```

Ausgabe

Wenn der Befehl erfolgreich ist, gibt er den ARN des skalierbaren Ziels zurück. Es folgt eine Beispielausgabe.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Um die geplante Skalierung fortzusetzen

Um die geplante Skalierung fortzusetzen, führen Sie den register-scalable-target Befehl erneut aus und geben Sie `false` als Wert für an`ScheduledScalingSuspended`.

Löschen Sie eine geplante Aktion für Application Auto Scaling mit dem AWS CLI

Wird eine geplante Aktion nicht mehr benötigt, können Sie sie löschen.

Um Ihre geplante Aktion zu löschen

Verwenden Sie den Befehl delete-scheduled-action. Bei Erfolg gibt dieser Befehl keine Ausgabe zurück.

Linux, macOS oder Unix

```
aws application-autoscaling delete-scheduled-action \
  --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-37294EXAMPLE \
  --scheduled-action-name my-recurring-action
```

Windows

```
aws application-autoscaling delete-scheduled-action ^
  --service-namespace ec2 ^
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity ^
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-37294EXAMPLE ^
  --scheduled-action-name my-recurring-action
```

So melden Sie das skalierbare Ziel ab

Wenn Sie auch mit dem skalierbaren Ziel fertig sind, können Sie es deregistrieren. Verwenden Sie den folgenden deregister-scalable-target-Befehl. Wenn Skalierungsrichtlinien oder geplante Aktionen noch nicht gelöscht wurden, werden sie mit diesem Befehl gelöscht. Bei Erfolg gibt dieser Befehl keine Ausgabe zurück.

Linux, macOS oder Unix

```
aws application-autoscaling deregister-scalable-target \
  --service-namespace ec2 \
```

```
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-37294EXAMPLE
```

Windows

```
aws application-autoscaling deregister-scalable-target ^
  --service-namespace ec2 ^
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity ^
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-37294EXAMPLE
```

Zielverfolgungs-Skalierungsrichtlinien für Application Auto Scaling

Eine Skalierungsrichtlinie für die Ziel-Nachverfolgung skaliert Ihre Anwendung automatisch basierend auf einem Zielmetrikwert. Auf diese Weise kann Ihre Anwendung ohne manuelles Eingreifen eine optimale Leistung und Kosteneffizienz aufrechterhalten.

Bei der Ziel-Nachverfolgung wählen Sie eine Metrik und einen Zielwert aus, der die ideale durchschnittliche Auslastung oder den idealen Durchsatz für Ihre Anwendung darstellt. Application Auto Scaling erstellt und verwaltet die CloudWatch Alarme, die Skalierungsereignisse auslösen, wenn die Metrik vom Ziel abweicht. Dies ist vergleichbar mit der Art und Weise, wie ein Thermostat eine Zieltemperatur aufrechterhält.

Ein Beispiel: Angenommen, Sie verfügen über eine Anwendung, die derzeit in der Spot-Flotte ausgeführt wird, und die CPU-Auslastung der Flotte soll bei etwa 50 Prozent bleiben, wenn sich die Anwendungslast ändert. Auf diese Weise erlangen Sie zusätzliche Kapazität für Datenverkehrsspitzen, ohne übermäßig viele Ressourcen im Leerlauf zu verwalten.

Hierzu können Sie eine Skalierungsrichtlinie für die Zielverfolgung erstellen, die eine durchschnittliche CPU-Auslastung von 50 Prozent vorsieht. Dann skaliert Application Auto Scaling auf (erhöht die Kapazität), wenn die CPU 50 Prozent überschreitet, um die erhöhte Auslastung zu bewältigen. Wenn die CPU-Auslastung unter 50 Prozent sinkt, wird abskaliert (die Kapazität verringert), um die Kosten in Zeiten geringer Auslastung zu optimieren.

Richtlinien zur Zielverfolgung machen die manuelle Definition von CloudWatch Alarmen und Skalierungsanpassungen überflüssig. Application Auto Scaling erledigt dies automatisch auf der Grundlage des von Ihnen festgelegten Ziels.

Richtlinien für die Ziel-Nachverfolgung können entweder auf vordefinierten oder benutzerdefinierten Metriken basieren:

- Vordefinierte Metriken – von Application Auto Scaling bereitgestellte Metriken wie die durchschnittliche CPU-Auslastung oder die durchschnittliche Anzahl von Anfragen pro Ziel.
- Benutzerdefinierte Metriken — Sie können Metriken verwenden, um Metriken zu kombinieren, bestehende Metriken zu nutzen oder Ihre eigenen benutzerdefinierten Metriken zu CloudWatch verwenden, die in veröffentlicht wurden.

Wählen Sie eine Metrik, die sich umgekehrt proportional zu einer Änderung der Kapazität Ihres skalierbaren Ziels ändert. Wenn Sie also die Kapazität verdoppeln, sinkt die Metrik um 50 Prozent. Auf diese Weise können die Metrikdaten genau proportionale Skalierungsereignisse auslösen.

Inhalt

So funktioniert die Zielverfolgungsskalierung für Application Auto Scaling

In diesem Thema wird beschrieben, wie die Skalierung der Zielverfolgung funktioniert, und es werden die wichtigsten Elemente einer Skalierungsrichtlinie für die Zielverfolgung vorgestellt.

Inhalt

Funktionsweise

Um die Skalierung der Zielverfolgung zu verwenden, erstellen Sie eine Skalierungsrichtlinie für die Zielverfolgung und geben Folgendes an:

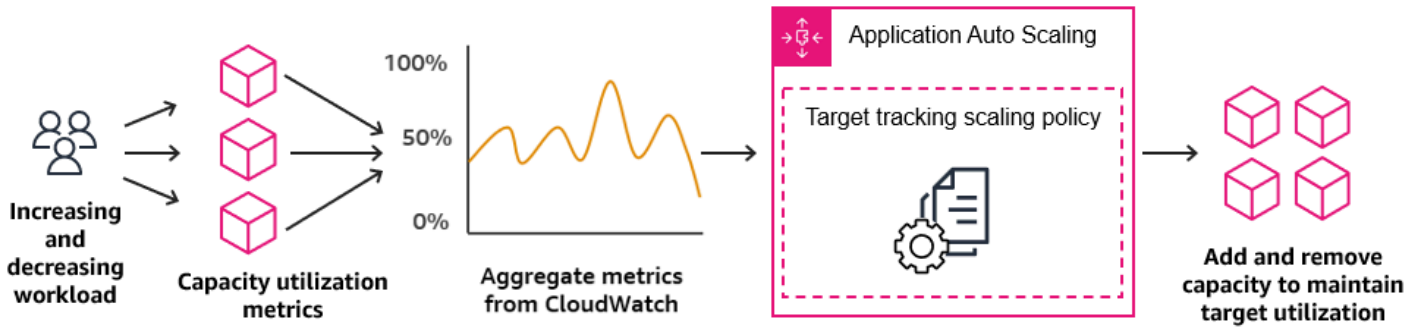
- CloudWatch Metrik — Eine zu verfolgende Metrik, z. B. die durchschnittliche CPU-Auslastung oder die durchschnittliche Anzahl von Anfragen pro Ziel.
- Zielwert – der Zielwert für die Metrik, z. B. 50 Prozent CPU-Auslastung oder 1 000 Anfragen pro Ziel pro Minute.

Application Auto Scaling erstellt und verwaltet die CloudWatch Alarme, die die Skalierungsrichtlinie aufrufen, und berechnet die Skalierungsanpassung auf der Grundlage der Metrik und des Zielwerts. Es wird so viel Kapazität wie erforderlich hinzugefügt oder entfernt, damit die Metrik auf oder nahe an dem Zielwert gehalten wird.

Wenn die Metrik über dem Zielwert liegt, skaliert Application Auto Scaling auf, indem Kapazität hinzugefügt wird, um die Differenz zwischen dem Metrikwert und dem Zielwert zu verringern. Wenn die Metrik unter dem Zielwert liegt, skaliert Application Auto Scaling ab, indem Kapazität entfernt wird.

Zwischen den Skalierungsaktivitäten liegen Ruhephasen, um schnelle Kapazitätsschwankungen zu vermeiden. Sie können die Ruhephasen für Ihre Richtlinie optional konfigurieren.

Das folgende Diagramm zeigt einen Überblick über die Funktionsweise einer Zielverfolgungsrichtlinie, wenn die Einrichtung abgeschlossen ist.



Hinweis: Eine Skalierungsrichtlinie für die Ziel-Nachverfolgung ist aggressiver beim Hinzufügen von Kapazität bei steigender Auslastung als beim Entfernen von Kapazität bei sinkender Auslastung. Wenn zum Beispiel die in der Richtlinie angegebene Metrik ihren Zielwert erreicht, geht die Richtlinie davon aus, dass Ihre Anwendung bereits stark belastet ist. Daher reagiert sie, indem sie so schnell wie möglich Kapazität proportional zum metrischen Wert hinzufügt. Je höher die Metrik, desto mehr Kapazität wird hinzugefügt.

Wenn die Kennzahl unter den Zielwert fällt, wird die Richtlinie nicht skaliert, wenn sie berechnet, dass das Entfernen einer Mindestkapazitätseinheit die Metrik wahrscheinlich wieder über den Zielwert bringen würde. In diesem Fall verlangsamt sie die Skalierung, indem sie Kapazitäten nur dann entfernt, wenn die Auslastung einen Schwellenwert überschreitet, der weit genug unter dem Zielwert liegt (in der Regel um mehr als 10 %), damit die Auslastung als verlangsamt angesehen werden kann. Mit diesem vorsichtigeren Verhalten soll sichergestellt werden, dass Kapazitäten erst dann entfernt werden, wenn die Anwendung nicht mehr so häufig aufgerufen wird wie zuvor.

Auswahl von Metriken

Sie können Skalierungsrichtlinien zur Zielverfolgung mit vordefinierten oder benutzerdefinierten Metriken erstellen.

Wenn Sie eine Skalierungsrichtlinie zur Zielverfolgung mit einem vordefinierten Metriktyp erstellen, wählen Sie eine Metrik aus der Liste der vordefinierten Metriken in Vordefinierte Metriken für Skalierungsrichtlinien für die Zielverfolgung aus.

Berücksichtigen Sie die folgenden Aspekte, wenn Sie eine Metrik auswählen:

- Nicht alle benutzerdefinierten Metriken funktionieren für die Zielverfolgung. Die Metrik muss eine gültige Auslastungsmetrik sein und beschreiben, wie ausgelastet ein skalierbares Ziel ist. Der Metrikwert muss proportional zur Kapazität des skalierbaren Ziels steigen oder fallen, damit die metrischen Daten zur proportionalen Skalierung des skalierbaren Ziels verwendet werden können.
- Um die Metrik `ALBRequestCountPerTarget` zu verwenden, müssen Sie den Parameter `ResourceLabel` angeben, um die Zielgruppe zu identifizieren, die der Metrik zugeordnet ist.
- Wenn eine Metrik echte Werte von 0 ausgibt CloudWatch (z. B.`ALBRequestCountPerTarget`), kann Application Auto Scaling auf 0 skalieren, wenn über einen längeren Zeitraum kein Datenverkehr zu Ihrer Anwendung erfolgt. Damit Ihr skalierbares Ziel auf 0 skaliert wird, wenn keine Anforderungen an es weitergeleitet werden, muss die Mindestkapazität des skalierbaren Ziels auf 0 gesetzt werden.
- Anstatt neue Metriken zur Verwendung in Ihrer Skalierungsrichtlinie zu veröffentlichen, können Sie mit metrischer Mathematik bestehende Metriken kombinieren. Weitere Informationen finden Sie unter Erstellen einer Skalierungsrichtlinie für Zielnachverfolgung für Application Auto Scaling mit Metrikberechnungen.
- Informationen dazu, ob der von Ihnen verwendete Service die Angabe einer benutzerdefinierten Metrik in der Konsole des Service unterstützt, finden Sie in der Dokumentation für den betreffenden Service.

- Wir empfehlen, Metriken zu verwenden, die in einminütigen Intervallen verfügbar sind, damit Sie schneller auf Änderungen der Auslastung reagieren können. Die Zielverfolgung wertet Metriken für alle vordefinierten und benutzerdefinierten Metriken aus, die mit einer Granularität von einer Minute aggregiert sind, aber die zugrunde liegende Metrik veröffentlicht die Daten möglicherweise weniger häufig. So werden beispielsweise alle Amazon-EC2-Metriken standardmäßig in Fünf-Minuten-Intervallen gesendet, können aber auch auf eine Minute konfiguriert werden (bekannt als detaillierte Überwachung). Diese Entscheidung liegt bei den einzelnen Services. Die meisten versuchen, das kleinstmögliche Intervall zu verwenden.

Definieren des Zielwerts

Wenn Sie eine Skalierungsrichtlinie für die Zielverfolgung erstellen, müssen Sie einen Zielwert angeben. Der Zielwert stellt die optimale durchschnittliche Auslastung oder den idealen durchschnittlichen Durchsatz für Ihre Anwendung dar. Für eine kosteneffiziente Ressourcennutzung sollte der Zielwert auf einen möglichst hohen Wert mit einem angemessenen Puffer für unerwartete Datenverkehrserhöhungen festgelegt werden. Wenn Ihre Anwendung optimal für einen normalen Datenverkehrsfluss aufskaliert wird, sollte der tatsächliche Metrikwert dem Zielwert entsprechen oder knapp darunter liegen.

Wenn eine Skalierungsrichtlinie auf dem Durchsatz basiert, z. B. der Anzahl der Anfragen pro Ziel für einen Application Load Balancer, dem Netzwerk-E/A oder anderen Zählmetriken, stellt der Zielwert den optimalen durchschnittlichen Durchsatz einer einzelnen Einheit (z. B. eines einzelnen Ziels aus Ihrer Zielgruppe für Application Load Balancer) für einen Zeitraum von einer Minute dar.

Ruhephasen definieren

In Ihrer Skalierungsrichtlinie für die Zielverfolgung können Sie optional Ruhephasen definieren.

Eine Ruhephase ist die Zeitspanne, die die Skalierungsrichtlinie warten muss, bis eine vorherige Skalierungsaktivität wirksam wird.

Es gibt zwei Arten von Ruhephasen:

- Mit der Abkühlungsphase der Aufskalierung wird beabsichtigt, kontinuierlich (aber nicht übermäßig) zu skalieren. Nachdem Application Auto Scaling unter Verwendung einer Skalierungsrichtlinie erfolgreich aufskaliert wurde, wird die Berechnung der Ruhezeit gestartet. Eine Skalierungsrichtlinie erhöht die gewünschte Kapazität nicht erneut, es sei denn, es wird eine größere Aufskalierung ausgelöst oder die Ruhephase endet. Während die Scale-Out-Ruhephase wirksam ist, wird die

durch die initiierende horizontale Skalierung nach oben (Scale-Out) hinzugefügte Kapazität als Teil der gewünschten Kapazität für die nächste horizontale Skalierung nach oben berechnet.

- Mit der Ruhephase für die Abskalierung ist beabsichtigt, die Abskalierung konservativ durchzuführen, um die Verfügbarkeit Ihrer Anwendung zu schützen, sodass Abskalierungsaktivitäten blockiert werden, bis die Ruhephase für die Abskalierung abgelaufen ist. Wenn jedoch ein anderer Alarm während der Abkühlphase nach einer Abskalier-Aktivität eine Aufskalier-Aktivität auslöst, wird das Ziel durch Application Auto Scaling sofort abskaliert. In diesem Fall wird die Ruhephase für die Abskalierung angehalten und nicht abgeschlossen.

Jede Ruhephase wird in Sekunden gemessen und gilt nur für Skalierungsrichtlinien-bezogene Skalierungen. Wenn eine geplante Aktion während einer Ruhephase zum geplanten Zeitpunkt beginnt, kann sie umgehend eine Skalierung auslösen, ohne das Ablaufen der Ruhephase abzuwarten.

Sie können mit den Standardwerten beginnen, die später optimiert werden können. So kann es beispielsweise erforderlich sein, die Ruhephase zu verlängern, um zu verhindern, dass Ihre Skalierungsrichtlinie zur Ziel-Nachverfolgung zu aggressiv auf Änderungen reagiert, die über kurze Zeiträume auftreten.

Standardwerte

Application Auto Scaling bietet einen Standardwert von 600 für ElastiCache und einen Standardwert von 300 für die folgenden skalierbaren Ziele:

- WorkSpaces Flotten von Anwendungen
- Aurora-DB-Cluster
- ECS-Services
- Neptune-Cluster
- SageMaker Varianten von KI-Endpunkten
- SageMaker Komponenten der KI-Inferenz
- SageMaker Serverlos bereitgestellte Parallelität mit KI
- Spot Flotten
- Pool von WorkSpaces
- Benutzerdefinierte Ressourcen

Für alle anderen skalierbaren Ziele ist der Standardwert 0 oder NULL:

- Amazon Comprehend-Dokumentklassifizierungs- und Entitätserkennungs-Endpunkte
- DynamoDB-Tabellen und globale sekundäre Indizes
- Amazon Keyspace-Tabellen
- Parallelität per Lambda
- Amazon MSK-Broker-Speicher

NULL-Werte werden genauso behandelt wie Nullwerte, wenn Application Auto Scaling die Ruhephase auswertet.

Sie können jeden der Standardwerte, einschließlich NULL-Werte, aktualisieren, um Ihre eigenen Ruhephasen festzulegen.

Überlegungen

Bei der Arbeit mit Skalierungsrichtlinien für die Zielverfolgung ist Folgendes zu beachten:

- Erstellen, bearbeiten oder löschen Sie keine CloudWatch Alarme, die mit einer Skalierungsrichtlinie für die Zielverfolgung verwendet werden. Application Auto Scaling erstellt und verwaltet die CloudWatch Alarme, die mit Ihren Skalierungsrichtlinien für die Zielverfolgung verknüpft sind, und löscht sie, wenn sie nicht mehr benötigt werden.
- Wenn der Metrik Datenpunkte fehlen, führt dies dazu, dass der CloudWatch Alarmstatus auf `INSUFFICIENT_DATA` geändert wird. In diesem Fall kann Application Auto Scaling das skalierbare Ziel erst wieder skalieren, wenn neue Datenpunkte gefunden wurden. Weitere Informationen finden Sie unter Konfiguration der Behandlung fehlender Daten durch CloudWatch Alarme im CloudWatch Amazon-Benutzerhandbuch.
- Wenn die Metrik konstruktionsbedingt nur spärlich gemeldet wird, kann metrische Mathematik hilfreich sein. Um beispielsweise die neuesten Werte zu verwenden, verwenden Sie die Funktion `FILL(m1,REPEAT)`, wobei `m1` die Metrik ist.
- Möglicherweise werden Lücken zwischen den Datenpunkten für den Zielwert und die aktuelle Metrik angezeigt. Dies liegt daran, dass Application Auto Scaling immer konservativ vorgeht, indem sie auf- oder abrundet, wenn sie bestimmt, wie viel Kapazität hinzugefügt oder entfernt werden soll. Dadurch wird verhindert, dass zu wenig Kapazität hinzufügt oder zu viel Kapazität entfernt wird. Bei einem skalierbaren Ziel mit geringer Kapazität können die tatsächlichen metrischen Datenpunkte jedoch scheinbar weit vom Zielwert entfernt sein.

Nehmen wir beispielsweise an, Sie legen einen Zielwert von 50 Prozent für die CPU-Auslastung fest und Ihre Auto Scaling Scaling-Gruppe überschreitet dann das Ziel. Wir könnten bestimmen, dass durch das Hinzufügen von 1,5 Instances die CPU-Auslastung auf beinahe 50 Prozent sinkt. Da es nicht möglich ist, 1,5 Instances hinzuzufügen, runden wir diesen Wert auf und fügen zwei Instances hinzu. Dadurch wird die CPU-Auslastung möglicherweise auf einen Wert unter 50 Prozent verringert, es wird jedoch sichergestellt, dass Ihre Anwendung über genügend Ressourcen verfügt, um dies zu unterstützen. Wenn wir feststellen, dass das Entfernen von 0,5 Instances Ihre CPU-Auslastung auf über 50 Prozent erhöht, entscheiden wir uns ebenfalls dafür, erst dann zu skalieren, wenn die Metrik so niedrig ist, dass wir glauben, dass die Skalierung keine Schwankungen verursacht.

Bei einem skalierbaren Ziel mit größerer Kapazität führt das Hinzufügen oder Entfernen von Kapazität zu einem geringeren Abstand zwischen dem Zielwert und den tatsächlichen metrischen Datenpunkten.

- Eine Skalierungsrichtlinie für die Ziel-Nachverfolgung geht davon aus, dass sie eine horizontale Skalierung nach oben vornehmen soll, wenn die angegebene Metrik über dem Zielwert liegt. Sie können keine Skalierungsrichtlinie für die Ziel-Nachverfolgung verwenden, um eine horizontale Skalierung nach oben vorzunehmen, wenn die angegebene Metrik unter dem Zielwert liegt.

Mehrere Skalierungsrichtlinien

Sie können mehrere Skalierungsrichtlinien für die Ziel-Nachverfolgung für ein skalierbares Ziel besitzen, vorausgesetzt, dass diese alle verschiedene Metriken verwenden. Die Absicht von Application Auto Scaling ist es, der Verfügbarkeit immer Vorrang einzuräumen. Daher unterscheidet sich das Verhalten von Application Auto Scaling, je nachdem, ob die Ziel-Tracking-Richtlinien für die Skalierung nach außen oder nach innen bereit sind. Sofern Richtlinien für die Ziel-Nachverfolgung für die horizontale Skalierung nach oben bereit sind, findet eine horizontale Skalierung des skalierbaren Ziels nach oben statt. Eine horizontale Skalierung nach unten wird jedoch nur vorgenommen, wenn alle Richtlinien für die Ziel-Nachverfolgung (mit aktivierter horizontaler Skalierung nach unten) zur horizontalen Skalierung nach unten bereit sind.

Wenn mehrere Richtlinien das skalierbare Ziel gleichzeitig zum Auf- oder Abskalieren anweisen, erfolgt die Skalierung durch Application Auto Scaling auf Grundlage der Richtlinie, die die größte Kapazität für die Ab- und Aufskalierung bereitstellt. Das bietet Ihnen eine größere Flexibilität für verschiedene Szenarien und stellt sicher, dass immer ausreichend Kapazität zur Verarbeitung Ihrer Workloads vorhanden ist.

Sie können den Abskalierungsteil einer Skalierungsrichtlinie für die Zielnachverfolgung deaktivieren, um eine andere Methode für die Abskalierung zu verwenden als für die Aufskalierung. Sie können z. B. eine Schrittskalierungsrichtlinie für die Skalierung nach unten verwenden, während Sie eine Skalierungsrichtlinie für die Ziel-Nachverfolgung für die Skalierung nach oben verwenden.

Sie sollten bei der Verwendung von Zielverfolgungs-Skalierungsrichtlinien mit Schrittskalierungsrichtlinien jedoch vorsichtig sein, da Konflikte zwischen diesen Richtlinien zu unerwünschtem Verhalten führen können. Wenn beispielsweise die Schrittskalierungsrichtlinie eine Abwärtsskalierungsaktivität initiiert, bevor die Zielverfolgungsrichtlinie abwärts skaliert werden kann, wird die Abwärtsskalierungsaktivität nicht blockiert. Nach Abschluss der herunterskalierenden Aktivität könnte die Zielverfolgungsrichtlinie das skalierbare Ziel anweisen, erneut zu skalieren.

Für zyklisch anfallende Verarbeitungslasten haben Sie außerdem die Möglichkeit, Kapazitätsänderungen in einem Zeitplan mithilfe der geplanten Skalierung zu automatisieren. Für jede geplante Aktion können ein neuer Kapazitätsmindestwert und ein neuer Kapazitätshöchstwert festgelegt werden. Diese Werte bilden die Grenzen der Skalierungsrichtlinie. Die Kombination aus geplanter Skalierung und Skalierung zur Zielnachverfolgung kann dazu beitragen, die Auswirkungen eines starken Anstiegs des Auslastungsgrades zu verringern, wenn die Kapazität sofort benötigt wird.

Häufig verwendete Befehle zur Erstellung, Verwaltung und Löschung von Skalierungsrichtlinien

Zu den häufig verwendeten Befehlen für die Arbeit mit Skalierungsrichtlinien gehören:

- register-scalable-targetum Ressourcen als skalierbare Ziele zu registrieren AWS oder anzupassen (eine Ressource, die Application Auto Scaling skalieren kann) und die Skalierung auszusetzen und wieder aufzunehmen.
- put-scaling-policyum Skalierungsrichtlinien für ein vorhandenes skalierbares Ziel hinzuzufügen oder zu ändern.
- describe-scaling-activitiesum Informationen über Skalierungsaktivitäten in einer AWS Region zurückzugeben.
- describe-scaling-policiesum Informationen über Skalierungsrichtlinien in einer AWS Region zurückzugeben.
- delete-scaling-policyum eine Skalierungsrichtlinie zu löschen.

Zugehörige Ressourcen

Informationen zum Erstellen von Skalierungsrichtlinien für die Ziel-Nachverfolgung für Auto-Scaling-Gruppen finden Sie unter Skalierungsrichtlinien für die Ziel-Nachverfolgung für Amazon EC2 Auto Scaling im Benutzerhandbuch für Amazon EC2 Auto Scaling.

Einschränkungen

Bei der Verwendung von Zielverfolgungs-Skalierungsrichtlinien gibt es folgende Einschränkungen:

- Das skalierbare Ziel kann nicht ein Amazon EMR-Cluster sein. Zielverfolgungs-Skalierungsrichtlinien werden für Amazon EMR nicht unterstützt.
- Wenn ein Amazon MSK-Cluster das skalierbare Ziel ist, ist scale in deaktiviert und kann nicht aktiviert werden.
- Sie können die `RegisterScalableTarget` oder `PutScalingPolicy` API-Operationen nicht verwenden, um einen AWS Auto Scaling Skalierungsplan zu aktualisieren.
- Der Konsolenzugriff zum Anzeigen, Hinzufügen, Aktualisieren oder Entfernen von Skalierungsrichtlinien für die Ziel-Nachverfolgung auf skalierbaren Ressourcen hängt von der verwendeten Ressource ab. Weitere Informationen finden Sie unter AWS-Services die Sie mit Application Auto Scaling verwenden können.

Erstellen Sie eine Skalierungsrichtlinie für die Zielverfolgung für Application Auto Scaling mithilfe der AWS CLI

In diesem Beispiel AWS CLI werden Befehle verwendet, um eine Ziel-Racking-Richtlinie für eine Amazon EC2-Spot-Flotte zu erstellen. Geben Sie für ein anderes skalierbares Ziel seinen Namespace in `--service-namespace`, seine skalierbare Dimension in `--scalable-dimension` und seine Ressourcen-ID in an. `--resource-id`

Denken Sie bei der Verwendung von daran AWS CLI, dass Ihre Befehle in der für Ihr Profil AWS-Region konfigurierten Version ausgeführt werden. Wenn Sie die Befehle in einer anderen Region ausführen möchten, ändern Sie entweder die Standardregion für Ihr Profil, oder verwenden Sie den `--region`-Parameter mit dem Befehl.

Aufgaben

- Schritt 1: Registrieren Sie ein skalierbares Ziel

Schritt 1: Registrieren Sie ein skalierbares Ziel

Wenn Sie dies noch nicht getan haben, registrieren Sie das skalierbare Ziel. Verwenden Sie den register-scalable-targetBefehl, um eine bestimmte Ressource im Zieldienst als skalierbares Ziel zu registrieren. Im folgenden Beispiel wird eine Spot Fleet-Anfrage mit Application Auto Scaling registriert. Application Auto Scaling kann die Anzahl der Instanzen in der Spot Fleet auf ein Minimum von 2 und ein Maximum von 10 Instanzen skalieren. Ersetzen Sie jeden *user input placeholder* durch Ihre Informationen.

Linux, macOS oder Unix

```
aws application-autoscaling register-scalable-target --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --min-capacity 2 --max-capacity 10
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace ec2 ^
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity ^
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE ^
  --min-capacity 2 --max-capacity 10
```

Ausgabe

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück. Es folgt eine Beispielausgabe.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Schritt 2: Erstellen einer Skalierungsrichtlinie für die Ziel-Nachverfolgung

Um eine Skalierungsrichtlinie für die Zielverfolgung zu erstellen, können Sie die folgenden Beispiele verwenden, um Ihnen den Einstieg zu erleichtern.

So erstellen Sie eine Skalierungsrichtlinie für die Ziel-Nachverfolgung

1. Verwenden Sie den folgenden `cat` Befehl, um einen Zielwert für Ihre Skalierungsrichtlinie und eine vordefinierte Metrikspezifikation in einer JSON-Datei mit dem Namen `config.json` in Ihrem Home-Verzeichnis zu speichern. Im Folgenden finden Sie ein Beispiel für eine Konfiguration zur Zielverfolgung, mit der die durchschnittliche CPU-Auslastung bei 50 Prozent gehalten wird.

```
$ cat ~/config.json
{
  "TargetValue": 50.0,
  "PredefinedMetricSpecification":
    {
      "PredefinedMetricType": "EC2SpotFleetRequestAverageCPUUtilization"
    }
}
```

   Weitere Informationen finden Sie PredefinedMetricSpecificationin der API-Referenz für Application Auto Scaling.

   Alternativ können Sie eine benutzerdefinierte Metrik für die Skalierung verwenden, indem Sie eine benutzerdefinierte Metrikspezifikation erstellen und Werte für jeden Parameter aus CloudWatch hinzufügen. Im Folgenden finden Sie ein Beispiel für eine Konfiguration zur Zielverfolgung, bei der die durchschnittliche Auslastung der angegebenen Metrik bei 100 belassen wird.

```
$ cat ~/config.json
{
   "TargetValue": 100.0,
   "CustomizedMetricSpecification":{
      "MetricName": "MyUtilizationMetric",
      "Namespace": "MyNamespace",
      "Dimensions": [
         {
            "Name": "MyOptionalMetricDimensionName",
            "Value": "MyOptionalMetricDimensionValue"
         }
      ],
      "Statistic": "Average",
      "Unit": "Percent"
   }
```

```
}
```

Weitere Informationen finden Sie CustomizedMetricSpecificationin der API-Referenz für Application Auto Scaling.

2. Verwenden Sie den folgenden Befehl put-scaling-policy zusammen mit der von Ihnen erstellten Datei `config.json`, um eine Skalierungsrichtlinie mit dem Namen `cpu50-target-tracking-scaling-policy` zu erstellen.

   Linux, macOS oder Unix

```
aws application-autoscaling put-scaling-policy --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --policy-name cpu50-target-tracking-scaling-policy --policy-type
 TargetTrackingScaling \
  --target-tracking-scaling-policy-configuration file://config.json
```

   Windows

```
aws application-autoscaling put-scaling-policy --service-namespace ec2 ^
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity ^
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE ^
  --policy-name cpu50-target-tracking-scaling-policy --policy-type
 TargetTrackingScaling ^
  --target-tracking-scaling-policy-configuration file://config.json
```

   Ausgabe

   Bei Erfolg gibt dieser Befehl die Namen ARNs und der beiden CloudWatch Alarme zurück, die in Ihrem Namen erstellt wurden. Es folgt eine Beispielausgabe.

```
{
    "PolicyARN": "arn:aws:autoscaling:region:account-
id:scalingPolicy:policy-id:resource/ec2/spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE:policyName/cpu50-target-tracking-scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-
b46e-434a-a60f-3b36d653feca",
```

```
            "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-
d19b-4a63-a812-6c67aaf2910d",
            "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d"
        }
    ]
}
```

Schritt 3: Beschreiben Sie die Skalierungsrichtlinien für die Zielverfolgung

Sie können alle Skalierungsrichtlinien für den angegebenen Service-Namespace beschreiben, indem Sie den folgenden describe-scaling-policies-Befehl verwenden.

```
aws application-autoscaling describe-scaling-policies --service-namespace ec2
```

Sie können die Ergebnisse mithilfe des Parameters `--query` filtern, um nur die Skalierungsrichtlinien zur Ziel-Nachverfolgung zu erhalten. Weitere Informationen über die Syntax von `query`, finden Sie unter Steuerung der Befehlsausgabe vom AWS CLI im AWS Command Line Interface Benutzerhandbuch.

Linux, macOS oder Unix

```
aws application-autoscaling describe-scaling-policies --service-namespace ec2 \
  --query 'ScalingPolicies[?PolicyType==`TargetTrackingScaling`]'
```

Windows

```
aws application-autoscaling describe-scaling-policies --service-namespace ec2 ^
  --query "ScalingPolicies[?PolicyType==`TargetTrackingScaling`]"
```

Ausgabe

Es folgt eine Beispielausgabe.

```
[
```

```
    {
        "PolicyARN": "PolicyARN",
        "TargetTrackingScalingPolicyConfiguration": {
            "PredefinedMetricSpecification": {
                "PredefinedMetricType": "EC2SpotFleetRequestAverageCPUUtilization"
            },
            "TargetValue": 50.0
        },
        "PolicyName": "cpu50-target-tracking-scaling-policy",
        "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
        "ServiceNamespace": "ec2",
        "PolicyType": "TargetTrackingScaling",
        "ResourceId": "spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
        "Alarms": [
            {
                "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-
b46e-434a-a60f-3b36d653feca",
                "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca"
            },
            {
                "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-
d19b-4a63-a812-6c67aaf2910d",
                "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d"
            }
        ],
        "CreationTime": 1515021724.807
    }
]
```

Löschen Sie eine Skalierungsrichtlinie für die Zielverfolgung für Application Auto Scaling mithilfe der AWS CLI

Wenn Sie mit einer Richtlinie für die Skalierung der Ziel-Nachverfolgung fertig sind, können Sie sie mit dem Befehl delete-scaling-policy löschen.

Mit dem folgenden Befehl wird die angegebene Skalierungsrichtlinie zur Ziel-Nachverfolgung für die angegebene Spot-Flottenanforderung gelöscht. Außerdem werden die CloudWatch Alarme gelöscht, die Application Auto Scaling in Ihrem Namen erstellt hat.

Linux, macOS oder Unix

```
aws application-autoscaling delete-scaling-policy --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --policy-name cpu50-target-tracking-scaling-policy
```

Windows

```
aws application-autoscaling delete-scaling-policy --service-namespace ec2 ^
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity ^
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE ^
  --policy-name cpu50-target-tracking-scaling-policy
```

Erstellen einer Skalierungsrichtlinie für Zielnachverfolgung für Application Auto Scaling mit Metrikberechnungen

Mithilfe metrischer Mathematik können Sie mehrere CloudWatch Metriken abfragen und mathematische Ausdrücke verwenden, um neue Zeitreihen auf der Grundlage dieser Metriken zu erstellen. Sie können die resultierenden Zeitreihen in der CloudWatch -Konsole visualisieren und sie Dashboards hinzufügen. Weitere Informationen zur metrischen Mathematik finden Sie unter Verwenden von metrischer Mathematik im CloudWatch Amazon-Benutzerhandbuch.

Für metrische mathematische Ausdrücke gelten folgende Überlegungen:

- Sie können jede verfügbare CloudWatch Metrik abfragen. Jede Metrik ist eine eindeutige Kombination aus Metrikname, Namespace und null oder mehr Dimensionen.
- Sie können einen beliebigen arithmetischen Operator (+ - */^), jede statistische Funktion (wie AVG oder SUM) oder eine andere Funktion verwenden, die diese CloudWatch Funktion unterstützt.
- Sie können sowohl Metriken als auch die Ergebnisse anderer mathematischer Ausdrücke in den Formeln des mathematischen Ausdrucks verwenden.
- Alle Ausdrücke, die in einer metrischen Spezifikation verwendet werden, müssen letztendlich eine einzige Zeitreihe ergeben.

- Sie können überprüfen, ob ein metrischer mathematischer Ausdruck gültig ist, indem Sie die CloudWatch Konsole oder die CloudWatch GetMetricDataAPI verwenden.

Themen

Beispiel: Amazon-SQS-Warteschlangenrückstand pro Aufgabe

Um den Rückstand der Amazon SQS-Warteschlange pro Aufgabe zu berechnen, nehmen Sie die ungefähre Anzahl der Nachrichten, die für den Abruf aus der Warteschlange zur Verfügung stehen, und teilen Sie diese Zahl durch die Anzahl der im Service laufenden Amazon ECS-Aufgaben. Weitere Informationen finden Sie im AWS Compute-Blog unter Amazon Elastic Container Service (ECS) Auto Scaling using custom metrics.

Die Logik für den Ausdruck lautet wie folgt:

```
sum of (number of messages in the queue)/(number of tasks that are
currently in the RUNNING state)
```

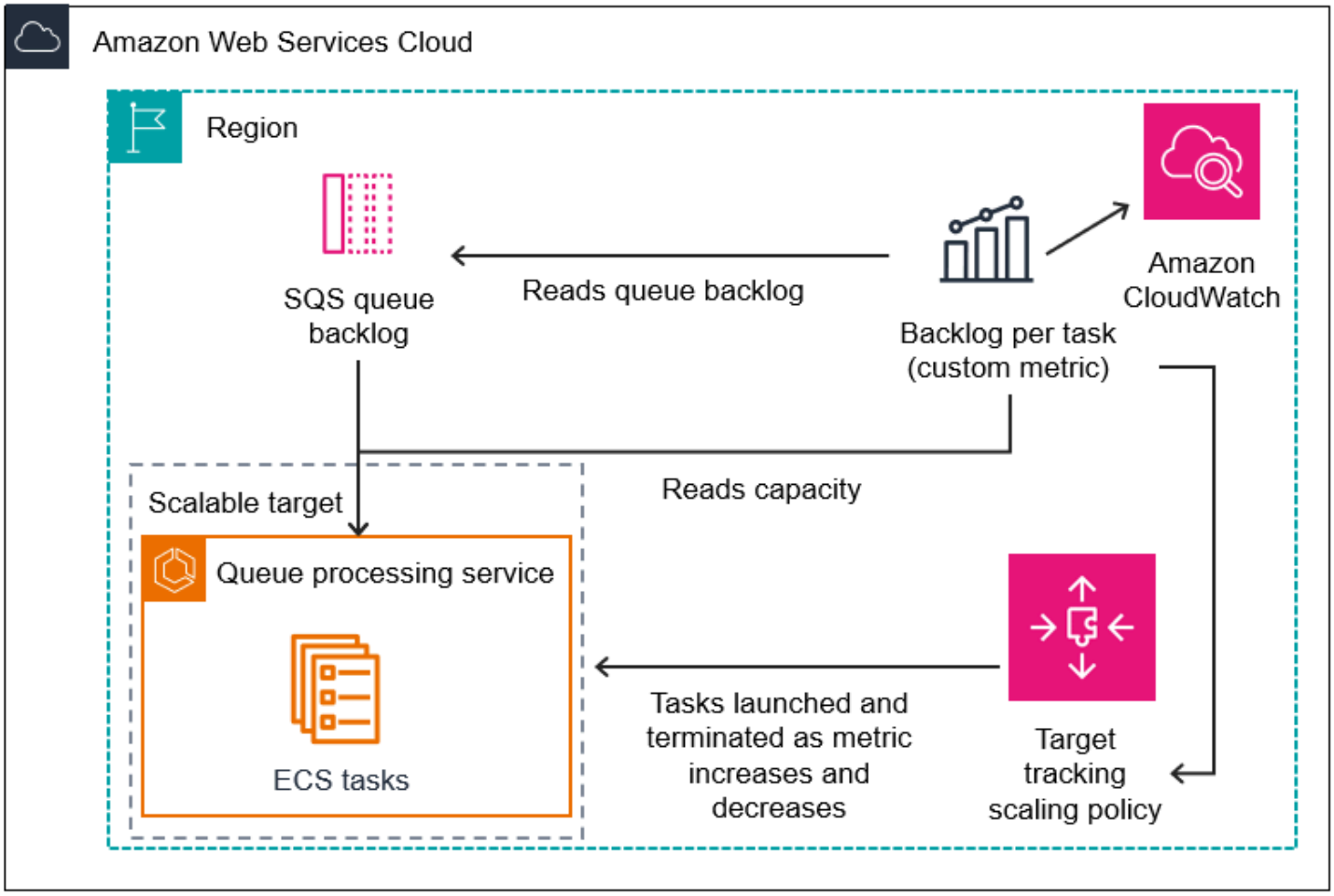
Dann lauten Ihre CloudWatch Metrikinformationen wie folgt.

ID (ID)	CloudWatch Metrik	Statistik	Zeitraum
m1	ApproximateNumberOfMessagesVisible	Summe	1 Minute
m2	RunningTaskCount	Durchschnitt	1 Minute

ID und Ausdruck Ihrer Metrikberechnung lauten wie folgt.

ID (ID)	Expression
e1	`(m1)/(m2)`

Das folgende Diagramm veranschaulicht die Architektur dieser Metrik:



So erstellen Sie mithilfe dieser Metrikberechnung eine Skalierungsrichtlinie für die Zielnachverfolgung (AWS CLI)

1. Speichern Sie den metrischen mathematischen Ausdruck als Teil einer benutzerdefinierten Metrikspezifikation in einer JSON-Datei namens `config.json`.

   Das folgende Beispiel hilft Ihnen bei den ersten Schritten. Ersetzen Sie jeden *`user input placeholder`* durch Ihre Informationen.

```
{
    "CustomizedMetricSpecification": {
        "Metrics": [
            {
                "Label": "Get the queue size (the number of messages waiting to be
 processed)",
                "Id": "m1",
                "MetricStat": {
```

```
                    "Metric": {
                        "MetricName": "ApproximateNumberOfMessagesVisible",
                        "Namespace": "AWS/SQS",
                        "Dimensions": [
                            {
                                "Name": "QueueName",
                                "Value": "my-queue"
                            }
                        ]
                    },
                    "Stat": "Sum"
                },
                "ReturnData": false
            },
            {
                "Label": "Get the ECS running task count (the number of currently
running tasks)",
                "Id": "m2",
                "MetricStat": {
                    "Metric": {
                        "MetricName": "RunningTaskCount",
                        "Namespace": "ECS/ContainerInsights",
                        "Dimensions": [
                            {
                                "Name": "ClusterName",
                                "Value": "my-cluster"
                            },
                            {
                                "Name": "ServiceName",
                                "Value": "my-service"
                            }
                        ]
                    },
                    "Stat": "Average"
                },
                "ReturnData": false
            },
            {
                "Label": "Calculate the backlog per instance",
                "Id": "e1",
                "Expression": "m1 / m2",
                "ReturnData": true
            }
        ]
```

```
    },
    "TargetValue": 100
}
```

Weitere Informationen finden Sie TargetTrackingScalingPolicyConfigurationin der API-Referenz für Application Auto Scaling.

> **Note**
>
> Im Folgenden finden Sie einige zusätzliche Ressourcen, die Ihnen bei der Suche nach Metriknamen, Namespaces, Dimensionen und Statistiken für Metriken helfen können: CloudWatch
>
> - Informationen zu den verfügbaren Metriken für AWS Services finden Sie im CloudWatch Amazon-Benutzerhandbuch unter AWS Services, die CloudWatch Metriken veröffentlichen.
> - Den genauen Metriknamen, den Namespace und die Dimensionen (falls zutreffend) für eine CloudWatch Metrik mit dem finden Sie unter AWS CLIlist-metrics.

2. Um diese Richtlinie zu erstellen, führen Sie den put-scaling-policyBefehl mit der JSON-Datei als Eingabe aus, wie im folgenden Beispiel gezeigt.

```
aws application-autoscaling put-scaling-policy --policy-name sqs-backlog-target-
tracking-scaling-policy \
  --service-namespace ecs --scalable-dimension ecs:service:DesiredCount --resource-
id service/my-cluster/my-service \
  --policy-type TargetTrackingScaling --target-tracking-scaling-policy-
configuration file://config.json
```

Bei Erfolg gibt dieser Befehl den Amazon-Ressourcennamen (ARN) der Richtlinie und den ARNs der beiden in Ihrem Namen erstellten CloudWatch Alarme zurück.

```
{
    "PolicyARN": "arn:aws:autoscaling:us-west-2:012345678910:scalingPolicy:
 8784a896-b2ba-47a1-b08c-27301cc499a1:resource/ecs/service/my-cluster/my-
service:policyName/sqs-backlog-target-tracking-scaling-policy",
    "Alarms": [
        {
```

```
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/my-cluster/my-service-
AlarmHigh-9bc77b56-0571-4276-ba0f-d4178882e0a0",
            "AlarmName": "TargetTracking-service/my-cluster/my-service-
AlarmHigh-9bc77b56-0571-4276-ba0f-d4178882e0a0"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/my-cluster/my-service-
AlarmLow-9b6ad934-6d37-438e-9e05-02836ddcbdc4",
            "AlarmName": "TargetTracking-service/my-cluster/my-service-
AlarmLow-9b6ad934-6d37-438e-9e05-02836ddcbdc4"
        }
    ]
}
```

> **Note**
>
> Wenn dieser Befehl einen Fehler auslöst, stellen Sie sicher, dass Sie die AWS CLI lokale Version auf die neueste Version aktualisiert haben.

Einschränkungen

- Die maximale Größe der Anfrage ist 50 KB. Dies ist die gesamte Payload-Größe für die PutScalingPolicyAPI-Anfrage, wenn Sie metrische Mathematik in der Richtliniendefinition verwenden. Wenn Sie diese Grenze überschreiten, lehnt Application Auto Scaling die Anfrage ab.
- Die folgenden Dienste werden bei Verwendung der metrischen Mathematik mit Skalierungsrichtlinien für die Zielverfolgung nicht unterstützt:
  - Amazon Keyspaces (für Apache Cassandra)
  - DynamoDB
  - Amazon EMR
  - Amazon MSK
  - Amazon Neptune

Stufenskalierungsrichtlinien für Application Auto Scaling

Eine schrittweise Skalierungsrichtlinie skaliert die Kapazität Ihrer Anwendung in vordefinierten Schritten auf der Grundlage von CloudWatch Alarmen. Sie können separate Skalierungsrichtlinien definieren, um die Aufskalierung (Erhöhung der Kapazität) und die Abskalierung (Verringerung der Kapazität) zu handhaben, wenn ein Alarmschwellenwert überschritten wird.

Mit Richtlinien zur schrittweisen Skalierung erstellen und verwalten Sie die CloudWatch Alarme, die den Skalierungsprozess auslösen. Wenn ein Alarm ausgelöst wird, initiiert Application Auto Scaling die mit diesem Alarm verbundene Skalierungsrichtlinie.

Die Richtlinie zur schrittweisen Skalierung skaliert die Kapazität anhand einer Reihe von Anpassungen, die als schrittweise Anpassungen bezeichnet werden. Die Größe der Anpassung richtet sich nach dem Ausmaß der Alarmüberschreitung.

- Wenn der Verstoß den ersten Schwellenwert überschreitet, wendet Application Auto Scaling die erste schrittweise Anpassung an.
- Wenn der Verstoß den zweiten Schwellenwert überschreitet, wendet Application Auto Scaling die zweite schrittweise Anpassung an, und so weiter.

Auf diese Weise kann die Skalierungsrichtlinie sowohl auf kleinere als auch auf größere Änderungen der Alarmmetrik angemessen reagieren.

Die Richtlinie reagiert auch während einer laufenden Skalierungsaktivität auf weitere Alarmverstöße. Das bedeutet, dass Application Auto Scaling alle Alarmverstöße auswertet, sobald sie auftreten. Eine Ruhephase dient zum Schutz vor zu hoher Skalierung aufgrund mehrerer schnell aufeinanderfolgender Alarmverstöße.

Wie die Ziel-Nachverfolgung kann auch die schrittweise Skalierung dazu beitragen, die Kapazität Ihrer Anwendung automatisch zu skalieren, wenn sich der Datenverkehr ändert. Richtlinien für die Ziel-Nachverfolgung sind jedoch einfacher zu implementieren und zu verwalten, wenn eine stetige Skalierung erforderlich ist.

Unterstützte skalierbare Ziele

Sie können Richtlinien zur schrittweisen Skalierung mit den folgenden skalierbaren Zielen verwenden:

- WorkSpaces Anwendungsflotten

- Aurora-DB-Cluster
- ECS-Services
- EMR-Cluster
- SageMaker Varianten von KI-Endpunkten
- SageMaker Komponenten der KI-Inferenz
- SageMaker Serverlos bereitgestellte Parallelität mit KI
- Spot Flotten
- Benutzerdefinierte Ressourcen

Inhalt

So funktioniert Step Scaling für Application Auto Scaling

In diesem Thema wird beschrieben, wie die schrittweise Skalierung funktioniert, und es werden die wichtigsten Elemente einer schrittweisen Skalierungsrichtlinie vorgestellt.

Inhalt

Funktionsweise

Um Step Scaling zu verwenden, erstellen Sie einen CloudWatch Alarm, der eine Metrik für Ihr skalierbares Ziel überwacht. Sie definieren die Metrik, den Schwellenwert und die Anzahl der Bewertungszeiträume, die einen Alarmverstoß bestimmen. Außerdem erstellen Sie eine Richtlinie zur schrittweisen Skalierung, die definiert, wie die Kapazität skaliert werden soll, wenn der Alarmschwellenwert überschritten wird, und verknüpfen sie mit Ihrem skalierbaren Ziel.

Sie fügen die schrittweisen Anpassungen in der Richtlinie hinzu. Sie können verschiedene schrittweise Anpassungen basierend auf der Größe der Alarmüberschreitung definieren. Beispiel:

- Aufskalierung um 10 Kapazitätseinheiten, wenn die Alarmmetrik 60 Prozent erreicht
- Aufskalierung um 30 Kapazitätseinheiten, wenn die Alarmmetrik 75 Prozent erreicht
- Aufskalierung um 40 Kapazitätseinheiten, wenn die Alarmmetrik 85 Prozent erreicht

Wenn der Alarmschwellenwert für die angegebene Anzahl von Auswertungszeiträumen überschritten wird, wendet Application Auto Scaling die in der Richtlinie definierten schrittweisen Anpassungen an. Die Anpassungen können bei weiteren Überschreitungen des Alarms fortgesetzt werden, bis der Alarmstatus `OK` wieder erreicht ist.

Zwischen den Skalierungsaktivitäten liegen Ruhephasen, um schnelle Kapazitätsschwankungen zu vermeiden. Sie können die Ruhephasen für Ihre Richtlinie optional konfigurieren.

Schrittweise Anpassungen

Wenn Sie eine Richtlinie zur schrittweise Skalierung erstellen, geben Sie eine oder mehrere Stufenanpassungen an, die automatisch die Kapazität des Ziels dynamisch basierend auf der Größe der Alarmüberschreitung skalieren. Jede Schrittanpassung gibt Folgendes an:

- Eine Untergrenze für den Metrikwert
- Eine Obergrenze für den Metrikwert
- Den Skalierungswert basierend auf dem Skalierungsanpassungstyp

CloudWatch aggregiert metrische Datenpunkte auf der Grundlage der Statistik für die mit Ihrem CloudWatch Alarm verknüpfte Metrik. Wenn der Alarm ausgelöst wird, wird die entsprechende Skalierungsrichtlinie ausgelöst. Application Auto Scaling wendet den angegebenen Aggregationstyp auf die neuesten metrischen Datenpunkte von an CloudWatch (im Gegensatz zu den metrischen

Rohdaten). Dieser aggregierte Metrikwert wird anschließend mit der Ober- und der Untergrenze verglichen, die durch die Schrittanpassungen definiert wurden. Dadurch wird ermittelt, welche Schrittanpassung auszuführen ist.

Sie geben die Ober- und Untergrenzen relativ zum Verletzungsschwellenwert an. Nehmen wir zum Beispiel an, Sie haben einen CloudWatch Alarm ausgelöst und eine Scale-Out-Richtlinie für den Fall festgelegt, dass die Metrik über 50 Prozent liegt. Dann haben Sie einen zweiten Alarm und eine Abskalierungsrichtlinie für den Fall erstellt, dass die Metrik unter 50 Prozent liegt. Sie haben eine Reihe von schrittweisen Anpassungen mit dem Anpassungstyp `PercentChangeInCapacity` für jede Richtlinie vorgenommen:

Beispiel: Schrittanpassungen für die Richtlinie zur horizontalen Skalierung nach oben

Untergrenze	Obergrenze	Anpassung
0	10	0
10	20	10
20	Null	30

Beispiel: Schrittanpassungen für die Richtlinie zur horizontalen Skalierung nach unten

Untergrenze	Obergrenze	Anpassung
-10	0	0
-20	-10	-10
Null	-20	-30

Dadurch wird die folgende Skalierungskonfiguration erstellt.

```
Metric value

-infinity          30%    40%          60%     70%             infinity
-------------------------------------------------------------------------
          -30%      | -10% | Unchanged  | +10%  |       +30%
-------------------------------------------------------------------------
```

Angenommen, Sie verwenden diese Skalierungskonfiguration für ein skalierbares Ziel mit einer Kapazität von 10. Die folgenden Punkte fassen das Verhalten der Skalierungskonfiguration in Bezug auf die Kapazität des skalierbaren Ziels zusammen:

- Die ursprüngliche Kapazität wird aufrechterhalten, solange der aggregierte Metrikwert größer als 40 und kleiner als 60 ist.
- Wenn der Metrikwert 60 erreicht, erhöht Application Auto Scaling die Kapazität des skalierbaren Ziels um 1 auf 11. Dies basiert auf der zweiten Schrittanpassung der Richtlinie für die horizontale Skalierung nach oben (Erhöhen um 10 Prozent von 10). Nachdem die neue Kapazität hinzugefügt wurde, erhöht Application Auto Scaling die aktuelle Kapazität auf 11. Steigt der metrische Wert auch nach dieser Kapazitätserhöhung auf 70, erhöht Application Auto Scaling die Zielkapazität um 3 auf 14. Dies basiert auf der dritten Schrittanpassung der Richtlinie für die horizontale Skalierung nach oben (Erhöhen um 30 Prozent von 11, 3,3 abgerundet auf 3).
- Wenn der Metrikwert 40 erreicht, verringert Application Auto Scaling die Kapazität des skalierbaren Ziels um 1 auf 13, basierend auf der zweiten Anpassungsstufe der Scale-in-Richtlinie (Entfernen von 10 Prozent von 14, 1,4, abgerundet auf 1). Wenn der metrische Wert auch nach dieser Kapazitätsverringerung auf 30 fällt, verringert Application Auto Scaling die Zielkapazität um 3 auf 10, basierend auf der dritten Anpassungsstufe der Scale-in-Richtlinie (Entfernen von 30 Prozent von 13, 3,9, abgerundet auf 3).

Wenn Sie die Schrittanpassungen für Ihre Skalierungsrichtlinie angeben, beachten Sie Folgendes:

- Die Bereiche der Schrittanpassungen dürfen sich nicht überschneiden oder Lücken aufweisen.
- Nur eine Schrittanpassung darf über einen Nullwert als Untergrenze verfügen (negative Unendlichkeit). Verfügt eine Schrittanpassung über eine negative Untergrenze, muss eine Schrittanpassung mit einem Nullwert als Untergrenze vorhanden sein.
- Nur eine Schrittanpassung darf über einen Nullwert als Obergrenze verfügen (positive Unendlichkeit). Verfügt eine Schrittanpassung über eine positive Obergrenze, muss eine Schrittanpassung mit einem Nullwert als Obergrenze vorhanden sein.
- Ober- und Untergrenze einer Schrittanpassung können nicht gleichzeitig über einen Nullwert verfügen.
- Liegt der Metrikwert oberhalb des Verletzungsschwellenwerts, wird die Untergrenze eingeschlossen und die Obergrenze ausgeschlossen. Liegt der Metrikwert unterhalb des Verletzungsschwellenwerts, wird die Untergrenze ausgeschlossen und die Obergrenze eingeschlossen.

Skalierungsanpassungstypen

Sie können eine Skalierungsrichtlinie definieren, welche die optimale Skalierungsaktion basierend auf dem von Ihnen gewählten Skalierungsanpassungstyp ausführt. Sie können den Anpassungstyp als Prozentsatz der aktuellen Kapazität Ihres skalierbaren Ziels oder in absoluten Zahlen angeben.

Application Auto Scaling unterstützt die folgenden Anpassungstypen für Stufenskalierungsrichtlinien:

- ChangeInCapacity— Erhöht oder verringert die aktuelle Kapazität des skalierbaren Ziels um den angegebenen Wert. Ein positiver Wert erhöht die Kapazität, ein negativer Anpassungswert verringert die Kapazität. Ein Beispiel: Wenn die aktuelle Kapazität 3 ist und die Anpassung 5 beträgt, fügt Application Auto Scaling der Kapazität 5 hinzu, so dass sie insgesamt 8 beträgt.
- ExactCapacity— Ändert die aktuelle Kapazität des skalierbaren Ziels auf den angegebenen Wert. Geben Sie bei diesem Anpassungstyp einen nicht-negativen Wert an. Ein Beispiel: Wenn die aktuelle Kapazität 3 ist und die Anpassung 5 beträgt, ändert Application Auto Scaling die Kapazität auf 5.
- PercentChangeInCapacity— Erhöht oder verringert die aktuelle Kapazität des skalierbaren Ziels um den angegebenen Prozentsatz. Ein positiver Wert erhöht die Kapazität, ein negativer Anpassungswert verringert die Kapazität. Ein Beispiel: Wenn die aktuelle Kapazität 10 ist und die Anpassung 10 Prozent beträgt, fügt Application Auto Scaling 1 zur Kapazität hinzu, so dass sie insgesamt 11 beträgt.

  Wenn der resultierende Wert keine ganze Zahl ist, rundet Application Auto Scaling ihn wie folgt:

  - Werte größer als 1 werden abgerundet. Beispielsweise wird `12.7` auf `12` gerundet.
  - Werte zwischen 0 und 1 werden auf 1 gerundet. Beispielsweise wird `.67` auf `1` gerundet.
  - Werte zwischen 0 und -1 werden auf -1 gerundet. Beispielsweise wird `-.58` auf `-1` gerundet.
  - Werte kleiner als -1 werden aufgerundet. Beispielsweise wird `-6.67` auf `-6` gerundet.

  Mit PercentChangeInCapacitykönnen Sie auch den Mindestbetrag für die Skalierung mithilfe des `MinAdjustmentMagnitude` Parameters angeben. Angenommen, Sie erstellen eine Richtlinie zum Hinzufügen von 25 Prozent und geben an, dass mindestens 2 hinzugefügt werden sollen. Hat das skalierbare Ziel eine Kapazität von 4 und wird die Skalierungsrichtlinie ausgeführt, ist 25 Prozent von 4 gleich 1. Da Sie jedoch eine Mindestschrittweite von 2 angegeben haben, fügt Application Auto Scaling 2 hinzu.

Ruhephase

In Ihrer Richtlinie für die schrittweise Skalierung können Sie optional eine Ruhephase definieren.

Eine Ruhephase ist die Zeitspanne, die die Skalierungsrichtlinie warten muss, bis eine vorherige Skalierungsaktivität wirksam wird.

Es gibt zwei Möglichkeiten, die Verwendung von Ruhephasen für eine Konfiguration mit schrittweiser Skalierung zu planen:

- Mit den Richtlinien zur Ruhephase der Aufskalierung wird beabsichtigt, kontinuierlich (aber nicht übermäßig) aufzuskalieren. Nachdem Application Auto Scaling unter Verwendung einer Skalierungsrichtlinie erfolgreich aufskaliert wurde, wird die Berechnung der Ruhezeit gestartet. Eine Skalierungsrichtlinie erhöht die gewünschte Kapazität nicht erneut, es sei denn, es wird eine größere Aufskalierung ausgelöst oder die Ruhephase endet. Während die Scale-Out-Ruhephase wirksam ist, wird die durch die initiierende horizontale Skalierung nach oben (Scale-Out) hinzugefügte Kapazität als Teil der gewünschten Kapazität für die nächste horizontale Skalierung nach oben berechnet.
- Mit den Richtlinien der Ruhephase für die Abskalierung ist beabsichtigt, die Abskalierung konservativ durchzuführen, um die Verfügbarkeit Ihrer Anwendung zu schützen, sodass Abskalierungsaktivitäten blockiert werden, bis die Ruhephase für die Abskalierung abgelaufen ist. Wenn jedoch ein anderer Alarm während der Abkühlphase nach einer Abskalier-Aktivität eine Aufskalier-Aktivität auslöst, wird das Ziel durch Application Auto Scaling sofort abskaliert. In diesem Fall wird die Ruhephase für die Abskalierung angehalten und nicht abgeschlossen.

Wenn beispielsweise eine Spitze im Datenverkehr auftritt, wird ein Alarm ausgelöst und Application Auto Scaling fügt automatisch Kapazität hinzu, um die erhöhte Last zu bewältigen. Wenn Sie eine Ruhephase für Ihre Richtlinie für Aufskalierung festlegen und der Alarm die Richtlinie auslöst, um die Kapazität um 2 zu erhöhen, wird die Skalierung erfolgreich abgeschlossen und die Ruhephase für die Aufskalierung beginnt. Wenn ein Alarm die gleiche Richtlinie, aber mit einer aggressiveren Stufenanpassung, z. B. um 3, während der Ruhephase erneut auslöst, wird die vorherige Erhöhung um 2 als Teil der aktuellen Kapazität betrachtet. Daher wird der Kapazität nur 1 hinzugefügt. Dies ermöglicht eine schnellere Skalierung als das Warten auf den Ablauf der Ruhephase, ohne dass Sie mehr Kapazität hinzufügen, als Sie benötigen.

Die Ruhephase wird in Sekunden gemessen und gilt nur für Skalierungsrichtlinien-bezogene Skalierungen. Wenn eine geplante Aktion während einer Ruhephase zum geplanten Zeitpunkt

beginnt, kann sie umgehend eine Skalierung auslösen, ohne das Ablaufen der Ruhephase abzuwarten.

Der Standardwert ist 300, wenn kein Wert angegeben wird.

Häufig verwendete Befehle zur Erstellung, Verwaltung und Löschung von Skalierungsrichtlinien

Zu den häufig verwendeten Befehlen für die Arbeit mit Skalierungsrichtlinien gehören:

- register-scalable-targetum Ressourcen als skalierbare Ziele zu registrieren AWS oder anzupassen (eine Ressource, die Application Auto Scaling skalieren kann) und die Skalierung auszusetzen und wieder aufzunehmen.
- put-scaling-policyum Skalierungsrichtlinien für ein vorhandenes skalierbares Ziel hinzuzufügen oder zu ändern.
- describe-scaling-activitiesum Informationen über Skalierungsaktivitäten in einer AWS Region zurückzugeben.
- describe-scaling-policiesum Informationen über Skalierungsrichtlinien in einer AWS Region zurückzugeben.
- delete-scaling-policyum eine Skalierungsrichtlinie zu löschen.

Überlegungen

Bei der Arbeit mit Richtlinien zur schrittweisen Skalierung ist Folgendes zu beachten:

- Überlegen Sie, ob Sie die Schrittanpassungen in der Anwendung genau genug vorhersagen können, um die schrittweise Skalierung zu verwenden. Wenn Ihre Skalierungsmetrik die Kapazität des skalierbaren Ziels proportional vergrößert oder verkleinert, raten wir stattdessen zur Verwendung einer Skalierungsrichtlinie für die Ziel-Nachverfolgung. Sie haben weiterhin die Möglichkeit, die Schrittskalierung als zusätzliche Richtlinie für eine erweiterte Konfiguration zu verwenden. Beispiel: Sie können eine striktere Antwort konfigurieren, sobald die Auslastung ein bestimmtes Niveau erreicht.
- Achten Sie darauf, einen angemessenen Abstand zwischen den Schwellenwerten für Scale-Out und Scale-In zu wählen, um ein Flattern zu verhindern. Flattern beschreibt eine Endlosschleife aus Auf- und Abwärtsskalieren. Das heißt, wenn eine Skalierungsaktion durchgeführt wird, würde sich der Metrikwert ändern und eine weitere Skalierungsaktion in der umgekehrten Richtung starten.

Zugehörige Ressourcen

Informationen zum Erstellen von Richtlinien zur schrittweisen Skalierung für Auto Scaling-Gruppen finden Sie unter Schrittweise und einfache Skalierungsrichtlinien für Amazon EC2 Auto Scaling im Benutzerhandbuch zu Amazon EC2 Auto Scaling.

Konsolenzugriff

Der Konsolenzugriff zum Anzeigen, Hinzufügen, Aktualisieren oder Entfernen von Richtlinien zur schrittweisen Skalierung für die Ziel-Nachverfolgung auf skalierbaren Ressourcen hängt von der verwendeten Ressource ab. Weitere Informationen finden Sie unter AWS-Services die Sie mit Application Auto Scaling verwenden können.

Erstellen Sie eine Step Scaling-Richtlinie für Application Auto Scaling mit dem AWS CLI

In diesem Beispiel AWS CLI werden Befehle verwendet, um eine schrittweise Skalierungsrichtlinie für einen Amazon ECS-Service zu erstellen. Geben Sie für ein anderes skalierbares Ziel seinen Namespace in`--service-namespace`, seine skalierbare Dimension in `--scalable-dimension` und seine Ressourcen-ID in `--resource-id` an.

Denken Sie bei der Verwendung von daran AWS CLI, dass Ihre Befehle in der für Ihr Profil AWS-Region konfigurierten Version ausgeführt werden. Wenn Sie die Befehle in einer anderen Region ausführen möchten, ändern Sie entweder die Standardregion für Ihr Profil, oder verwenden Sie den `--region`-Parameter mit dem Befehl.

Aufgaben

- Schritt 1: Registrieren Sie ein skalierbares Ziel
- Schritt 2: Erstellen Sie eine Richtlinie zur schrittweisen Skalierung
- Schritt 3: Erstellen Sie einen Alarm, der eine Skalierungsrichtlinie aufruft

Schritt 1: Registrieren Sie ein skalierbares Ziel

Wenn Sie dies noch nicht getan haben, registrieren Sie das skalierbare Ziel. Verwenden Sie den register-scalable-targetBefehl, um eine bestimmte Ressource im Zieldienst als skalierbares Ziel zu registrieren. Im folgenden Beispiel wird ein Amazon ECS-Service mit Application Auto Scaling

registriert. Application Auto Scaling kann die Anzahl der Aufgaben auf ein Minimum von 2 und ein Maximum von 10 skalieren. Ersetzen Sie jeden *user input placeholder* durch Ihre Informationen.

Linux, macOS oder Unix

```
aws application-autoscaling register-scalable-target --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service \
  --min-capacity 2 --max-capacity 10
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace ecs ^
  --scalable-dimension ecs:service:DesiredCount ^
  --resource-id service/my-cluster/my-service ^
  --min-capacity 2 --max-capacity 10
```

Ausgabe

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück. Es folgt eine Beispielausgabe.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Schritt 2: Erstellen Sie eine Richtlinie zur schrittweisen Skalierung

Um eine Richtlinie zur schrittweisen Skalierung für Ihr skalierbares Ziel zu erstellen, können Sie die folgenden Beispiele verwenden, um Ihnen den Einstieg zu erleichtern.

Scale out

So erstellen Sie eine Richtlinie zur schrittweisen Skalierung für Scale-Out (Kapazitätserhöhung)

1. Verwenden Sie den folgenden `cat` Befehl, um eine Konfiguration einer Step Scaling-Richtlinie in einer JSON-Datei mit dem Namen `config.json` in Ihrem Home-Verzeichnis zu speichern. Im Folgenden finden Sie eine Beispielkonfiguration mit einem Anpassungstyp von`PercentChangeInCapacity`, die die Kapazität des skalierbaren Ziels auf der

Grundlage der folgenden schrittweisen Anpassungen erhöht (unter der Annahme eines CloudWatch Alarmschwellenwerts von 70):

- Erhöhen Sie die Kapazität um 10 Prozent, wenn der Wert der Metrik größer oder gleich 70, aber kleiner als 85 ist
- Erhöhen Sie die Kapazität um 20 Prozent, wenn der Wert der Metrik größer oder gleich 85, aber kleiner als 95 ist
- Erhöhen Sie die Kapazität um 30 Prozent, wenn der Wert der Metrik größer oder gleich 95 ist

```
$ cat ~/config.json
{
  "AdjustmentType": "PercentChangeInCapacity",
  "MetricAggregationType": "Average",
  "Cooldown": 60,
  "MinAdjustmentMagnitude": 1,
  "StepAdjustments": [
    {
      "MetricIntervalLowerBound": 0.0,
      "MetricIntervalUpperBound": 15.0,
      "ScalingAdjustment": 10
    },
    {
      "MetricIntervalLowerBound": 15.0,
      "MetricIntervalUpperBound": 25.0,
      "ScalingAdjustment": 20
    },
    {
      "MetricIntervalLowerBound": 25.0,
      "ScalingAdjustment": 30
    }
  ]
}
```

Weitere Informationen finden Sie StepScalingPolicyConfigurationin der API-Referenz für Application Auto Scaling.

2. Verwenden Sie den folgenden put-scaling-policyBefehl zusammen mit der `config.json` Datei, die Sie erstellt haben, um eine Skalierungsrichtlinie mit dem Namen zu erstellen`my-step-scaling-policy`.

Linux, macOS oder Unix

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service \
  --policy-name my-step-scaling-policy --policy-type StepScaling \
  --step-scaling-policy-configuration file://config.json
```

Windows

```
aws application-autoscaling put-scaling-policy --service-namespace ecs ^
  --scalable-dimension ecs:service:DesiredCount ^
  --resource-id service/my-cluster/my-service ^
  --policy-name my-step-scaling-policy --policy-type StepScaling ^
  --step-scaling-policy-configuration file://config.json
```

Ausgabe

Die Ausgabe enthält den ARN, der als eindeutiger Name für die Richtlinie dient. Sie benötigen ihn, um einen CloudWatch Alarm für Ihre Richtlinie zu erstellen. Es folgt eine Beispielausgabe.

```
{
    "PolicyARN":
 "arn:aws:autoscaling:region:123456789012:scalingPolicy:ac542982-cbeb-4294-891c-a5a941dfa787:resource/ecs/service/my-cluster/my-service:policyName/my-step-scaling-policy"
}
```

Scale in

Um eine schrittweise Skalierungsrichtlinie für die Skalierung (Kapazitätsreduzierung) zu erstellen

1. Verwenden Sie den folgenden `cat` Befehl, um eine Konfiguration einer schrittweisen Skalierungsrichtlinie in einer JSON-Datei mit dem Namen `config.json` in Ihrem Home-Verzeichnis zu speichern. Im Folgenden finden Sie eine Beispielkonfiguration mit einem Anpassungstyp von`ChangeInCapacity`, der die Kapazität des skalierbaren Ziels auf der

Grundlage der folgenden schrittweisen Anpassungen verringert (unter der Annahme eines CloudWatch Alarmschwellenwerts von 50):

- Verringern Sie die Kapazität um 1, wenn der Wert der Metrik kleiner oder gleich 50, aber größer als 40 ist
- Verringern Sie die Kapazität um 2, wenn der Wert der Metrik kleiner oder gleich 40, aber größer als 30 ist
- Verringern Sie die Kapazität um 3, wenn der Wert der Metrik kleiner oder gleich 30 ist

```
$ cat ~/config.json
{
  "AdjustmentType": "ChangeInCapacity",
  "MetricAggregationType": "Average",
  "Cooldown": 60,
  "StepAdjustments": [
    {
      "MetricIntervalUpperBound": 0.0,
      "MetricIntervalLowerBound": -10.0,
      "ScalingAdjustment": -1
    },
    {
      "MetricIntervalUpperBound": -10.0,
      "MetricIntervalLowerBound": -20.0,
      "ScalingAdjustment": -2
    },
    {
      "MetricIntervalUpperBound": -20.0,
      "ScalingAdjustment": -3
    }
  ]
}
```

Weitere Informationen finden Sie StepScalingPolicyConfigurationin der API-Referenz für Application Auto Scaling.

2. Verwenden Sie den folgenden put-scaling-policyBefehl zusammen mit der `config.json` Datei, die Sie erstellt haben, um eine Skalierungsrichtlinie mit dem Namen zu erstellen`my-step-scaling-policy`.

   Linux, macOS oder Unix

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service \
  --policy-name my-step-scaling-policy --policy-type StepScaling \
  --step-scaling-policy-configuration file://config.json
```

Windows

```
aws application-autoscaling put-scaling-policy --service-namespace ecs ^
  --scalable-dimension ecs:service:DesiredCount ^
  --resource-id service/my-cluster/my-service ^
  --policy-name my-step-scaling-policy --policy-type StepScaling ^
  --step-scaling-policy-configuration file://config.json
```

Ausgabe

Die Ausgabe enthält den ARN, der als eindeutiger Name für die Richtlinie dient. Sie benötigen diesen ARN, um einen CloudWatch Alarm für Ihre Richtlinie zu erstellen. Es folgt eine Beispielausgabe.

```
{
    "PolicyARN":
 "arn:aws:autoscaling:region:123456789012:scalingPolicy:ac542982-cbeb-4294-891c-a5a941dfa787:resource/ecs/service/my-cluster/my-service:policyName/my-step-scaling-policy"
}
```

Schritt 3: Erstellen Sie einen Alarm, der eine Skalierungsrichtlinie aufruft

Verwenden Sie abschließend den folgenden CloudWatch put-metric-alarmBefehl, um einen Alarm zu erstellen, der mit Ihrer schrittweisen Skalierungsrichtlinie verwendet werden kann. In diesem Beispiel haben Sie einen Alarm, der auf der durchschnittlichen CPU-Auslastung basiert. Der Alarm ist so konfiguriert, dass er sich in einem ALARM-Zustand befindet, wenn er für mindestens zwei aufeinanderfolgende Auswerteperioden von 60 Sekunden einen Schwellenwert von 70 Prozent erreicht. Um eine andere CloudWatch Metrik anzugeben oder Ihre eigene benutzerdefinierte Metrik zu verwenden, geben Sie ihren Namen `--metric-name` und ihren Namespace in `--namespace` an.

Linux, macOS oder Unix

```
aws cloudwatch put-metric-alarm --alarm-name Step-Scaling-AlarmHigh-ECS:service/my-
cluster/my-service \
  --metric-name CPUUtilization --namespace AWS/ECS --statistic Average \
  --period 60 --evaluation-periods 2 --threshold 70 \
  --comparison-operator GreaterThanOrEqualToThreshold \
  --dimensions Name=ClusterName,Value=default Name=ServiceName,Value=sample-app-service
 \
  --alarm-actions PolicyARN
```

Windows

```
aws cloudwatch put-metric-alarm --alarm-name Step-Scaling-AlarmHigh-ECS:service/my-
cluster/my-service ^
  --metric-name CPUUtilization --namespace AWS/ECS --statistic Average ^
  --period 60 --evaluation-periods 2 --threshold 70 ^
  --comparison-operator GreaterThanOrEqualToThreshold ^
  --dimensions Name=ClusterName,Value=default Name=ServiceName,Value=sample-app-service
 ^
  --alarm-actions PolicyARN
```

Beschreiben Sie die schrittweisen Skalierungsrichtlinien für Application Auto Scaling mithilfe der AWS CLI

Mit dem describe-scaling-policiesBefehl können Sie alle Skalierungsrichtlinien für einen Service-Namespace beschreiben. Das folgende Beispiel beschreibt alle Skalierungsrichtlinien für alle Amazon ECS-Services. Um sie nur für einen bestimmten Amazon ECS-Service aufzulisten, fügen Sie die `--resource-id` Option hinzu.

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs
```

Sie können die Ergebnisse mithilfe des Parameters `--query` filtern, um nur die Richtlinien zur schrittweisen Skalierung zu erhalten. Weitere Informationen über die Syntax von `query` finden Sie unter Steuerung der Befehlsausgabe vom AWS CLI im AWS Command Line Interface Benutzerhandbuch.

Linux, macOS oder Unix

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs \
  --query 'ScalingPolicies[?PolicyType==`StepScaling`]'
```

Windows

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs ^
  --query "ScalingPolicies[?PolicyType==`StepScaling`]"
```

Ausgabe

Es folgt eine Beispielausgabe.

```
[
    {
        "PolicyARN": "PolicyARN",
        "StepScalingPolicyConfiguration": {
            "MetricAggregationType": "Average",
            "Cooldown": 60,
            "StepAdjustments": [
                {
                    "MetricIntervalLowerBound": 0.0,
                    "MetricIntervalUpperBound": 15.0,
                    "ScalingAdjustment": 1
                },
                {
                    "MetricIntervalLowerBound": 15.0,
                    "MetricIntervalUpperBound": 25.0,
                    "ScalingAdjustment": 2
                },
                {
                    "MetricIntervalLowerBound": 25.0,
                    "ScalingAdjustment": 3
                }
            ],
            "AdjustmentType": "ChangeInCapacity"
        },
        "PolicyType": "StepScaling",
        "ResourceId": "service/my-cluster/my-service",
        "ServiceNamespace": "ecs",
        "Alarms": [
            {
                "AlarmName": "Step-Scaling-AlarmHigh-ECS:service/my-cluster/my-
service",
```

```
                "AlarmARN": "arn:aws:cloudwatch:region:012345678910:alarm:Step-Scaling-
AlarmHigh-ECS:service/my-cluster/my-service"
            }
        ],
        "PolicyName": "my-step-scaling-policy",
        "ScalableDimension": "ecs:service:DesiredCount",
        "CreationTime": 1515024099.901
    }
]
```

Löschen Sie eine Step Scaling-Richtlinie für Application Auto Scaling mit dem AWS CLI

Wenn Sie keine Richtlinie für eine schrittweise Skalierung mehr benötigen, können Sie diese löschen. Führen Sie die folgenden Aufgaben aus, um sowohl die Skalierungsrichtlinie als auch den zugehörigen CloudWatch Alarm zu löschen.

So löschen Sie Ihre Skalierungsrichtlinie

Verwenden Sie den Befehl delete-scaling-policy.

Linux, macOS oder Unix

```
aws application-autoscaling delete-scaling-policy --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service \
  --policy-name my-step-scaling-policy
```

Windows

```
aws application-autoscaling delete-scaling-policy --service-namespace ecs ^
  --scalable-dimension ecs:service:DesiredCount ^
  --resource-id service/my-cluster/my-service ^
  --policy-name my-step-scaling-policy
```

Um den CloudWatch Alarm zu löschen

Verwenden Sie den delete-alarms-Befehl. Sie können einen oder mehrere Alarme gleichzeitig löschen. Sie können beispielsweise den folgenden Befehl verwenden, um die Alarme `Step-`

`Scaling-AlarmHigh-ECS:service/my-cluster/my-service` und `Step-Scaling-AlarmLow-ECS:service/my-cluster/my-service` zu löschen.

```
aws cloudwatch delete-alarms --alarm-name Step-Scaling-AlarmHigh-ECS:service/my-cluster/my-service Step-Scaling-AlarmLow-ECS:service/my-cluster/my-service
```

Prädiktive Skalierung für Application Auto Scaling

Predictive Scaling skaliert Ihre Anwendung proaktiv. Predictive Scaling analysiert historische Lastdaten, um tägliche oder wöchentliche Muster im Verkehrsfluss zu erkennen. Es verwendet diese Informationen, um den future Kapazitätsbedarf zu prognostizieren, um die Kapazität Ihrer Anwendung proaktiv an die erwartete Auslastung anzupassen.

Die prädiktive Skalierung eignet sich gut für Situationen, in denen Sie Folgendes haben:

- Zyklischen Datenverkehr, z. B. hohe Auslastung von Ressourcen während der normalen Geschäftszeiten und niedrige Auslastung von Ressourcen am Abend und am Wochenende
- Wiederkehrende on-and-off Workload-Muster, wie z. B. Stapelverarbeitung, Tests oder regelmäßige Datenanalysen.
- Anwendungen, die eine lange Zeit in Anspruch nehmen, was zu einer spürbaren Latenzauswirkung auf die Anwendungsleistung bei Aufskalierungsereignissen führt

Inhalt

So funktioniert die prädiktive Skalierung von Application Auto Scaling

Um prädiktive Skalierung zu verwenden, erstellen Sie eine Richtlinie für prädiktive Skalierung, die die zu überwachende und zu analysierende CloudWatch Metrik festlegt. Sie können eine vordefinierte Metrik oder eine benutzerdefinierte Metrik verwenden. Damit die prädiktive Skalierung mit der Prognose future Werte beginnen kann, muss diese Metrik Daten für mindestens 24 Stunden enthalten.

Nachdem Sie die Richtlinie erstellt haben, beginnt die prädiktive Skalierung mit der Analyse von Metrikdaten der letzten 14 Tage, um Muster zu identifizieren. Anhand dieser Analyse wird eine stündliche Prognose des Kapazitätsbedarfs für die nächsten 48 Stunden erstellt. Die Prognose wird

alle 6 Stunden anhand der neuesten CloudWatch Daten aktualisiert. Wenn neue Daten eintreffen, kann die prädiktive Skalierung die Genauigkeit zukünftiger Prognosen kontinuierlich verbessern.

Sie können die prädiktive Skalierung zunächst nur im Prognosemodus aktivieren. In diesem Modus werden Kapazitätsprognosen generiert, Ihre Kapazität wird jedoch nicht auf der Grundlage dieser Prognosen skaliert. Auf diese Weise können Sie die Genauigkeit und Eignung der Prognose bewerten.

Nachdem Sie die Prognosedaten überprüft und beschlossen haben, mit der Skalierung auf der Grundlage dieser Daten zu beginnen, schalten Sie die Skalierungsrichtlinie in den Prognose- und Skalierungsmodus um. In diesem Modus:

- Wenn in der Prognose ein Anstieg der Last erwartet wird, wird durch vorausschauende Skalierung die Kapazität erhöht.
- Wenn in der Prognose ein Rückgang der Auslastung erwartet wird, wird die vorausschauende Skalierung nicht so skaliert, dass Kapazität abgebaut wird. Dadurch wird sichergestellt, dass Sie nur dann skalieren, wenn die Nachfrage tatsächlich sinkt, und nicht nur aufgrund von Prognosen. Um Kapazität zu entfernen, die nicht mehr benötigt wird, müssen Sie eine Target Tracking- oder Step Scaling-Richtlinie erstellen, da diese auf Metrikdaten in Echtzeit reagieren.

Standardmäßig skaliert Predictive Scaling Ihre skalierbaren Ziele zu Beginn jeder Stunde auf der Grundlage der Prognose für diese Stunde. Sie können optional eine frühere Startzeit angeben, indem Sie die `SchedulingBufferTime` Eigenschaft im `PutScalingPolicy` API-Vorgang verwenden. Auf diese Weise können Sie die prognostizierte Kapazität vor dem prognostizierten Bedarf in Betrieb nehmen, sodass die neue Kapazität ausreichend Zeit hat, um den Verkehr abzuwickeln.

Maximale Kapazitätsgrenze

Standardmäßig können Skalierungsrichtlinien die Kapazität nicht über die maximale Kapazität hinaus erhöhen.

Alternativ können Sie zulassen, dass die maximale Kapazität des skalierbaren Ziels automatisch erhöht wird, wenn sich die prognostizierte Kapazität der maximalen Kapazität des skalierbaren Ziels nähert oder diese überschreitet. Um dieses Verhalten zu aktivieren, verwenden Sie die `MaxCapacityBuffer` Eigenschaften `MaxCapacityBreachBehavior` und im `PutScalingPolicy` API-Vorgang oder die Einstellung für das Verhalten „Max. Kapazität“ in AWS-Managementkonsole.

> **Warning**
>
> Seien Sie vorsichtig, wenn Sie zulassen, dass die maximale Kapazität automatisch erhöht wird. Die maximale Kapazität wird nicht automatisch auf das ursprüngliche Maximum zurückgesetzt.

Häufig verwendete Befehle zur Erstellung, Verwaltung und Löschung von Skalierungsrichtlinien

Zu den häufig verwendeten Befehlen für die Arbeit mit Richtlinien zur vorausschauenden Skalierung gehören:

- `register-scalable-target`um Ressourcen als skalierbare Ziele zu registrieren AWS oder anzupassen, die Skalierung auszusetzen und die Skalierung wieder aufzunehmen.
- `put-scaling-policy`um eine prädiktive Skalierungsrichtlinie zu erstellen.
- `get-predictive-scaling-forecast`um die Prognosedaten für eine prädiktive Skalierungsrichtlinie abzurufen.
- `describe-scaling-activities`um Informationen über Skalierungsaktivitäten in einem AWS-Region zurückzugeben.
- `describe-scaling-policies`um Informationen über Skalierungsrichtlinien in einem zurückzugeben AWS-Region.
- `delete-scaling-policy`um eine Skalierungsrichtlinie zu löschen.

Benutzerdefinierte Metriken

Benutzerdefinierte Metriken können verwendet werden, um die für eine Anwendung benötigte Kapazität vorherzusagen. Benutzerdefinierte Metriken sind nützlich, wenn vordefinierte Metriken nicht ausreichen, um die Auslastung Ihrer Anwendung zu erfassen.

Überlegungen

Die folgenden Überlegungen gelten für die Arbeit mit prädiktiver Skalierung.

- Prüfen Sie, ob Predictive Scaling für Ihre Anwendung geeignet ist. Eine Anwendung eignet sich gut für die prädiktive Skalierung, wenn sie wiederkehrende Lastmuster aufweist, die für den Wochentag

oder die Tageszeit spezifisch sind. Evaluieren Sie die Prognose, bevor Sie Predictive Scaling Ihre Anwendung aktiv skalieren lassen.

- Für die prädiktive Skalierung werden mindestens 24 Stunden an historischen Daten benötigt, damit mit der Prognose begonnen werden kann. Prognosen sind jedoch effektiver, wenn Verlaufsdaten für zwei volle Wochen zur Verfügung stehen.
- Wählen Sie eine Lastmetrik, die die volle Auslastung Ihrer Anwendung genau wiedergibt und den Aspekt Ihrer Anwendung darstellt, der für die Skalierung am wichtigsten ist.

Erstellen Sie eine Richtlinie zur vorausschauenden Skalierung für Application Auto Scaling

Die folgende Beispielrichtlinie verwendet die AWS CLI , um eine prädiktive Skalierungsrichtlinie für den Amazon ECS-Service zu konfigurieren. Ersetzen Sie jeden *user input placeholder* durch Ihre Informationen.

Weitere Informationen zu den CloudWatch Metriken, die Sie angeben können, finden Sie PredictiveScalingMetricSpecificationin der Amazon EC2 Auto Scaling API-Referenz.

Nachstehend finden Sie eine Beispielrichtlinie mit einer vordefinierten Speicherkonfiguration.

```
cat policy.json
{
    "MetricSpecifications": [
        {
            "TargetValue": 40,
            "PredefinedMetricPairSpecification": {
                "PredefinedMetricType": "ECSServiceMemoryUtilization"
            }
        }
    ],
    "SchedulingBufferTime": 3600,
    "MaxCapacityBreachBehavior": "HonorMaxCapacity",
    "Mode": "ForecastOnly"
}
```

Das folgende Beispiel veranschaulicht die Erstellung der Richtlinie, indem der put-scaling-policyBefehl mit der angegebenen Konfigurationsdatei ausgeführt wird.

```
aws aas put-scaling-policy \
```

```
--service-namespace ecs \
--region us-east-1 \
--policy-name predictive-scaling-policy-example \
--resource-id service/MyCluster/test \
--policy-type PredictiveScaling \
--scalable-dimension ecs:service:DesiredCount \
--predictive-scaling-policy-configuration file://policy.json
```

Wenn der Befehl erfolgreich ausgeführt wurde, gibt er den ARN der Richtlinie zurück.

```
{
"PolicyARN": "arn:aws:autoscaling:us-
east-1:012345678912:scalingPolicy:d1d72dfe-5fd3-464f-83cf-824f16cb88b7:resource/ecs/
service/MyCluster/test:policyName/predictive-scaling-policy-example",
"Alarms": []
}
```

Überschreiben von Prognosewerten mithilfe geplanter Aktionen

Manchmal haben Sie möglicherweise zusätzliche Informationen zu Ihren zukünftigen Anwendungsanforderungen, die bei der Prognoseberechnung nicht berücksichtigt werden können. Prognoseberechnungen können beispielsweise die Kapazität unterschätzen, die für eine bevorstehende Marketingveranstaltung benötigt wird. Sie können geplante Aktionen verwenden, um die Prognose in zukünftigen Zeiträumen vorübergehend zu überschreiben. Die geplanten Aktionen können auf einer wiederkehrenden Basis oder zu einem bestimmten Zeitpunkt ausgeführt werden, wenn einmalige Nachfrageschwankungen auftreten.

Sie können beispielsweise eine geplante Aktion mit einer höheren Mindestkapazität als die prognostizierte Aktion erstellen. Zur Laufzeit aktualisiert Application Auto Scaling die Mindestkapazität Ihres skalierbaren Ziels. Da die prädiktive Skalierung für die Kapazität optimiert wird, wird eine geplante Aktion mit einer minimalen Kapazität, die höher als die Prognosewerte ist, berücksichtigt. Dadurch wird verhindert, dass die Kapazität geringer ist als erwartet. Um das Überschreiben der Prognose zu beenden, setzen Sie über eine zweite geplante Aktion die minimale Kapazität auf ihre ursprüngliche Einstellung zurück.

Im folgenden Verfahren werden die Schritte zum Überschreiben der Prognose in zukünftigen Zeiträumen erläutert.

Themen

- Schritt 2: Erstellen von zwei geplanten Aktionen

> **Important**
>
> In diesem Thema wird davon ausgegangen, dass Sie versuchen, die Prognose zu überschreiben, um auf eine höhere Kapazität als die prognostizierte zu skalieren. Wenn Sie die Kapazität vorübergehend verringern müssen, ohne dass dies durch eine Richtlinie zur vorausschauenden Skalierung beeinträchtigt wird, verwenden Sie stattdessen den Modus „Nur Prognose“. Im Modus „Nur Prognose“ generiert die vorausschauende Skalierung zwar weiterhin Prognosen, erhöht aber nicht automatisch die Kapazität. Anschließend können Sie die Ressourcennutzung überwachen und die Größe Ihrer Gruppe nach Bedarf manuell verringern.

Schritt 1: (Optional) Analysieren von Zeitreihendaten

Beginnen Sie mit der Analyse der Prognose-Zeitreihendaten. Dies ist ein optionaler Schritt, aber es ist hilfreich, wenn Sie die Details der Prognose verstehen möchten.

1. Rufen Sie die Prognose ab

   Nachdem die Prognose erstellt wurde, können Sie einen bestimmten Zeitraum in der Prognose abfragen. Ziel der Abfrage ist es, einen vollständigen Überblick über die Zeitreihendaten für einen bestimmten Zeitraum zu erhalten.

   Ihre Abfrage kann Prognosedaten bis zwei Tage in die Zukunft enthalten. Wenn Sie die prädiktive Skalierung eine Weile verwenden, können Sie auch auf Ihre früheren Prognosedaten zugreifen. Der maximale Zeitraum zwischen der Start- und Endzeit beträgt jedoch 30 Tage.

   Verwenden Sie den get-predictive-scaling-forecastBefehl, um die Prognose abzurufen. Im folgenden Beispiel wird die Prognose für die vorausschauende Skalierung für den Amazon ECS-Service abgerufen.

```
aws application-autoscaling get-predictive-scaling-forecast --service-namespace ecs
 \
    --scalable-dimension ecs:service:DesiredCount \
    --resource-id 1234567890abcdef0
    --policy-name predictive-scaling-policy \
    --start-time "2021-05-19T17:00:00Z" \
```

```
    --end-time "2021-05-19T23:00:00Z"
```

Die Antwort umfasst zwei Prognosen: `LoadForecast` und`CapacityForecast`. `LoadForecast`zeigt die stündliche Lastprognose. `CapacityForecast`zeigt Prognosewerte für die Kapazität, die stündlich benötigt wird, um die prognostizierte Last unter Beibehaltung einer bestimmten `TargetValue` Last zu bewältigen.

2. Identifizieren des Zielzeitraums

   Ermitteln Sie die Stunde oder die Stunden, zu der/zu denen die einmalige Nachfrageschwankung stattfinden soll. Denken Sie daran, dass die in der Prognose angezeigten Datumsangaben und Uhrzeiten in UTC angegeben sind.

Schritt 2: Erstellen von zwei geplanten Aktionen

Erstellen Sie als Nächstes zwei geplante Aktionen für einen bestimmten Zeitraum, in dem Ihre Anwendung eine höhere Last aufweist als die prognostizierte Last. Wenn Sie beispielsweise während eines Marketing-Ereignisses für einen begrenzten Zeitraum ein erhöhtes Daten-Volume erwarten, können Sie eine einmalige Aktion planen, um die Mindestkapazität bei deren Beginn zu aktualisieren. Planen Sie dann eine weitere Aktion, um die Mindestkapazität auf die ursprüngliche Einstellung zurückzusetzen, wenn das Ereignis endet.

Erstellen von zwei geplanten Aktionen für einmalige Ereignisse (AWS CLI)

Verwenden Sie den put-scheduled-actionBefehl, um die geplanten Aktionen zu erstellen.

Das folgende Beispiel definiert einen Zeitplan für Amazon EC2 Auto Scaling, der acht Stunden lang eine Mindestkapazität von drei Instances am 19. Mai um 17:00 Uhr vorsieht. Die folgenden Befehle veranschaulichen die Implementierung dieses Szenarios.

Der erste Befehl put-scheduled-update-group-action weist Amazon EC2 Auto Scaling an, die Mindestkapazität der angegebenen Auto Scaling-Gruppe am 19. Mai 2021 um 17:00 Uhr UTC zu aktualisieren.

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-event-start \
  --auto-scaling-group-name my-asg --start-time "2021-05-19T17:00:00Z" --minimum-capacity 3
```

Der zweite Befehl weist Amazon EC2 Auto Scaling an, die Mindestkapazität der Gruppe am 20. Mai 2021 um 1:00 Uhr UTC auf eins zu setzen.

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-event-end
 \
  --auto-scaling-group-name my-asg --start-time "2021-05-20T01:00:00Z" --minimum-
capacity 1
```

Nachdem Sie der Auto-Scaling-Gruppe diese geplanten Aktionen hinzugefügt haben, führt Amazon EC2 Auto Scaling folgende Schritte aus:

- Um 17:00 Uhr UTC am 19. Mai 2021 wird die erste geplante Aktion ausgeführt. Wenn die Gruppe derzeit weniger als drei Instances hat, wird die Gruppe auf drei Instances skaliert. Während dieser Zeit und für die Dauer der nächsten acht Stunden kann Amazon EC2 Auto Scaling weiterhin skaliert werden, wenn die prognostizierte Kapazität höher als die tatsächliche Kapazität ist oder wenn eine Richtlinie für dynamische Skalierung in Kraft ist.
- Um 1:00 Uhr UTC am 20. Mai 2021 wird die zweite geplante Aktion ausgeführt. Dadurch wird die Mindestkapazität am Ende des Ereignisses auf die ursprüngliche Einstellung zurückgesetzt.

Skalierung basierend auf wiederkehrenden Zeitplänen

Um die Prognose jede Woche während des gleichen Zeitraums zu überschreiben, erstellen Sie zwei geplante Aktionen und stellen die Zeit- und Datumslogik mithilfe eines Cron-Ausdrucks bereit.

Der Cron-Ausdruck besteht aus fünf Feldern, getrennt durch Leerzeichen: [Minute] [Stunde] [Tag_des_Monats] [Monat_des_Jahres] [Wochentag]. Felder können alle zulässigen Werte enthalten, einschließlich Sonderzeichen.

Beispielsweise führt der folgende Cron-Ausdruck jeden Dienstag um 6:30 Uhr die Aktion aus. Das Sternchen wird als Platzhalter verwendet, um alle Werte für ein Feld abzugleichen.

```
30 6 * * 2
```

Erweiterte prädiktive Skalierungsrichtlinie unter Verwendung benutzerdefinierter Metriken

In einer prädiktiven Skalierungsrichtlinie können Sie vordefinierte oder benutzerdefinierte Metriken verwenden. Benutzerdefinierte Metriken sind nützlich, wenn die vordefinierten Metriken die Auslastung Ihrer Anwendung nicht ausreichend beschreiben.

Wenn Sie eine Richtlinie zur vorausschauenden Skalierung mit benutzerdefinierten Metriken erstellen, können Sie andere CloudWatch Metriken angeben, die von bereitgestellt werden AWS, oder Sie können Metriken angeben, die Sie selbst definieren und veröffentlichen. Sie können auch metrische Mathematik verwenden, um bestehende Metriken zu aggregieren und in eine neue Zeitreihe umzuwandeln, die AWS nicht automatisch erfasst wird. Wenn Sie Werte in Ihren Daten kombinieren, indem Sie z.B. neue Summen oder Durchschnittswerte berechnen, nennt man das Aggregieren. Die resultierenden Daten werden als Aggregat bezeichnet.

Der folgende Abschnitt enthält bewährte Verfahren und Beispiele für die Erstellung der JSON-Struktur für die Richtlinie.

Themen

Best Practices

Die folgenden bewährten Methoden können Ihnen helfen, benutzerdefinierte Metriken effektiver zu nutzen:

- Für die Lastmetrikspezifikation ist die nützlichste Metrik eine Metrik, die die Auslastung Ihrer Anwendung darstellt.
- Die Skalierungsmetrik muss umgekehrt proportional zur Kapazität sein. Das heißt, wenn das skalierbare Ziel zunimmt, sollte sich die Skalierungsmetrik ungefähr im gleichen Verhältnis verringern. Um sicherzustellen, dass sich die prädiktive Skalierung wie erwartet verhält, müssen die Lastmetrik und die Skalierungsmetrik auch stark miteinander korrelieren.

- Die Zielauslastung muss mit der Art der Skalierungsmetrik übereinstimmen. Bei einer Richtlinienkonfiguration, die die CPU-Auslastung verwendet, ist dies ein Zielprozentsatz. Bei einer Richtlinienkonfiguration, die den Durchsatz verwendet, wie z.B. die Anzahl der Anfragen oder Nachrichten, ist dies die angestrebte Anzahl von Anfragen oder Nachrichten pro Instance während eines einminütigen Intervalls.
- Wenn diese Empfehlungen nicht befolgt werden, werden die prognostizierten zukünftigen Werte der Zeitreihen wahrscheinlich falsch sein. Um zu überprüfen, ob die Daten korrekt sind, können Sie sich die prognostizierten Werte ansehen. Sie können auch nach der Erstellung Ihrer Richtlinie für vorausschauende Skalierung die `CapacityForecast` Objekte `LoadForecast` und überprüfen, die GetPredictiveScalingForecastdurch einen API-Aufruf zurückgegeben wurden.
- Wir empfehlen Ihnen dringend, die prädiktive Skalierung im Modus "Nur Prognose" zu konfigurieren, damit Sie die Prognose auswerten können, bevor die prädiktive Skalierung mit der aktiven Skalierung der Kapazität beginnt.

Voraussetzungen

Um benutzerdefinierte Metriken zu Ihrer prädiktiven Skalierungsrichtlinie hinzuzufügen, müssen Sie über entsprechende `cloudwatch:GetMetricData`-Berechtigungen verfügen.

Wenn Sie Ihre eigenen Metriken anstelle der bereitgestellten Metriken angeben möchten, müssen Sie Ihre Metriken zunächst auf CloudWatch veröffentlichen. AWS Weitere Informationen finden Sie unter Veröffentlichen benutzerdefinierter Metriken im CloudWatch Amazon-Benutzerhandbuch.

Sollten Sie Ihre eigenen Metriken veröffentlichen, achten Sie darauf, dass Sie die Datenpunkte mindestens alle fünf Minuten veröffentlichen. Application Auto Scaling ruft die Datenpunkte CloudWatch basierend auf der Länge des benötigten Zeitraums ab. Beispielsweise verwendet die Lastmetrikspezifikation stündliche Metriken, um die Auslastung Ihrer Anwendung zu messen. CloudWatch verwendet Ihre veröffentlichten Metrikdaten, um einen einzelnen Datenwert für einen beliebigen Zeitraum von einer Stunde bereitzustellen, indem alle Datenpunkte mit Zeitstempeln aggregiert werden, die in jeden Zeitraum von einer Stunde fallen.

Konstruieren von JSON für benutzerdefinierte Metriken

Der folgende Abschnitt enthält Beispiele für die Konfiguration von Predictive Scaling zur Abfrage von Daten CloudWatch für Amazon EC2 Auto Scaling. Es gibt zwei verschiedene Methoden, um diese Option zu konfigurieren, und die von Ihnen gewählte Methode wirkt sich darauf aus, welches Format Sie verwenden, um den JSON für Ihre prädiktive Skalierungsrichtlinie zu erstellen. Wenn

Sie metrische Berechnungen verwenden, variiert das Format von JSON je nach der durchgeführten metrischen Berechnung weiter.

1. Informationen zum Erstellen einer Richtlinie, mit der Daten direkt aus anderen CloudWatch Metriken abgerufen werden, die von bereitgestellt werden AWS oder für die Sie Daten veröffentlichen CloudWatch, finden Sie unter. Beispiel einer prädiktiven Skalierungsrichtlinie mit benutzerdefinierten Last- und Skalierungsmetriken (AWS CLI)
2. Informationen zum Erstellen einer Richtlinie, mit der mehrere CloudWatch Messwerte abgefragt und mithilfe mathematischer Ausdrücke neue Zeitreihen auf der Grundlage dieser Messwerte erstellt werden können, finden Sie unterMetrikberechnungs-Ausdrücke verwenden.

Beispiel einer prädiktiven Skalierungsrichtlinie mit benutzerdefinierten Last- und Skalierungsmetriken (AWS CLI)

Um eine prädiktive Skalierungsrichtlinie mit benutzerdefinierten Last- und Skalierungsmetriken mit dem zu erstellen AWS CLI, speichern Sie die Argumente für `--predictive-scaling-configuration` in einer JSON-Datei mit dem Namen`config.json`.

Sie beginnen mit dem Hinzufügen benutzerdefinierter Metriken, indem Sie die ersetzbaren Werte im folgenden Beispiel durch die Werte Ihrer Metriken und Ihrer Zielauslastung ersetzen.

```
{
  "MetricSpecifications": [
    {
      "TargetValue": 50,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "scaling_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "MyUtilizationMetric",
                "Namespace": "MyNameSpace",
                "Dimensions": [
                  {
                    "Name": "MyOptionalMetricDimensionName",
                    "Value": "MyOptionalMetricDimensionValue"
                  }
                ]
              },
```

```
            "Stat": "Average"
          }
        }
      ]
    },
    "CustomizedLoadMetricSpecification": {
      "MetricDataQueries": [
        {
          "Id": "load_metric",
          "MetricStat": {
            "Metric": {
              "MetricName": "MyLoadMetric",
              "Namespace": "MyNameSpace",
              "Dimensions": [
                {
                  "Name": "MyOptionalMetricDimensionName",
                  "Value": "MyOptionalMetricDimensionValue"
                }
              ]
            },
            "Stat": "Sum"
          }
        }
      ]
    }
  }
]
}
```

Weitere Informationen finden Sie MetricDataQueryin der Amazon EC2 Auto Scaling API-Referenz.

> **Note**
>
> Im Folgenden finden Sie einige zusätzliche Ressourcen, die Ihnen bei der Suche nach Metriknamen, Namespaces, Dimensionen und Statistiken für Metriken helfen können: CloudWatch
>
> - Informationen zu den verfügbaren Metriken für AWS Services finden Sie im CloudWatch Amazon-Benutzerhandbuch unter AWS Services, die CloudWatch Metriken veröffentlichen.
> - Den genauen Metriknamen, den Namespace und die Dimensionen (falls zutreffend) für eine CloudWatch Metrik mit dem finden Sie unter AWS CLIlist-metrics.

Um diese Richtlinie zu erstellen, führen Sie den put-scaling-policyBefehl mit der JSON-Datei als Eingabe aus, wie im folgenden Beispiel gezeigt.

```
aws autoscaling put-scaling-policy --policy-name my-predictive-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
```

Wenn der Befehl erfolgreich ausgeführt wurde, gibt er den Amazon-Ressourcennamen (ARN) der Richtlinie zurück.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-predictive-scaling-policy",
  "Alarms": []
}
```

Metrikberechnungs-Ausdrücke verwenden

Der folgende Abschnitt enthält Informationen und Beispiele für Richtlinien zur vorausschauenden Skalierung, die zeigen, wie Sie metrische Berechnungen in Ihrer Richtlinie verwenden können.

Themen

- Metrikberechnung verstehen
- Beispiel für eine prädiktive Skalierungsrichtlinie für Amazon EC2 Auto Scaling, die Metriken mithilfe von Metric Math () kombiniert AWS CLI
- Beispiel für eine Richtlinie zur vorausschauenden Skalierung zur Verwendung in einem blue/green Bereitstellungsszenario ()AWS CLI

Metrikberechnung verstehen

Wenn Sie lediglich vorhandene Metrikdaten aggregieren möchten, erspart Ihnen CloudWatch Metric Math den Aufwand und die Kosten für die Veröffentlichung einer weiteren Metrik in CloudWatch. Sie können jede verfügbare Metrik verwenden AWS , und Sie können auch Metriken verwenden, die Sie als Teil Ihrer Anwendungen definieren.

Weitere Informationen finden Sie unter Verwenden von metrischer Mathematik im CloudWatch Amazon-Benutzerhandbuch.

Wenn Sie sich für die Verwendung eines metrischen mathematischen Ausdrucks in Ihrer prädiktiven Skalierungsrichtlinie entscheiden, sollten Sie die folgenden Punkte beachten:

- Metrische Rechenoperationen verwenden die Datenpunkte der eindeutigen Kombination aus Metrikname, Namespace und Dimensionsschlüssel/Wertpaaren von Metriken.
- Sie können einen beliebigen arithmetischen Operator (+ - */^), jede statistische Funktion (wie AVG oder SUM) oder eine andere Funktion verwenden, die diese CloudWatch Funktion unterstützt.
- Sie können sowohl Metriken als auch die Ergebnisse anderer mathematischer Ausdrücke in den Formeln des mathematischen Ausdrucks verwenden.
- Ihre metrischen mathematischen Ausdrücke können aus verschiedenen Aggregationen zusammengesetzt sein. Für das endgültige Aggregationsergebnis ist es jedoch eine bewährte Methode, `Average` für die Skalierungsmetrik und `Sum` für die Lastmetrik zu verwenden.
- Alle Ausdrücke, die in einer metrischen Spezifikation verwendet werden, müssen letztendlich eine einzige Zeitreihe ergeben.

Um metrische Mathematik zu verwenden, gehen Sie wie folgt vor:

- Wählen Sie eine oder mehrere CloudWatch Metriken aus. Erstellen Sie dann den Ausdruck. Weitere Informationen finden Sie unter Verwenden von metrischer Mathematik im CloudWatch Amazon-Benutzerhandbuch.
- Stellen Sie mithilfe der CloudWatch Konsole oder der CloudWatch GetMetricDataAPI sicher, dass der metrische mathematische Ausdruck gültig ist.

Beispiel für eine prädiktive Skalierungsrichtlinie für Amazon EC2 Auto Scaling, die Metriken mithilfe von Metric Math () kombiniert AWS CLI

Manchmal müssen Sie die Metrik nicht direkt angeben, sondern die Daten erst auf irgendeine Weise verarbeiten. Sie könnten zum Beispiel eine Anwendung haben, die Arbeit aus einer Amazon SQS-Warteschlange abruft, und Sie könnten die Anzahl der Objekte in der Warteschlange als Kriterium für die prädiktive Skalierung verwenden wollen. Die Anzahl der Nachrichten in der Warteschlange bestimmt nicht allein die Anzahl der Instances, die Sie benötigen. Daher ist weitere Arbeit erforderlich, um eine Metrik zu erstellen, die zur Berechnung des Rückstands pro Instance verwendet werden kann.

Im Folgenden finden Sie ein Beispiel für eine prädiktive Skalierungsrichtlinie für dieses Szenario. Sie legt Skalierungs- und Auslastungsmetriken fest, die auf der Amazon SQS

`ApproximateNumberOfMessagesVisible`-Metrik basieren, d.h. der Anzahl der Nachrichten, die für den Abruf aus der Warteschlange verfügbar sind. Es verwendet auch die Amazon EC2 Auto Scaling `GroupInServiceInstances`-Metrik und einen mathematischen Ausdruck, um den Rückstand pro Instance für die Skalierungsmetrik zu berechnen.

```
aws autoscaling put-scaling-policy --policy-name my-sqs-custom-metrics-policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
{
  "MetricSpecifications": [
    {
      "TargetValue": 100,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Label": "Get the queue size (the number of messages waiting to be
 processed)",
            "Id": "queue_size",
            "MetricStat": {
              "Metric": {
                "MetricName": "ApproximateNumberOfMessagesVisible",
                "Namespace": "AWS/SQS",
                "Dimensions": [
                  {
                    "Name": "QueueName",
                    "Value": "my-queue"
                  }
                ]
              },
              "Stat": "Sum"
            },
            "ReturnData": false
          },
          {
            "Label": "Get the group size (the number of running instances)",
            "Id": "running_capacity",
            "MetricStat": {
              "Metric": {
                "MetricName": "GroupInServiceInstances",
                "Namespace": "AWS/AutoScaling",
                "Dimensions": [
                  {
                    "Name": "AutoScalingGroupName",
```

```
                    "Value": "my-asg"
                  }
                ]
              },
              "Stat": "Sum"
            },
            "ReturnData": false
          },
          {
            "Label": "Calculate the backlog per instance",
            "Id": "scaling_metric",
            "Expression": "queue_size / running_capacity",
            "ReturnData": true
          }
        ]
      },
      "CustomizedLoadMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "ApproximateNumberOfMessagesVisible",
                "Namespace": "AWS/SQS",
                "Dimensions": [
                  {
                    "Name": "QueueName",
                    "Value": "my-queue"
                  }
                ],
              },
              "Stat": "Sum"
            },
            "ReturnData": true
          }
        ]
      }
    }
  ]
}
```

Das Beispiel gibt den ARN der Richtlinie zurück.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-sqs-custom-metrics-policy",
  "Alarms": []
}
```

Beispiel für eine Richtlinie zur vorausschauenden Skalierung zur Verwendung in einem blue/green Bereitstellungsszenario ()AWS CLI

Ein Suchausdruck bietet eine erweiterte Option, mit der Sie eine Metrik aus mehreren Auto-Scaling-Gruppen abfragen und mathematische Ausdrücke auf sie anwenden können. Dies ist besonders nützlich für blue/green Bereitstellungen.

> **Note**
>
> Eine blau/grüne Bereitstellung ist eine Bereitstellungsmethode, bei der Sie zwei separate, aber identische Auto-Scaling-Gruppen erstellen. Nur eine der Gruppen empfängt den Produktionsverkehr. Der Benutzerverkehr wird zunächst auf die frühere ("blaue") Auto-Scaling-Gruppe geleitet, während eine neue Gruppe („grün“) zum Testen und Evaluieren einer neuen Version einer Anwendung oder eines Dienstes verwendet wird. Der Benutzerverkehr wird auf die grüne Auto-Scaling-Gruppe verlagert, nachdem eine neue Bereitstellung getestet und akzeptiert wurde. Sie können die blaue Gruppe dann löschen, nachdem die Bereitstellung erfolgreich war.

Wenn im Rahmen einer blue/green Bereitstellung neue Auto Scaling Scaling-Gruppen erstellt werden, kann der Metrikverlauf jeder Gruppe automatisch in die Predictive Scaling-Richtlinie aufgenommen werden, ohne dass Sie die Metrikspezifikationen ändern müssen. Weitere Informationen finden Sie im Compute-Blog unter Verwenden von Richtlinien zur vorausschauenden Skalierung von EC2 Auto Scaling mit Blue/Green-Bereitstellungen. AWS

Die folgende Beispielrichtlinie zeigt, wie dies geschehen kann. In diesem Beispiel verwendet die Richtlinie die von Amazon EC2 ausgegebene `CPUUtilization`-Metrik. Es verwendet die Amazon EC2 Auto Scaling `GroupInServiceInstances`-Metrik und einen mathematischen Ausdruck, um den Wert der Skalierungsmetrik pro Instance zu berechnen. Sie gibt auch eine Kapazitätsmetrik an, um die `GroupInServiceInstances`-Metrik zu erhalten.

Der Suchausdruck findet das `CPUUtilization` von Instances in mehreren Auto-Scaling-Gruppen anhand der angegebenen Suchkriterien. Wenn Sie zu einem späteren Zeitpunkt eine neue Auto-

Scaling-Gruppe erstellen, die denselben Suchkriterien entspricht, werden die `CPUUtilization` der Instances in der neuen Auto-Scaling-Gruppe automatisch einbezogen.

```
aws autoscaling put-scaling-policy --policy-name my-blue-green-predictive-scaling-
policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
{
  "MetricSpecifications": [
    {
      "TargetValue": 25,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_sum",
            "Expression": "SUM(SEARCH('{AWS/EC2,AutoScalingGroupName} MetricName=
\"CPUUtilization\" ASG-myapp', 'Sum', 300))",
            "ReturnData": false
          },
          {
            "Id": "capacity_sum",
            "Expression": "SUM(SEARCH('{AWS/AutoScaling,AutoScalingGroupName}
 MetricName=\"GroupInServiceInstances\" ASG-myapp', 'Average', 300))",
            "ReturnData": false
          },
          {
            "Id": "weighted_average",
            "Expression": "load_sum / capacity_sum",
            "ReturnData": true
          }
        ]
      },
      "CustomizedLoadMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_sum",
            "Expression": "SUM(SEARCH('{AWS/EC2,AutoScalingGroupName} MetricName=
\"CPUUtilization\" ASG-myapp', 'Sum', 3600))"
          }
        ]
      },
      "CustomizedCapacityMetricSpecification": {
        "MetricDataQueries": [
```

```
          {
            "Id": "capacity_sum",
            "Expression": "SUM(SEARCH('{AWS/AutoScaling,AutoScalingGroupName}
 MetricName=\"GroupInServiceInstances\" ASG-myapp', 'Average', 300))"
          }
        ]
      }
    }
  ]
}
```

Das Beispiel gibt den ARN der Richtlinie zurück.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-blue-green-predictive-
scaling-policy",
  "Alarms": []
}
```

Überlegungen zu benutzerdefinierten Metriken in einer Richtlinie zur prädiktiven Skalierung

Wenn bei der Verwendung von benutzerdefinierten Metriken ein Problem auftritt, empfehlen wir Ihnen, wie folgt vorzugehen:

- Wenn eine Fehlermeldung angezeigt wird, lesen Sie die Nachricht und beheben Sie das gemeldete Problem, falls möglich.
- Wenn Sie einen Ausdruck nicht im Voraus validiert haben, validiert der put-scaling-policyBefehl ihn, wenn Sie Ihre Skalierungsrichtlinie erstellen. Es besteht jedoch die Möglichkeit, dass dieser Befehl die genaue Ursache der erkannten Fehler nicht identifizieren kann. Um die Probleme zu beheben, beheben Sie die Fehler, die Sie als Antwort auf eine Anfrage an den get-metric-dataBefehl erhalten. Sie können den Ausdruck auch von der CloudWatch Konsole aus beheben.
- Sie müssen `false` für `ReturnData` angeben, wenn `MetricDataQueries` die Funktion SEARCH() allein ohne eine mathematische Funktion wie SUM() angibt. Das liegt daran, dass Suchausdrücke mehrere Zeitreihen zurückgeben können, während eine auf einem Ausdruck basierende Metrikspezifikation nur eine Zeitreihe zurückgeben kann.
- Alle an einem Suchausdruck beteiligten Metriken sollten die gleiche Auflösung haben.

Einschränkungen

Die folgenden Einschränkungen gelten:

- Sie können Datenpunkte von bis zu 10 Metriken in einer Metrikspezifikation abfragen.
- Für die Zwecke dieses Limits zählt ein Ausdruck als eine Metrik.

Tutorial: Auto Scaling zur Bewältigung eines hohen Workloads konfigurieren

In diesem Tutorial lernen Sie, wie Sie in Zeitfenstern, in denen Ihre Anwendung ein höheres als das normale Workload aufweist, horizontal skalieren. Dies ist hilfreich, wenn Sie eine Anwendung haben, die in regelmäßigen Abständen oder saisonal plötzlich eine große Anzahl von Besuchern haben kann.

Sie können eine Zielverfolgungs-Skalierungsrichtlinie zusammen mit einer geplanten Skalierung verwenden, um die zusätzliche Last zu bewältigen. Bei der geplanten Skalierung werden automatisch Änderungen an Ihren `MinCapacity` und `MaxCapacity` in Ihrem Namen nach einem von Ihnen festgelegten Zeitplan initiiert. Wenn eine Zielverfolgungs-Skalierungsrichtlinie für die Ressource aktiv ist, kann sie dynamisch auf der Grundlage der aktuellen Ressourcenauslastung innerhalb des neuen minimalen und maximalen Kapazitätsbereichs skaliert werden.

Nach Abschluss dieses Tutorials wissen Sie, wie Sie:

- Mit der geplanten Skalierung können Sie zusätzliche Kapazität hinzufügen, um eine hohe Last zu bewältigen, bevor sie eintrifft, und die zusätzliche Kapazität entfernen, wenn sie nicht mehr benötigt wird.
- Verwenden Sie eine Zielverfolgungs-Skalierungsrichtlinie, um Ihre Anwendung auf der Grundlage der aktuellen Ressourcenauslastung zu skalieren.

Inhalt

Voraussetzungen

In diesem Tutorial wird davon ausgegangen, dass Sie Folgendes bereits gemacht haben:

- Erstellt ein AWS-Konto.
- Installierte und konfigurierte die AWS CLI.
- Mit Application Auto Scaling wurden die erforderlichen Berechtigungen für die Registrierung und Deregistrierung von Ressourcen als skalierbare Ziele erteilt. Darüber hinaus wurden die erforderlichen Berechtigungen für die Erstellung von Skalierungsrichtlinien und geplanten Aktionen erteilt. Weitere Informationen finden Sie unter Identity and Access Management für Application Auto Scaling.
- Es wurde eine unterstützte Ressource in einer Umgebung außerhalb der Produktionsumgebung erstellt, die für dieses Tutorial zur Verfügung steht. Wenn Sie noch über keines verfügen, erstellen Sie jetzt eines. Weitere Informationen zu den AWS -Services und Ressourcen, die mit Application Auto Scaling zusammenarbeiten, finden Sie im AWS-Services die Sie mit Application Auto Scaling verwenden können-Abschnitt.

> **Note**
>
> Bei der Durchführung dieses Tutorials gibt es zwei Schritte, in denen Sie die minimalen und maximalen Kapazitätswerte Ihrer Ressource auf 0 festlegen, um die aktuelle Kapazität auf 0 zurückzusetzen. Je nachdem, welche Ressource Sie mit Application Auto Scaling verwenden, können Sie die aktuelle Kapazität während dieser Schritte möglicherweise nicht auf 0 zurücksetzen. Um Ihnen bei der Lösung des Problems zu helfen, wird in der Ausgabe eine Meldung angezeigt, dass die Mindestkapazität nicht unter dem angegebenen Wert liegen darf. Außerdem wird der Mindestkapazitätswert angegeben, den die AWS Ressource akzeptieren kann.

Schritt 1: Registrieren Sie Ihr skalierbares Ziel

Beginnen Sie damit, Ihre Ressource als skalierbares Ziel bei Application Auto Scaling zu registrieren. Ein skalierbares Ziel ist eine Ressource, die dank Application Auto Scaling auf- bzw. abskaliert werden kann.

So registrieren Sie Ihr skalierbares Ziel mit Application Auto Scaling

- Verwenden Sie den folgenden register-scalable-targetBefehl, um ein neues skalierbares Ziel zu registrieren. Setzen Sie die Werte `--min-capacity` und `--max-capacity` auf 0, um die aktuelle Kapazität auf 0 zurückzusetzen.

Ersetzen Sie den Beispieltext für `--service-namespace` mit dem Namespace des AWS -Services, den Sie mit Application Auto Scaling verwenden, `--scalable-dimension` mit der skalierbaren Dimension im Zusammenhang mit der Ressource, die Sie registrieren, `--resource-id` mit einem Bezeichner für die Ressource. Diese Werte variieren je nachdem, welche Ressource verwendet wird und wie die Ressourcen-ID erstellt wird. Sehen Sie sich die Themen im AWS-Services die Sie mit Application Auto Scaling verwenden können-Abschnitt für weitere Informationen an. Zu diesen Themen gehören Beispielbefehle, die Ihnen zeigen, wie Sie skalierbare Ziele bei Application Auto Scaling registrieren.

Linux, macOS oder Unix

```
aws application-autoscaling register-scalable-target \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --min-capacity 0 --max-capacity 0
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace namespace
 --scalable-dimension dimension --resource-id identifier --min-capacity 0 --max-
capacity 0
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-
id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Schritt 2: Richten Sie geplante Aktionen entsprechend Ihren Anforderungen ein

Sie können den put-scheduled-actionBefehl verwenden, um geplante Aktionen zu erstellen, die so konfiguriert sind, dass sie Ihren Geschäftsanforderungen entsprechen. In diesem Tutorial

konzentrieren wir uns auf eine Konfiguration, die den Verbrauch von Ressourcen außerhalb der Arbeitszeiten stoppt, indem die Kapazität auf 0 reduziert wird.

So erstellen Sie eine geplante Aktion, die am Morgen ausläuft

1. Verwenden Sie den folgenden put-scheduled-actionBefehl, um das skalierbare Ziel zu skalieren. Fügen Sie den Parameter `--schedule` mit einem wiederkehrenden Zeitplan in UTC ein, indem Sie einen cron-Ausdruck verwenden.

   Nach dem festgelegten Zeitplan (jeden Tag um 9:00 Uhr UTC) aktualisiert Application Auto Scaling die Werte `MinCapacity` und `MaxCapacity` auf den gewünschten Bereich von 1-5 Kapazitätseinheiten.

   Linux, macOS oder Unix

   ```
   aws application-autoscaling put-scheduled-action \
     --service-namespace namespace \
     --scalable-dimension dimension \
     --resource-id identifier \
     --scheduled-action-name my-first-scheduled-action \
     --schedule "cron(0 9 * * ? *)" \
     --scalable-target-action MinCapacity=1,MaxCapacity=5
   ```

   Windows

   ```
   aws application-autoscaling put-scheduled-action --service-namespace namespace --
   scalable-dimension dimension --resource-id identifier --scheduled-action-name my-
   first-scheduled-action --schedule "cron(0 9 * * ? *)" --scalable-target-action
    MinCapacity=1,MaxCapacity=5
   ```

   Dieser Befehl gibt keine Ausgabe zurück, wenn er nicht erfolgreich ist.

2. Verwenden Sie den folgenden describe-scheduled-actionsBefehl, um zu überprüfen, ob Ihre geplante Aktion vorhanden ist.

   Linux, macOS oder Unix

   ```
   aws application-autoscaling describe-scheduled-actions \
     --service-namespace namespace \
     --query 'ScheduledActions[?ResourceId==`identifier`]'
   ```

Windows

```
aws application-autoscaling describe-scheduled-actions --service-
namespace namespace --query "ScheduledActions[?ResourceId==`identifier`]"
```

Es folgt eine Beispielausgabe.

```
[
    {
        "ScheduledActionName": "my-first-scheduled-action",
        "ScheduledActionARN": "arn",
        "Schedule": "cron(0 9 * * ? *)",
        "ScalableTargetAction": {
            "MinCapacity": 1,
            "MaxCapacity": 5
        },
        ...
    }
]
```

So erstellen Sie eine geplante Aktion, die nachts aktiviert wird

1. Wiederholen Sie das vorangegangene Verfahren, um eine weitere geplante Aktion zu erstellen, mit der Application Auto Scaling am Ende des Tages aktiviert wird.

   Gemäß dem angegebenen Zeitplan (täglich um 20:00 Uhr UTC) aktualisiert Application Auto Scaling den Wert `MinCapacity` und des Ziels `MaxCapacity` auf 0, wie im folgenden put-scheduled-actionBefehl angegeben.

   Linux, macOS oder Unix

```
aws application-autoscaling put-scheduled-action \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --scheduled-action-name my-second-scheduled-action \
  --schedule "cron(0 20 * * ? *)" \
  --scalable-target-action MinCapacity=0,MaxCapacity=0
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace namespace --
scalable-dimension dimension --resource-id identifier --scheduled-action-name my-
second-scheduled-action --schedule "cron(0 20 * * ? *)" --scalable-target-action
 MinCapacity=0,MaxCapacity=0
```

2. Verwenden Sie den folgenden describe-scheduled-actionsBefehl, um zu überprüfen, ob Ihre geplante Aktion vorhanden ist.

   Linux, macOS oder Unix

   ```
   aws application-autoscaling describe-scheduled-actions \
     --service-namespace namespace \
     --query 'ScheduledActions[?ResourceId==`identifier`]'
   ```

   Windows

   ```
   aws application-autoscaling describe-scheduled-actions --service-
   namespace namespace --query "ScheduledActions[?ResourceId==`identifier`]"
   ```

   Es folgt eine Beispielausgabe.

   ```
   [
       {
           "ScheduledActionName": "my-first-scheduled-action",
           "ScheduledActionARN": "arn",
           "Schedule": "cron(0 9 * * ? *)",
           "ScalableTargetAction": {
               "MinCapacity": 1,
               "MaxCapacity": 5
           },
           ...
       },
       {
           "ScheduledActionName": "my-second-scheduled-action",
           "ScheduledActionARN": "arn",
           "Schedule": "cron(0 20 * * ? *)",
           "ScalableTargetAction": {
               "MinCapacity": 0,
               "MaxCapacity": 0
   ```

```
        },
        ...
    }
]
```

Schritt 3: Hinzufügen einer Skalierungsrichtlinie für die Zielverfolgung

Bei der Zielverfolgung vergleicht Application Auto Scaling den Zielwert in der Richtlinie mit dem aktuellen Wert der angegebenen Metrik.

Bei der Zielverfolgung vergleicht Application Auto Scaling den Zielwert in der Richtlinie mit dem aktuellen Wert der angegebenen Metrik. Wenn diese Werte über einen bestimmten Zeitraum hinweg nicht übereinstimmen, fügt Application Auto Scaling Kapazitäten hinzu oder entfernt sie, um eine gleichmäßige Leistung zu gewährleisten. Wenn die Belastung Ihrer Anwendung und der Metrikwert steigen, fügt Application Auto Scaling so schnell wie möglich Kapazität hinzu, ohne den Wert `MaxCapacity` zu überschreiten. Wenn Application Auto Scaling Kapazität entfernt, weil die Last minimal ist, geschieht dies, ohne dass der Wert unter `MinCapacity` fällt. Durch die Anpassung der Kapazität an die Nutzung zahlen Sie nur für das, was Ihre Anwendung benötigt.

Wenn die Metrik unzureichende Daten aufweist, weil Ihre Anwendung nicht ausgelastet ist, wird durch Application Auto Scaling keine Kapazität hinzugefügt oder entfernt. Mit anderen Worten: Application Auto Scaling priorisiert die Verfügbarkeit in Situationen, in denen nicht genügend Informationen verfügbar sind.

Sie können mehrere Skalierungsrichtlinien hinzufügen, aber stellen Sie sicher, dass Sie keine widersprüchlichen Stufenskalierungsrichtlinien hinzufügen, da dies zu unerwünschtem Verhalten führen könnte. Wenn beispielsweise die Schrittskalierungsrichtlinie eine Abwärtsskalierungsaktivität initiiert, bevor die Zielverfolgungsrichtlinie abwärts skaliert werden kann, wird die Abwärtsskalierungsaktivität nicht blockiert. Nach Abschluss der Skalierungsaktivität kann die Zielverfolgungsrichtlinie Application Auto Scaling anweisen, die Skalierung wieder zu beenden.

So erstellen Sie eine Skalierungsrichtlinie für die Ziel-Nachverfolgung

1. Verwenden Sie den folgenden put-scaling-policy-Befehl, um die Richtlinie zu erstellen.

   Die Metriken, die am häufigsten für die Zielverfolgung verwendet werden, sind vordefiniert, und Sie können sie verwenden, ohne die vollständige Metrikspezifikation von CloudWatch

bereitzustellen. Weitere Informationen zu den verfügbaren vordefinierten Metriken finden Sie unter Zielverfolgungs-Skalierungsrichtlinien für Application Auto Scaling.

Bevor Sie diesen Befehl ausführen, stellen Sie sicher, dass Ihre vordefinierte Metrik den Zielwert erwartet. Wenn Sie beispielsweise eine Skalierung vornehmen möchten, wenn die CPU-Auslastung 50 % erreicht, geben Sie einen Zielwert von 50,0 an. Oder geben Sie einen Zielwert von 0,7 an, um die von Lambda bereitgestellte Gleichzeitigkeit zu reduzieren, wenn die Auslastung 70 % erreicht. Informationen zu den Zielwerten für eine bestimmte Ressource finden Sie in der vom Dienst bereitgestellten Dokumentation zur Konfiguration der Zielverfolgung. Weitere Informationen finden Sie unter AWS-Services die Sie mit Application Auto Scaling verwenden können.

Linux, macOS oder Unix

```
aws application-autoscaling put-scaling-policy \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --policy-name my-scaling-policy --policy-type TargetTrackingScaling \
  --target-tracking-scaling-policy-configuration '{ "TargetValue": 50.0,
 "PredefinedMetricSpecification": { "PredefinedMetricType": "predefinedmetric" }}'
```

Windows

```
aws application-autoscaling put-scaling-policy --service-namespace namespace --
scalable-dimension dimension --resource-id identifier --policy-name my-scaling-
policy --policy-type TargetTrackingScaling --target-tracking-scaling-policy-
configuration "{ \"TargetValue\": 50.0, \"PredefinedMetricSpecification\":
 { \"PredefinedMetricType\": \"predefinedmetric\" }}"
```

Bei Erfolg gibt dieser Befehl die Namen ARNs und der beiden CloudWatch Alarme zurück, die in Ihrem Namen erstellt wurden.

2. Verwenden Sie den folgenden describe-scaling-policiesBefehl, um zu überprüfen, ob Ihre geplante Aktion vorhanden ist.

Linux, macOS oder Unix

```
aws application-autoscaling describe-scaling-policies --service-namespace namespace
 \
```

```
  --query 'ScalingPolicies[?ResourceId==`identifier`]'
```

Windows

```
aws application-autoscaling describe-scaling-policies --service-namespace namespace
 --query "ScalingPolicies[?ResourceId==`identifier`]"
```

Es folgt eine Beispielausgabe.

```
[
    {
        "PolicyARN": "arn",
        "TargetTrackingScalingPolicyConfiguration": {
            "PredefinedMetricSpecification": {
                "PredefinedMetricType": "predefinedmetric"
            },
            "TargetValue": 50.0
        },
        "PolicyName": "my-scaling-policy",
        "PolicyType": "TargetTrackingScaling",
        "Alarms": [],
        ...
    }
]
```

Schritt 4: Nächste Schritte

Wenn eine Skalierung auftritt, wird in der Ausgabe der Skalierung für das skalierbare Ziel eine Aufzeichnung angezeigt, zum Beispiel:

```
Successfully set desired count to 1. Change successfully fulfilled by ecs.
```

Um Ihre Skalierungsaktivitäten mit Application Auto Scaling zu überwachen, können Sie den folgenden describe-scaling-activitiesBefehl verwenden.

Linux, macOS oder Unix

```
aws application-autoscaling describe-scaling-activities
  --service-namespace namespace \
```

```
  --scalable-dimension dimension \
  --resource-id identifier
```

Windows

```
aws application-autoscaling describe-scaling-activities --service-namespace namespace
 --scalable-dimension dimension --resource-id identifier
```

Schritt 5: Bereinigen

Um zu verhindern, dass für Ihr Konto Gebühren für Ressourcen anfallen, die während der aktiven Skalierung erstellt wurden, können Sie die zugehörige Skalierungskonfiguration wie folgt bereinigen.

Durch das Löschen der Skalierungskonfiguration wird die zugrunde liegende AWS Ressource nicht gelöscht. Sie wird auch nicht auf ihre ursprüngliche Kapazität zurückgesetzt. Sie können die Konsole des Dienstes, in dem Sie die Ressource erstellt haben, verwenden, um sie zu löschen oder ihre Kapazität anzupassen.

So löschen Sie die geplanten Aktionen

Der folgende Befehl delete-scheduled-action löscht eine angegebene geplante Aktion. Sie können diesen Schritt überspringen, wenn Sie die von Ihnen erstellten geplanten Aktionen beibehalten möchten.

Linux, macOS oder Unix

```
aws application-autoscaling delete-scheduled-action \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --scheduled-action-name my-second-scheduled-action
```

Windows

```
aws application-autoscaling delete-scheduled-action --service-namespace namespace
 --scalable-dimension dimension --resource-id identifier --scheduled-action-name my-
second-scheduled-action
```

So löschen Sie die Skalierungsrichtlinie

Mit dem folgenden delete-scaling-policyBefehl wird eine angegebene Skalierungsrichtlinie für die Zielverfolgung gelöscht. Sie können diesen Schritt überspringen, wenn Sie die von Ihnen erstellte Skalierungsrichtlinie beibehalten möchten.

Linux, macOS oder Unix

```
aws application-autoscaling delete-scaling-policy \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --policy-name my-scaling-policy
```

Windows

```
aws application-autoscaling delete-scaling-policy --service-namespace namespace --
scalable-dimension dimension --resource-id identifier --policy-name my-scaling-policy
```

So melden Sie das skalierbare Ziel ab

Mit dem folgenden Befehl deregister-scalable-target können Sie die Registrierung des skalierbaren Ziels aufheben. Mit diesem Befehl werden alle Skalierungsrichtlinien, die Sie erstellt haben, und alle geplanten Aktionen, die Sie noch nicht gelöscht haben, gelöscht. Sie können diesen Schritt überspringen, wenn das skalierbare Ziel für eine zukünftige Verwendung registriert bleiben soll.

Linux, macOS oder Unix

```
aws application-autoscaling deregister-scalable-target \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier
```

Windows

```
aws application-autoscaling deregister-scalable-target --service-namespace namespace --
scalable-dimension dimension --resource-id identifier
```

Die Skalierung von Application Auto Scaling unterbrechen und wiederaufnehmen

In diesem Thema wird erläutert, wie Sie mindestens eine der Skalierungsaktivitäten für die skalierbare Ziele in Ihrer Anwendung anhalten und anschließend wieder fortsetzen. Die Funktion zum Aus- und Fortsetzen wird zum vorübergehenden Anhalten von Skalierungsaktivitäten verwendet, die von Ihren Skalierungsrichtlinien und geplanten Aktionen ausgelöst wurden. Dies kann beispielsweise nützlich sein, wenn die automatische Skalierung Sie nicht beim Ausführen von Änderungen oder Untersuchen von Konfigurationsproblemen unterbrechen soll. Ihre Skalierungsrichtlinien und geplanten Aktionen werden beibehalten und die Skalierungsaktivitäten werden fortgesetzt werden, wenn Sie bereit sind.

In den folgenden CLI-Beispielbefehlen übergeben Sie die JSON-formatierten Parameter in einer config.json-Datei. Sie können diese Parameter auch in der Befehlszeile übergeben, indem Sie die JSON-Datenstruktur in Anführungszeichen einschließen. Weitere Informationen finden Sie unter Verwendung von Anführungszeichen bei Zeichenketten im AWS CLI in the AWS Command Line Interface User Guide.

Inhalt

> **Note**
>
> Anweisungen zum Aussetzen von Scale-Out-Prozessen während laufender Amazon ECS-Bereitstellungen finden Sie in der folgenden Dokumentation:
> auto Skalierung und Bereitstellungen von Services im Amazon Elastic Container Service Developer Guide

Skalierung von Aktivitäten

Application Auto Scaling unterstützt die Aussetzung der folgenden Skalierungsaktivitäten:

- Alle Skalierungsaktivitäten nach unten, die von einer Skalierungsrichtlinie ausgelöst werden.

- Alle Skalierungsaktivitäten nach oben, die von einer Skalierungsrichtlinie ausgelöst werden.
- Alle Skalierungen, die geplante Aktionen umfassen.

In den folgenden Beschreibungen wird erläutert, was passiert, wenn einzelne Skalierungen ausgesetzt werden. Jede davon kann unabhängig voneinander ausgesetzt und fortgesetzt werden. Je nach Grund für das Aussetzen einer Skalierungsaktivität müssen Sie möglicherweise mehrere Skalierungsaktivitäten zusammen aussetzen.

`DynamicScalingInSuspended`

- Application Auto Scaling entfernt keine Kapazität, wenn eine Zielverfolgungs-Skalierungsrichtlinie oder eine Stufenskalierungsrichtlinie ausgelöst wird. Auf diese Weise können Sie Skalierungsaktivitäten nach unten im Zusammenhang mit Skalierungsrichtlinien vorübergehend deaktivieren, ohne die Skalierungsrichtlinien oder die zugehörigen CloudWatch -Alarme löschen zu müssen. Wenn Sie die Skalierung wieder aufnehmen, bewertet Application Auto Scaling Richtlinien mit Alarmschwellenwerten, die derzeit verletzt werden.

`DynamicScalingOutSuspended`

- Application Auto Scaling fügt keine Kapazität hinzu, wenn eine Zielverfolgungs-Skalierungsrichtlinie oder eine Stufenskalierungsrichtlinie ausgelöst wird. Auf diese Weise können Sie horizontale Skalierungsaktivitäten nach oben im Zusammenhang mit Skalierungsrichtlinien vorübergehend deaktivieren, ohne die Skalierungsrichtlinien oder die zugehörigen CloudWatch -Alarme löschen zu müssen. Wenn Sie die Skalierung wieder aufnehmen, bewertet Application Auto Scaling Richtlinien mit Alarmschwellenwerten, die derzeit verletzt werden.

`ScheduledScalingSuspended`

- Application Auto Scaling initiiert nicht die Skalierungsaktionen, die für den Zeitraum der Aussetzung geplant sind. Wenn Sie die geplante Skalierung wieder aufnehmen, wertet Application Auto Scaling nur die geplanten Aktionen aus, deren Ausführungszeit noch nicht abgelaufen ist.

Skalierungsaktivitäten aussetzen und wieder aufnehmen

Sie können einzelne Skalierungsaktivitäten oder alle Skalierungsaktivitäten für Ihr skalierbares Ziel von Application Auto Scaling aussetzen und wieder aufnehmen.

> **Note**
>
> Der Kürze halber wird in diesen Beispielen gezeigt, wie die Skalierung für eine DynamoDB-Tabelle ausgesetzt und wieder aufgenommen wird. Um ein anderes skalierbares Ziel anzugeben, geben Sie seinen Namespace in `--service-namespace`, seine skalierbare Dimension in `--scalable-dimension` und seine Ressourcen-ID in `--resource-id` an. Weitere Informationen und Beispiele für die einzelnen Services finden Sie in den Themen unter AWS-Services die Sie mit Application Auto Scaling verwenden können.

So setzen Sie eine Skalierung aus

Öffnen Sie ein Befehlszeilenfenster und verwenden Sie den register-scalable-target-Befehl mit der `--suspended-state`-Option wie folgt.

Linux, macOS oder Unix

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table \
  --suspended-state file://config.json
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb --
scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table --
suspended-state file://config.json
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Wenn Sie nur horizontale Skalierungsaktivitäten nach unten aussetzen möchten, die von einer Skalierungsrichtlinie, ausgelöst werden, geben Sie Folgendes in der config.json-Datei an.

```
{
    "DynamicScalingInSuspended":true
```

```
}
```

Wenn Sie nur horizontale Skalierungsaktivitäten nach oben aussetzen möchten, die von einer Skalierungsrichtlinie, ausgelöst werden, geben Sie in der config.json-Datei Folgendes an.

```
{
    "DynamicScalingOutSuspended":true
}
```

Wenn Sie nur Skalierungen aussetzen möchten, die geplante Aktionen umfassen, geben Sie in der config.json-Datei Folgendes an.

```
{
    "ScheduledScalingSuspended":true
}
```

So setzen Sie alle Skalierungen aus

Verwenden Sie den register-scalable-targetBefehl mit der `--suspended-state` Option wie folgt.

Linux, macOS oder Unix

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table \
  --suspended-state file://config.json
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb --
scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table --
suspended-state file://config.json
```

In diesem Beispiel wird davon ausgegangen, dass die config.json-Datei die folgenden JSON-formatierten Parameter enthält.

```
{
    "DynamicScalingInSuspended":true,
    "DynamicScalingOutSuspended":true,
    "ScheduledScalingSuspended":true
}
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Ausgesetzte Skalierungsaktivitäten anzeigen

Mit dem describe-scalable-targets-Befehl können Sie für ein skalierbares Ziel bestimmen, welche Skalierungen sich in einem ausgesetzten Zustand befinden.

Linux, macOS oder Unix

```
aws application-autoscaling describe-scalable-targets --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table
```

Windows

```
aws application-autoscaling describe-scalable-targets --service-namespace dynamodb --
scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table
```

Es folgt eine Beispielausgabe.

```
{
    "ScalableTargets": [
        {
            "ServiceNamespace": "dynamodb",
            "ScalableDimension": "dynamodb:table:ReadCapacityUnits",
            "ResourceId": "table/my-table",
            "MinCapacity": 1,
            "MaxCapacity": 20,
            "SuspendedState": {
                "DynamicScalingOutSuspended": true,
                "DynamicScalingInSuspended": true,
                "ScheduledScalingSuspended": true
            },
            "CreationTime": 1558125758.957,
            "RoleARN": "arn:aws:iam::123456789012:role/aws-
service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable"
```

```
        }
    ]
}
```

Wiederaufnahme der Skalierungsaktivitäten

Wenn die Skalierungsaktivität fortgesetzt werden kann, ist dies mit dem register-scalable-target-Befehl möglich.

Mit dem folgenden Beispielbefehl werden alle Skalierungen für das angegebene skalierbare Ziel fortgesetzt.

Linux, macOS oder Unix

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table \
  --suspended-state file://config.json
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb --
scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table --
suspended-state file://config.json
```

In diesem Beispiel wird davon ausgegangen, dass die config.json-Datei die folgenden JSON-formatierten Parameter enthält.

```
{
    "DynamicScalingInSuspended":false,
    "DynamicScalingOutSuspended":false,
    "ScheduledScalingSuspended":false
}
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Skalierungsaktivitäten für Application Auto Scaling

Application Auto Scaling überwacht die CloudWatch Metriken Ihrer Skalierungsrichtlinie und leitet eine Skalierungsaktivität ein, wenn Schwellenwerte überschritten werden. Es initiiert auch Skalierungsaktivitäten, wenn Sie die maximale oder minimale Größe des skalierbaren Ziels ändern, entweder manuell oder nach einem Zeitplan.

Wenn eine Skalierungsaktivität auftritt, führt Application Auto Scaling eine der folgenden Aktionen aus:

- Erhöht die Kapazität des skalierbaren Ziels (als Hochskalieren bezeichnet)
- Verringert die Kapazität des skalierbaren Ziels (als Herunterskalieren bezeichnet)

Sie können die Skalierungsaktivitäten der letzten sechs Wochen abrufen.

Suchen Sie nach Skalierungsaktivitäten nach skalierbarem Ziel

Verwenden Sie den folgenden describe-scaling-activitiesBefehl, um die Skalierungsaktivitäten für ein bestimmtes skalierbares Ziel anzuzeigen.

Linux, macOS oder Unix

```
aws application-autoscaling describe-scaling-activities --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount --resource-id service/my-cluster/my-service
```

Windows

```
aws application-autoscaling describe-scaling-activities --service-namespace ecs --
scalable-dimension ecs:service:DesiredCount --resource-id service/my-cluster/my-service
```

Im Folgenden finden Sie eine Beispielantwort, bei der `StatusCode` den aktuellen Status der Aktivität und `StatusMessage` Informationen über den Status der Skalierungsaktivität enthält.

```
{
    "ScalingActivities": [
        {
```

```
            "ScalableDimension": "ecs:service:DesiredCount",
            "Description": "Setting desired count to 1.",
            "ResourceId": "service/my-cluster/my-service",
            "ActivityId": "e6c5f7d1-dbbb-4a3f-89b2-51f33e766399",
            "StartTime": 1462575838.171,
            "ServiceNamespace": "ecs",
            "EndTime": 1462575872.111,
            "Cause": "monitor alarm web-app-cpu-lt-25 in state ALARM triggered policy
 web-app-cpu-lt-25",
            "StatusMessage": "Successfully set desired count to 1. Change successfully
 fulfilled by ecs.",
            "StatusCode": "Successful"
        }
    ]
}
```

Eine Beschreibung der Felder in der Antwort finden Sie ScalingActivityin der Application Auto Scaling API-Referenz.

Die folgenden Statuscodes zeigen an, wann das Skalierungsereignis, das zur Skalierungsaktivität führt, einen abgeschlossenen Status erreicht:

- `Successful` – Die Skalierung wurde erfolgreich abgeschlossen
- `Overridden` – Die gewünschte Kapazität wurde durch ein neueres Skalierungsereignis aktualisiert
- `Unfulfilled` – Das Zeitlimit für die Skalierung wurde überschritten oder der Ziel-Service kann die Anfrage nicht erfüllen
- `Failed` – Die Skalierung ist mit einer Ausnahme fehlgeschlagen

> **Note**
>
> Die Skalierungsaktivität kann auch den Status `Pending` oder `InProgress` haben. Alle Skalierungsaktivitäten haben einen `Pending`-Status, bevor der Zielservice reagiert. Nachdem das Ziel reagiert hat, ändert sich der Status der Skalierungsaktivität zu `InProgress`.

Schließen Sie nicht skalierte Aktivitäten ein

Standardmäßig spiegeln die Skalierungsaktivitäten nicht die Zeiten wider, in denen Application Auto Scaling eine Entscheidung darüber trifft, ob keine Skalierung durchgeführt werden soll.

Nehmen wir beispielsweise an, dass ein Amazon-ECS-Service den maximalen Schwellenwert einer bestimmten Metrik überschreitet, aber die Anzahl der Aufgaben bereits die maximal zulässige Anzahl von Aufgaben erreicht. In diesem Fall wird Application Auto Scaling nicht die gewünschte Anzahl von Aufgaben aufskalieren.

Um Aktivitäten, die nicht skaliert sind (nicht skalierte Aktivitäten), in die Antwort aufzunehmen, fügen Sie dem Befehl die `--include-not-scaled-activities` Option hinzu. describe-scaling-activities

Linux, macOS oder Unix

```
aws application-autoscaling describe-scaling-activities --include-not-scaled-activities
 \
  --service-namespace ecs --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service
```

Windows

```
aws application-autoscaling describe-scaling-activities --include-not-scaled-activities
 --service-namespace ecs --scalable-dimension ecs:service:DesiredCount --resource-
id service/my-cluster/my-service
```

> **Note**
>
> Wenn dieser Befehl einen Fehler auslöst, stellen Sie sicher, dass Sie den Befehl AWS CLI lokal auf die neueste Version aktualisiert haben.

Um zu bestätigen, dass die Antwort die nicht skalierten Aktivitäten enthält, wird das `NotScaledReasons`-Element in der Ausgabe für einige, wenn nicht alle fehlgeschlagenen Skalierungsaktivitäten angezeigt.

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "ecs:service:DesiredCount",
            "Description": "Attempting to scale due to alarm triggered",
            "ResourceId": "service/my-cluster/my-service",
            "ActivityId": "4d759079-a31f-4d0c-8468-504c56e2eecf",
            "StartTime": 1664928867.915,
```

```
            "ServiceNamespace": "ecs",
            "Cause": "monitor alarm web-app-cpu-gt-75 in state ALARM triggered policy
 web-app-cpu-gt-75",
            "StatusCode": "Failed",
            "NotScaledReasons": [
                {
                    "Code": "AlreadyAtMaxCapacity",
                    "MaxCapacity": 4
                }
            ]
        }
    ]
}
```

Eine Beschreibung der Felder in der Antwort finden Sie ScalingActivityin der Application Auto Scaling API-Referenz.

Wenn eine nicht skalierte Aktivität zurückgegeben wird, können abhängig vom in `Code` aufgeführten Ursachencode Attribute wie `CurrentCapacity`, `MaxCapacity` und `MinCapacity` in der Antwort vorhanden sein.

Um große Mengen doppelter Einträge zu vermeiden, wird nur die erste Aktivität, die nicht skaliert wurde, im Verlauf der Skalierungsaktivitäten aufgezeichnet. Alle nachfolgenden nicht skalierten Aktivitäten generieren keine neuen Einträge, es sei denn, der Grund für die Nichtskalierung ändert sich.

Ursachencodes

Im Folgenden sind die Ursachencodes für eine nicht skalierte Aktivität aufgeführt.

Ursachencode	Definition			
AutoScalingAnticipatedFlapping	Der automatische Skalierungsalgorithmus hat beschlossen, keine Skalierungsaktion durchzuführen, da dies zu einem			

Ursachencode	Definition			
	Flattern führen würde. Flattern beschreibt eine Endlosschleife aus Auf- und Abwärtsskalieren. Das heißt, wenn eine Skalierungsaktion durchgeführt wird, würde sich der Metrikwert ändern, um eine weitere Skalierungsaktion in der umgekehrten Richtung zu starten.			
TargetServicePutResourceAsUnscalable	Der Zieldienst hat die Ressource vorübergehend in einen nicht skalierbaren Zustand versetzt. Application Auto Scaling versucht erneut zu skalieren, wenn die in der Skalierungsrichtlinie angegebenen Bedingungen für die automatische Skalierung erfüllt sind.			

Ursachencode	Definition			
AlreadyAtMaxCapacity	Die Skalierung wird durch die von Ihnen angegebene maximale Kapazität blockiert. Wenn Application Auto Scaling aufskalieren soll, müssen Sie die maximale Kapazität erhöhen.			
AlreadyAtMinCapacity	Die Skalierung wird durch die von Ihnen angegebene minimale Kapazität blockiert. Wenn Application Auto Scaling abskalieren soll, müssen Sie die Mindestkapazität verringern.			
AlreadyAtDesiredCapacity	Der automatische Skalierungsalgorithmus berechnete die geänderte Kapazität so, dass sie der aktuellen Kapazität entspricht.			

Auto Scaling von Überwachungsanwendungen

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung von Application Auto Scaling und Ihren anderen AWS Lösungen. Sie sollten Überwachungsdaten aus allen Teilen Ihrer AWS Lösung sammeln, damit Sie einen etwaigen Ausfall an mehreren Stellen leichter debuggen können. AWS bietet Überwachungstools, um Application Auto Scaling zu beobachten, zu melden, wenn etwas nicht stimmt, und bei Bedarf automatische Maßnahmen zu ergreifen.

Sie können die folgenden Funktionen verwenden, um Ihre AWS Ressourcen zu verwalten:

AWS CloudTrail

Mit AWS CloudTrail können Sie die Aufrufe der Application Auto Scaling API von oder in Ihrem Namen verfolgen AWS-Konto. CloudTrail speichert die Informationen in Protokolldateien im Amazon S3 S3-Bucket, den Sie angeben. Sie können feststellen, welche Benutzer und Konten Application Auto Scaling aufgerufen haben, von welcher Quell-IP-Adresse die Anrufe ausgingen und wann die Anrufe erfolgten. Weitere Informationen finden Sie unter Automatische Skalierungs-API-Aufrufe von Anwendungen protokollieren mit AWS CloudTrail.

> **Note**
>
> Informationen zu anderen AWS Diensten, die Ihnen bei der Protokollierung und Erfassung von Daten über Ihre Workloads helfen können, finden Sie im Leitfaden zur Protokollierung und Überwachung für Anwendungsbesitzer in der AWS Prescriptive Guidance.

Amazon CloudWatch

Amazon CloudWatch unterstützt Sie bei der Analyse von Protokollen und bei der Überwachung der Kennzahlen Ihrer AWS Ressourcen und gehosteten Anwendungen in Echtzeit. Sie können Kennzahlen erfassen und verfolgen, benutzerdefinierte Dashboards erstellen und Alarme festlegen, die Sie benachrichtigen oder Maßnahmen ergreifen, wenn eine bestimmte Metrik einen von Ihnen festgelegten Schwellenwert erreicht. Sie können beispielsweise die Ressourcennutzung CloudWatch verfolgen und Sie benachrichtigen, wenn die Auslastung sehr hoch ist oder wenn der Alarm der Metrik in den `INSUFFICIENT_DATA` Status übergegangen ist. Weitere Informationen finden Sie unter Überwachen Sie die Nutzung skalierbarer Ressourcen mit CloudWatch.

CloudWatch verfolgt auch AWS API-Nutzungsmetriken für Application Auto Scaling. Sie können diese Metriken verwenden, um Alarme zu konfigurieren, die Sie benachrichtigen, wenn Ihr API-Aufrufvolumen einen von Ihnen definierten Schwellenwert überschreitet. Weitere Informationen finden Sie unter AWS Nutzungsmetriken im CloudWatch Amazon-Benutzerhandbuch.

Amazon EventBridge

Amazon EventBridge ist ein serverloser Event-Bus-Service, der es einfach macht, Ihre Anwendungen mit Daten aus einer Vielzahl von Quellen zu verbinden. EventBridge liefert einen Stream von Echtzeitdaten aus Ihren eigenen Anwendungen, Software-as-a-Service (SaaS-) Anwendungen und AWS Diensten und leitet diese Daten an Ziele wie Lambda weiter. Auf diese Weise können Sie Ereignisse überwachen, die in Services auftreten, und ereignisgesteuerte Architekturen erstellen. Weitere Informationen finden Sie unter Überwachen von Application Auto Scaling Scaling-Ereignissen mithilfe von Amazon EventBridge.

AWS Health Dashboard

Das Health Dashboard (PHD) zeigt Informationen an und bietet auch Benachrichtigungen, die bei Änderungen im Zustand der AWS Ressourcen ausgelöst werden. Diese Informationen werden auf zweierlei Weise dargestellt: in einem Dashboard, das kürzliche und kommende Ereignisse nach Kategorie sortiert anzeigt, und in einem vollständigen Ereignisprotokoll, das alle Ereignisse der letzten 90 Tage enthält. Weitere Informationen finden Sie unter Erste Schritte mit Ihrem Health Dashboard.

Überwachen Sie die Nutzung skalierbarer Ressourcen mit CloudWatch

Mit Amazon CloudWatch erhalten Sie nahezu kontinuierlichen Einblick in Ihre Anwendungen über skalierbare Ressourcen hinweg. CloudWatch ist ein Monitoring-Service für AWS Ressourcen. Sie können CloudWatch damit Kennzahlen sammeln und verfolgen, Alarme einrichten und automatisch auf Änderungen Ihrer AWS Ressourcen reagieren. Sie können auch Dashboards erstellen, um die spezifischen Metriken oder Gruppen von Metriken zu überwachen, die Sie benötigen.

Wenn Sie mit den Diensten interagieren, die in Application Auto Scaling integriert sind, senden sie die in der folgenden Tabelle aufgeführten Metriken an CloudWatch. In CloudWatch werden die Metriken zuerst nach dem Dienst-Namespace und dann nach den verschiedenen Dimensionskombinationen innerhalb der einzelnen Namespaces gruppiert. Diese Metriken können Ihnen helfen, die Ressourcennutzung zu überwachen und die Kapazität Ihrer Anwendungen zu planen. Wenn der Workload Ihrer Anwendung nicht konstant ist, sollten Sie die Verwendung von Auto Scaling in

Betracht ziehen. Ausführliche Beschreibungen dieser Metriken finden Sie in der Dokumentation zur jeweiligen Metrik.

Inhalt

CloudWatch Metriken zur Überwachung der Ressourcennutzung

In der folgenden Tabelle sind die CloudWatch Metriken aufgeführt, die zur Überwachung der Ressourcennutzung verfügbar sind. Die Liste ist nicht vollständig, bietet Ihnen aber einen guten Ausgangspunkt. Wenn Sie diese Metriken nicht in der CloudWatch Konsole sehen, stellen Sie sicher, dass Sie die Einrichtung der Ressource abgeschlossen haben. Weitere Informationen finden Sie im CloudWatch Amazon-Benutzerhandbuch.

Skalierbare Ressource	Namespace	CloudWatch Metrik	Link zur Dokumentation
WorkSpaces Anwendungen			
Flotten	AWS/AppStream	Bezeichnung: AvailableCapacity Dimension: Flotte	WorkSpaces Metriken für Anwendungen
Flotten	AWS/AppStream	Bezeichnung: CapacityUtilization Dimension: Flotte	WorkSpaces Metriken für Anwendungen

Skalierbare Ressource	Namespace	CloudWatch Metrik	Link zur Dokumentation
Aurora			
Replikas	AWS/RDS	Name: CPUUtilization Abmessungen: DBClusterKennung, Rolle (READER)	Aurora-Metriken auf Clusterebene
Replikas	AWS/RDS	Name: DatabaseConnections Abmessungen: DBClusterKennung, Rolle (READER)	Aurora-Metriken auf Clusterebene
Amazon Comprehend			
Dokumentklassifizierungsendpunkte	AWS/Comprehend	Name: InferenceUtilization Dimension: EndpointArn	Endpoint-Metriken für Amazon Comprehend

Skalierbare Ressource	Namespace	CloudWatch Metrik	Link zur Dokumentation
Endpunkte der Entitätserkennung	AWS/Comprehend	Name: InferenceUtilization Dimension: EndpointArn	Endpoint-Metriken für Amazon Comprehend
DynamoDB			
Tabellen und globale sekundäre Indizes	AWS/DynamoDB	Name: ProvisionedReadCapacityUnits Abmessungen: TableName, GlobalSecondaryIndexName	DynamoDB-Metriken

Skalierbare Ressource	Namespace	CloudWatch Metrik	Link zur Dokumentation
Tabellen und globale sekundäre Indizes	AWS/DynamoDB	Name: ProvisionedWriteCapacityUnits Abmessungen: TableName, GlobalSecondaryIndexName	DynamoDB-Metriken
Tabellen und globale sekundäre Indizes	AWS/DynamoDB	Name: ConsumedReadCapacityUnits Abmessungen: TableName, GlobalSecondaryIndexName	DynamoDB-Metriken

Skalierbare Ressource	Namespace	CloudWatch Metrik	Link zur Dokumentation
Tabellen und globale sekundäre Indizes	AWS/DynamoDB	Name: ConsumedWriteCapacityUnits Abmessungen: TableName, GlobalSecondaryIndexName	DynamoDB-Metriken
Amazon ECS			
Dienstleistungen	AWS/ECS	Name: CPUUtilization Abmessungen: ClusterName, ServiceName	Amazon-ECS-Metriken

Skalierbare Ressource	Namespace	CloudWatch Metrik	Link zur Dokumentation
Dienstleistungen	AWS/ECS	Name: MemoryUtilization Abmessungen: ClusterName, ServiceName	Amazon-ECS-Metriken
Dienstleistungen	AWS/ApplicationELB	Name: RequestCountPerTarget Dimension: TargetGroup	Application-Load-Balancer-Metriken
ElastiCache			
Cluster (Replikationsgruppen)	AWS/ElastiCache	Bezeichnung: DatabaseMemoryUsageCountedForEvictPercentage Dimension: ReplicationGroupId	ElastiCache OSS-Metriken von Valkey und Redis

Skalierbare Ressource	Namespace	CloudWatch Metrik	Link zur Dokumentation
Cluster (Replikationsgruppen)	AWS/ElastiCache	Bezeichnung: DatabaseCapacityUsageCountedForEvictPercentage Dimension: ReplicationGroupId	ElastiCache OSS-Metriken von Valkey und Redis
Cluster (Replikationsgruppen)	AWS/ElastiCache	Name: MotorCPUUtilization Abmessungen: ReplicationGroupId, Rolle (primär)	ElastiCache OSS-Metriken von Valkey und Redis

Skalierbare Ressource	Namespace	CloudWatch Metrik	Link zur Dokumentation
Cluster (Replikationsgruppen)	AWS/ElastiCache	Name: EngineCPUUtilization Abmessungen: ReplicationGroupId, Rolle (Replik)	ElastiCache OSS-Metriken von Valkey und Redis
Cluster (Cache)	AWS/ElastiCache	Name: EngineCPUUtilization Abmessungen: CacheClusterId, Knoten	ElastiCache Memcached-Metriken
Cluster (Cache)	AWS/ElastiCache	Bezeichnung: DatabaseCapacityMemoryUsagePercentage Abmessungen: CacheClusterId	ElastiCache Memcached-Metriken

Skalierbare Ressource	Namespace	CloudWatch Metrik	Link zur Dokumentation
Amazon EMR			
Cluster	AWS/ElasticMapReduce	Bezeichnung: YARNMemory Available Percentage Dimension: ClusterId	Amazon-EMR-Metriken
Amazon Keyspaces			
Tabellen	AWS/Cassandra	Name: ProvisionedReadCapacityUnits Abmessungen: Keyspace, TableName	Amazon-Keyspaces-Metriken

Skalierbare Ressource	Namespace	CloudWatch Metrik	Link zur Dokumentation
Tabellen	AWS/Cassandra	Name: ProvisionedWriteCapacityUnits Abmessungen: Keyspace, TableName	Amazon-Keyspaces-Metriken
Tabellen	AWS/Cassandra	Name: ConsumedReadCapacityUnits Abmessungen: Keyspace, TableName	Amazon-Keyspaces-Metriken
Tabellen	AWS/Cassandra	Name: ConsumedWriteCapacityUnits Abmessungen: Keyspace, TableName	Amazon-Keyspaces-Metriken
Lambda			

Skalierbare Ressource	Namespace	CloudWatch Metrik	Link zur Dokumentation
Bereitgestellte Gleichzeitigkeit	AWS/Lambda	Name: ProvisionedConcurrencyUtilization Abmessungen: FunctionName, Ressource	Lambda-Funktionsmetriken
Amazon MSK			
Broker-Speicher	AWS/Kafka	Name: KafkaDataLogsDiskUsed Dimensionen: Clustername	Amazon-MSK-Metriken
Broker-Speicher	AWS/Kafka	Name: KafkaDataLogsDiskUsed Dimensionen: Clustername, Broker-ID	Amazon-MSK-Metriken

Skalierbare Ressource	Namespace	CloudWatch Metrik	Link zur Dokumentation
Neptune			
Cluster	AWS/Neptune	Name: CPUUtilization Abmessungen: DBClusterKennung, Rolle (READER)	Neptune-Metriken
SageMaker AI			
Endpunktvarianten	AWS/SageMaker	Bezeichnung: InvocationsPerInstance Abmessungen: EndpointName, VariantName	Aufrufmetriken

Skalierbare Ressource	Namespace	CloudWatch Metrik	Link zur Dokumentation
Inferenzkomponenten	AWS/SageMaker	Bezeichnung: InvocationsPerCopy Abmessungen: InferenceComponentName	Aufrufmetriken
Bereitgestellte Gleichzeitigkeit für einen Serverless-Endpunkt	AWS/SageMaker	Bezeichnung: ServerlessProvisionedConcurrencyUtilization Abmessungen: EndpointName, VariantName	Metriken für Serverless-Endgeräte
Amazon EC2-Spot-Flotte			

Skalierbare Ressource	Namespace	CloudWatch Metrik	Link zur Dokumentation
Spot Flotten	AWS/Spot EC2	Bezeichnung: CPUUtilization Dimension: FleetRequestId	Metriken für Spot-Flotten
Spot Flotten	AWS/Spot EC2	Bezeichnung: NetworkIn Dimension: FleetRequestId	Metriken für Spot-Flotten
Spot Flotten	AWS/Spot EC2	Bezeichnung: NetworkOut Dimension: FleetRequestId	Metriken für Spot-Flotten

Skalierbare Ressource	Namespace	CloudWatch Metrik	Link zur Dokumentation
Spot Flotten	AWS/ApplicationELB	Name: RequestCountPerTarget Dimension: TargetGroup	Application-Load-Balancer-Metriken

Vordefinierte Metriken für Skalierungsrichtlinien für die Zielverfolgung

In der folgenden Tabelle sind die vordefinierten Metriktypen aus der Application Auto Scaling API-Referenz mit ihren entsprechenden CloudWatch Metriknamen aufgeführt. Jede vordefinierte Metrik stellt eine Aggregation der Werte der zugrunde liegenden CloudWatch Metrik dar. Das Ergebnis ist die durchschnittliche Ressourcennutzung über einen Zeitraum von einer Minute, basierend auf einem Prozentsatz, sofern nicht anders angegeben. Die vordefinierten Metriken werden nur im Rahmen der Einrichtung von Skalierungsrichtlinien für die Zielverfolgung verwendet.

Weitere Informationen zu diesen Metriken finden Sie in der Dokumentation des von Ihnen verwendeten Service, die Sie in der Tabelle unter CloudWatch Metriken zur Überwachung der Ressourcennutzung finden.

Vordefinierter Metriktyp	CloudWatch Name der Metrik
WorkSpaces Anwendungen	
`AppStreamAverageCapacityUtilization`	CapacityUtilization
Aurora	
`RDSReaderAverageCPUUtilization`	CPUUtilization

Vordefinierter Metriktyp	CloudWatch Name der Metrik
`RDSReaderAverageDatabaseConnections`	DatabaseConnections[1]
Amazon Comprehend	
`ComprehendInferenceUtilization`	InferenceUtilization
DynamoDB	
`DynamoDBReadCapacityUtilization`	ProvisionedReadCapacityUnits, ConsumedReadCapacityUnits [2]
`DynamoDBWriteCapacityUtilization`	ProvisionedWriteCapacityUnits, ConsumedWriteCapacityUnits [2]
Amazon ECS	
`ECSServiceAverageCPUUtilization`	CPUUtilization
`ECSServiceAverageMemoryUtilization`	MemoryUtilization
`ALBRequestCountPerTarget`	RequestCountPerTarget[1]
ElastiCache	
`ElastiCacheDatabaseMemoryUsageCountedForEvictPercentage`	DatabaseMemoryUsageCountedForEvictPercentage
`ElastiCacheDatabaseCapacityUsageCountedForEvictPercentage`	DatabaseCapacityUsageCountedForEvictPercentage
`ElastiCachePrimaryEngineCPUUtilization`	Motor CPUUtilization
`ElastiCacheReplicaEngineCPUUtilization`	Motor CPUUtilization
`ElastiCacheEngineCPUUtilization`	Motor CPUUtilization

Vordefinierter Metriktyp	CloudWatch Name der Metrik
`ElastiCacheDatabaseMemoryUsagePercentage`	DatabaseMemoryUsagePercentage
Amazon Keyspaces	
`CassandraReadCapacityUtilization`	ProvisionedReadCapacityUnits, ConsumedReadCapacityUnits [2]
`CassandraWriteCapacityUtilization`	ProvisionedWriteCapacityUnits, ConsumedWriteCapacityUnits [2]
Lambda	
`LambdaProvisionedConcurrencyUtilization`	ProvisionedConcurrencyUtilization
Amazon MSK	
`KafkaBrokerStorageUtilization`	KafkaDataLogsDiskUsed
Neptune	
`NeptuneReaderAverageCPUUtilization`	CPUUtilization
SageMaker AI	
`SageMakerVariantInvocationsPerInstance`	InvocationsPerInstance[1]
`SageMakerInferenceComponentInvocationsPerCopy`	InvocationsPerCopy[1]
`SageMakerVariantProvisionedConcurrencyUtilization`	ServerlessProvisionedConcurrencyUtilization

Vordefinierter Metriktyp	CloudWatch Name der Metrik
`SageMakerInferenceComponentConcurrentRequestsPerCopyHighResolution`	ConcurrentRequestsPerCopy
`SageMakerVariantConcurrentRequestsPerModelHighResolution`	ConcurrentRequestsPerModel
Spot-Flotte	
`EC2SpotFleetRequestAverageCPUUtilization`	CPUUtilization[3]
`EC2SpotFleetRequestAverageNetworkIn`[3]	NetworkIn[1] [3]
`EC2SpotFleetRequestAverageNetworkOut`[3]	NetworkOut[1] [3]
`ALBRequestCountPerTarget`	RequestCountPerTarget[1]

[1]Metrik basiert auf einer Anzahl statt auf einem Prozentsatz.

[2] Für DynamoDB und Amazon Keyspaces sind die vordefinierten Metriken eine Aggregation von zwei CloudWatch Metriken, um die Skalierung auf der Grundlage des bereitgestellten Durchsatzverbrauchs zu unterstützen.

[3]Für eine optimale Skalierungsleistung sollte die detaillierte Überwachung von Amazon EC2 verwendet werden.

Metriken und Dimensionen für die prädiktive Skalierung

Der `AWS/ApplicationAutoScaling` Namespace umfasst die folgenden Metriken für Richtlinien zur vorausschauenden Skalierung. Diese Metriken sind mit einer Auflösung von einer Stunde verfügbar und können Ihnen helfen, die Prognosegenauigkeit zu bewerten, indem Sie prognostizierte Werte mit tatsächlichen Werten vergleichen.

Metrik	Description	Dimensionen
`PredictiveScalingLoadForecast`	Die Last, die voraussichtlich von Ihrer Anwendung generiert wird. Die Statistiken `Average`, `Minimum` und `Maximum` sind hilfreich, die Statistik `Sum` allerdings nicht. Berichtskriterien: Werden nach Erstellung der ursprüngliche Prognose gemeldet.	`ResourceId` , `ServiceNamespace` , `PolicyName` , `ScalableDimension` , `PairIndex`
`PredictiveScalingCapacityForecast`	Die voraussichtliche Kapazität, die zur Deckung des Anwendungsbedarfs erforderlich ist. Dies basiert auf der Lastprognose und dem Zielnutzungsgrad, auf dem Sie Ihre Application Auto Scaling Scaling-Ressourcen verwalten möchten. Die Statistiken `Average`, `Minimum` und `Maximum` sind hilfreich, die Statistik `Sum` allerdings nicht. Berichtskriterien: Werden nach Erstellung der ursprüngliche Prognose gemeldet.	`ResourceId` , `ServiceNamespace` , `PolicyName` , `ScalableDimension`
`PredictiveScalingMetricPairCorrelation`	Die Korrelation zwischen der Skalierungsmetrik und dem Durchschnitt der Lastmetrik pro Instance. Prädiktive Skalierung geht immer von einer hohen Korrelation aus. Wenn Sie also einen niedrigen Wert für diese Metrik beobachten, ist es besser, kein Metrikpaar zu verwenden. Die Statistiken `Average`, `Minimum` und `Maximum` sind hilfreich, die Statistik `Sum` allerdings nicht.	`ResourceId` , `ServiceNamespace` , `PolicyName` , `ScalableDimension` , `PairIndex`

Metrik	Description	Dimensionen
	Berichtskriterien: Werden nach Erstellung der ursprüngliche Prognose gemeldet.	

Automatische Skalierungs-API-Aufrufe von Anwendungen protokollieren mit AWS CloudTrail

Application Auto Scaling ist in einen Dienst integriert AWS CloudTrail, der eine Aufzeichnung der von einem Benutzer, einer Rolle oder einem ausgeführten Aktionen bereitstellt AWS-Service. CloudTrail erfasst API-Aufrufe für Application Auto Scaling als Ereignisse. Zu den erfassten Aufrufen gehören Aufrufe von AWS-Managementkonsole und Codeaufrufen zu den API-Vorgängen von Application Auto Scaling. Anhand der von gesammelten Informationen können Sie die Anfrage CloudTrail, die an Application Auto Scaling gestellt wurde, die IP-Adresse, von der aus die Anfrage gestellt wurde, wann sie gestellt wurde, und weitere Details ermitteln.

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Ob die Anforderung mit Anmeldeinformationen des Root-Benutzers oder des Benutzers gestellt wurde.
- Die Anforderung wurde im Namen eines IAM-Identity-Center-Benutzers erstellt.
- Gibt an, ob die Anforderung mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen Verbundbenutzer gesendet wurde.
- Ob die Anforderung aus einem anderen AWS-Service gesendet wurde.

CloudTrail ist in Ihrem Konto aktiv AWS-Konto , wenn Sie das Konto erstellen, und Sie haben automatisch Zugriff auf den CloudTrail Event-Verlauf. Der CloudTrail Ereignisverlauf bietet eine einsehbare, durchsuchbare, herunterladbare und unveränderliche Aufzeichnung der aufgezeichneten Verwaltungsereignisse der letzten 90 Tage in einem. AWS-Region Weitere Informationen finden Sie im AWS CloudTrail Benutzerhandbuch unter Arbeiten mit dem CloudTrail Ereignisverlauf. Für die Anzeige des Eventverlaufs CloudTrail fallen keine Gebühren an.

Für eine fortlaufende Aufzeichnung der Ereignisse in AWS-Konto den letzten 90 Tagen erstellen Sie einen Trail.

CloudTrail Pfade

Ein Trail ermöglicht CloudTrail die Übermittlung von Protokolldateien an einen Amazon S3 S3-Bucket. Alle mit dem erstellten Pfade AWS-Managementkonsole sind regionsübergreifend. Sie können mithilfe von AWS CLI einen Einzel-Region- oder einen Multi-Region-Trail erstellen. Es wird empfohlen, einen Trail mit mehreren Regionen zu erstellen, da Sie alle Aktivitäten AWS-Regionen in Ihrem Konto erfassen. Wenn Sie einen Einzel-Region-Trail erstellen, können Sie nur die Ereignisse anzeigen, die im AWS-Region des Trails protokolliert wurden. Weitere Informationen zu Trails finden Sie unter Erstellen eines Trails für Ihr AWS-Konto und Erstellen eines Trails für eine Organisation im AWS CloudTrail -Benutzerhandbuch.

Sie können eine Kopie Ihrer laufenden Verwaltungsereignisse kostenlos an Ihren Amazon S3 S3-Bucket senden, CloudTrail indem Sie einen Trail erstellen. Es fallen jedoch Amazon S3 S3-Speichergebühren an. Weitere Informationen zur CloudTrail Preisgestaltung finden Sie unter AWS CloudTrail Preise. Informationen zu Amazon-S3-Preisen finden Sie unter Amazon S3 – Preise.

Auto Scaling-Verwaltungsereignisse für Anwendungen in CloudTrail

Verwaltungsereignisse enthalten Informationen über Verwaltungsvorgänge, die an Ressourcen in Ihrem ausgeführt werden AWS-Konto. Sie werden auch als Vorgänge auf Steuerebene bezeichnet. CloudTrail Protokolliert standardmäßig Verwaltungsereignisse.

Application Auto Scaling protokolliert alle Operationen der Application Auto Scaling-Steuerebene als Verwaltungsereignisse. Eine Liste der Vorgänge auf der Application Auto Scaling-Steuerebene, die Application Auto Scaling protokolliert CloudTrail, finden Sie in der Application Auto Scaling Scaling-API-Referenz.

Beispiele Application Auto Scaling Scaling-Ereignisse

Ein Ereignis stellt eine einzelne Anfrage aus einer beliebigen Quelle dar und enthält Informationen über den angeforderten API-Vorgang, Datum und Uhrzeit des Vorgangs, Anforderungsparameter usw. CloudTrail Protokolldateien sind kein geordneter Stack-Trace der öffentlichen API-Aufrufe, sodass Ereignisse nicht in einer bestimmten Reihenfolge angezeigt werden.

Das folgende Beispiel zeigt ein CloudTrail Ereignis, das den `DescribeScalableTargets` Vorgang demonstriert.

```
{
    "eventVersion": "1.05",
```

```
    "userIdentity": {
        "type": "Root",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:root",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-08-21T17:05:42Z"
            }
        }
    },
    "eventTime": "2018-08-16T23:20:32Z",
    "eventSource": "autoscaling.amazonaws.com",
    "eventName": "DescribeScalableTargets",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "72.21.196.68",
    "userAgent": "EC2 Spot Console",
    "requestParameters": {
        "serviceNamespace": "ec2",
        "scalableDimension": "ec2:spot-fleet-request:TargetCapacity",
        "resourceIds": [
            "spot-fleet-request/sfr-05ceaf79-3ba2-405d-e87b-612857f1357a"
        ]
    },
    "responseElements": null,
    "additionalEventData": {
        "service": "application-autoscaling"
    },
    "requestID": "0737e2ea-fb2d-11e3-bfd8-99133058e7bb",
    "eventID": "3fcfb182-98f8-4744-bd45-b38835ab61cb",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```


Informationen zu CloudTrail Datensatzinhalten finden Sie im AWS CloudTrail Benutzerhandbuch unter CloudTrailDatensatzinhalt.

Application Auto Scaling RemoveAction ruft auf CloudWatch

In Ihrem AWS CloudTrail Protokoll wird möglicherweise angezeigt, dass Application Auto Scaling die CloudWatch `RemoveAction` API aufruft, wenn Application Auto Scaling anweist CloudWatch ,

die automatische Skalierungsaktion aus einem Alarm zu entfernen. Dies kann passieren, wenn Sie ein skalierbares Ziel abmelden, eine Skalierungsrichtlinie löschen oder wenn ein Alarm eine nicht existierende Skalierungsrichtlinie aufruft.

Überwachen von Application Auto Scaling Scaling-Ereignissen mithilfe von Amazon EventBridge

Amazon EventBridge, früher CloudWatch Events genannt, hilft Ihnen dabei, Ereignisse zu überwachen, die für Application Auto Scaling spezifisch sind, und Zielaktionen zu initiieren, die andere verwenden AWS-Services. Ereignisse von AWS-Services werden nahezu EventBridge in Echtzeit zugestellt.

Mithilfe können Sie Regeln erstellen EventBridge, die eingehenden Ereignissen entsprechen, und diese zur Verarbeitung an Ziele weiterleiten.

Weitere Informationen finden Sie unter Erste Schritte mit Amazon EventBridge im EventBridge Amazon-Benutzerhandbuch.

Application Auto Scaling-Ereignisse

Die folgenden Beispiele zeigen Ereignisse für Application Auto Scaling. Ereignisse werden auf die bestmögliche Weise ausgegeben.

Derzeit sind für Application Auto Scaling nur Ereignisse verfügbar, die spezifisch für Scaled to Max und API-Aufrufe über CloudTrail sind.

Event types (Ereignistypen)

- Ereignis für Statusänderung: skaliert auf Maximum
- Ereignisse für API-Aufrufe über CloudTrail

Ereignis für Statusänderung: skaliert auf Maximum

Das folgende Beispielereignis zeigt, dass Application Auto Scaling die Kapazität des skalierbaren Ziels auf seine maximale Größe erhöhte (aufskalierte). Wenn die Anforderungen erneut zunehmen, wird verhindert, dass Application Auto Scaling das Ziel noch weiter skaliert, da es bereits auf seine maximale Größe skaliert ist.

Im Objekt `detail` identifizieren die Werte für die Attribute `resourceId`, `serviceNamespace` und `scalableDimension` das skalierbare Ziel. Die Werte für die Attribute `newDesiredCapacity` und `oldDesiredCapacity` beziehen sich auf die neue Kapazität nach dem Aufskalierungsereignis und die ursprüngliche Kapazität vor dem Aufskalierungsereignis. Bei `maxCapacity` handelt es sich um die maximale Größe des skalierbaren Ziels.

```
{
  "version": "0",
  "id": "11112222-3333-4444-5555-666677778888",
  "detail-type": "Application Auto Scaling Scaling Activity State Change",
  "source": "aws.application-autoscaling",
  "account": "123456789012",
  "time": "2019-06-12T10:23:40Z",
  "region": "us-west-2",
  "resources": [],
  "detail": {
    "startTime": "2022-06-12T10:20:43Z",
    "endTime": "2022-06-12T10:23:40Z",
    "newDesiredCapacity": 8,
    "oldDesiredCapacity": 5,
    "minCapacity": 2,
    "maxCapacity": 8,
    "resourceId": "table/my-table",
    "scalableDimension": "dynamodb:table:WriteCapacityUnits",
    "serviceNamespace": "dynamodb",
    "statusCode": "Successful",
    "scaledToMax": true,
    "direction": "scale-out"
}
```

Um eine Regel zu erstellen, die alle `scaledToMax`-Statusänderungsereignisse für alle skalierbaren Ziele erfasst, verwenden Sie das folgende beispielhafte Ereignismuster.

```
{
  "source": [
    "aws.application-autoscaling"
  ],
  "detail-type": [
    "Application Auto Scaling Scaling Activity State Change"
  ],
  "detail": {
    "scaledToMax": [
```

```
      true
    ]
  }
}
```

Ereignisse für API-Aufrufe über CloudTrail

Ein Trail ist eine Konfiguration, die AWS CloudTrail verwendet wird, um Ereignisse als Protokolldateien an einen Amazon S3 S3-Bucket zu übertragen. CloudTrail Protokolldateien enthalten Protokolleinträge. Ein Ereignis stellt einen Protokolleintrag dar und enthält Informationen über die angeforderte Aktion, das Datum und die Uhrzeit der Aktion sowie Anforderungsparameter. Informationen zu den ersten Schritten finden Sie unter Erstellen eines Pfads im AWS CloudTrail Benutzerhandbuch. CloudTrail

Ereignisse, die über bereitgestellt werden CloudTrail `AWS API Call via CloudTrail`, haben den Wert für`detail-type`.

Das folgende Beispielereignis stellt einen CloudTrail Protokolldateieintrag dar, der zeigt, dass ein Konsolenbenutzer die Application Auto Scaling RegisterScalableTargetScaling-Aktion aufgerufen hat.

```
{
  "version": "0",
  "id": "99998888-7777-6666-5555-444433332222",
  "detail-type": "AWS API Call via CloudTrail",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "2022-07-13T16:50:15Z",
  "region": "us-west-2",
  "resources": [],
  "detail": {
    "eventVersion": "1.08",
    "userIdentity": {
      "type": "IAMUser",
      "principalId": "123456789012",
      "arn": "arn:aws:iam::123456789012:user/Bob",
      "accountId": "123456789012",
      "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
      "sessionContext": {
        "sessionIssuer": {
          "type": "Role",
          "principalId": "123456789012",
          "arn": "arn:aws:iam::123456789012:role/Admin",
```

```
          "accountId": "123456789012",
          "userName": "Admin"
        },
        "webIdFederationData": {},
        "attributes": {
          "creationDate": "2022-07-13T15:17:08Z",
          "mfaAuthenticated": "false"
        }
      }
    },
    "eventTime": "2022-07-13T16:50:15Z",
    "eventSource": "autoscaling.amazonaws.com",
    "eventName": "RegisterScalableTarget",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "EC2 Spot Console",
    "requestParameters": {
      "resourceId": "spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
      "serviceNamespace": "ec2",
      "scalableDimension": "ec2:spot-fleet-request:TargetCapacity",
      "minCapacity": 2,
      "maxCapacity": 10
    },
    "responseElements": null,
    "additionalEventData": {
      "service": "application-autoscaling"
    },
    "requestID": "e9caf887-8d88-11e5-a331-3332aa445952",
    "eventID": "49d14f36-6450-44a5-a501-b0fdcdfaeb98",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "sessionCredentialFromConsole": "true"
  }
}
```

Verwenden Sie das folgende Beispielereignismuster, um eine Regel zu erstellen, die auf allen DeleteScalingPolicyund DeregisterScalableTargetAPI-Aufrufen für alle skalierbaren Ziele basiert:

```
{
  "source": [
```

```
    "aws.autoscaling"
  ],
  "detail-type": [
    "AWS API Call via CloudTrail"
  ],
  "detail": {
    "eventSource": [
      "autoscaling.amazonaws.com"
    ],
    "eventName": [
       "DeleteScalingPolicy",
       "DeregisterScalableTarget"
    ],
    "additionalEventData": {
      "service": [
        "application-autoscaling"
      ]
    }
  }
}
```

Weitere Informationen zur Verwendung von CloudTrail finden Sie unterAutomatische Skalierungs-API-Aufrufe von Anwendungen protokollieren mit AWS CloudTrail.

Verwenden Sie diesen Dienst mit einem AWS SDK

AWS Software Development Kits (SDKs) sind für viele beliebte Programmiersprachen verfügbar. Jedes SDK bietet eine API, Codebeispiele und Dokumentation, die es Entwicklern erleichtern, Anwendungen in ihrer bevorzugten Sprache zu erstellen.

SDK-Dokumentation	Codebeispiele
AWS SDK für C++	AWS SDK für C++ Codebeispiele
AWS CLI	AWS CLI Code-Beispiele
AWS SDK für Go	AWS SDK für Go Code-Beispiele
AWS SDK für Java	AWS SDK für Java Code-Beispiele
AWS SDK für JavaScript	AWS SDK für JavaScript Code-Beispiele
AWS SDK für Kotlin	AWS SDK für Kotlin Code-Beispiele
AWS SDK für .NET	AWS SDK für .NET Code-Beispiele
AWS SDK für PHP	AWS SDK für PHP Code-Beispiele
AWS -Tools für PowerShell	AWS -Tools für PowerShell Code-Beispiele
AWS SDK für Python (Boto3)	AWS SDK für Python (Boto3) Code-Beispiele
AWS SDK für Ruby	AWS SDK für Ruby Code-Beispiele
AWS SDK für Rust	AWS SDK für Rust Code-Beispiele
AWS SDK für SAP ABAP	AWS SDK für SAP ABAP Code-Beispiele
AWS SDK für Swift	AWS SDK für Swift Code-Beispiele

Beispiel für die Verfügbarkeit

Sie können nicht finden, was Sie brauchen? Fordern Sie ein Codebeispiel an, indem Sie unten den Link Feedback geben auswählen.

Codebeispiele für Application Auto Scaling mit AWS SDKs

Die folgenden Codebeispiele zeigen, wie Application Auto Scaling mit einem AWS Software Development Kit (SDK) verwendet wird.

Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Während Aktionen Ihnen zeigen, wie Sie einzelne Service-Funktionen aufrufen, können Sie Aktionen im Kontext der zugehörigen Szenarien anzeigen.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Codebeispiele

Grundlegende Beispiele für Application Auto Scaling mit AWS SDKs

Die folgenden Codebeispiele zeigen, wie Sie die Grundlagen von Application Auto Scaling mit verwenden können AWS SDKs.

Beispiele

Aktionen für Application Auto Scaling mit AWS SDKs

Die folgenden Codebeispiele zeigen, wie einzelne Application Auto Scaling Scaling-Aktionen mit ausgeführt AWS SDKs werden. Jedes Beispiel enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Die folgenden Beispiele enthalten nur die am häufigsten verwendeten Aktionen. Eine komplette Liste finden Sie in der API-Referenz zum Application Auto Scaling.

Beispiele

- Verwendung DeleteScalingPolicy mit einem AWS SDK oder CLI
- Verwendung von DeleteScheduledAction mit einer CLI
- Verwendung von DeregisterScalableTarget mit einer CLI
- Verwendung von DescribeScalableTargets mit einer CLI
- Verwendung von DescribeScalingActivities mit einer CLI
- Verwendung DescribeScalingPolicies mit einem AWS SDK oder CLI
- Verwendung von DescribeScheduledActions mit einer CLI
- Verwendung von PutScalingPolicy mit einer CLI
- Verwendung von PutScheduledAction mit einer CLI
- Verwendung RegisterScalableTarget mit einem AWS SDK oder CLI

Verwendung **`DeleteScalingPolicy`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `DeleteScalingPolicy` verwendet wird.

CLI

AWS CLI

So löschen Sie eine Skalierungsrichtlinie

In diesem Beispiel wird eine Skalierungsrichtlinie für die Web-App des Amazon-ECS-Service gelöscht, die im Standard-Cluster ausgeführt wird.

Befehl:

```
aws application-autoscaling delete-scaling-policy --policy-name web-app-cpu-lt-25
 --scalable-dimension ecs:service:DesiredCount --resource-id service/default/web-
app --service-namespace ecs
```

- Einzelheiten zur API finden Sie DeleteScalingPolicyin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.applicationautoscaling.ApplicationAutoScalingClient;
import
 software.amazon.awssdk.services.applicationautoscaling.model.ApplicationAutoScalingExcep
import
 software.amazon.awssdk.services.applicationautoscaling.model.DeleteScalingPolicyRequest;
import
 software.amazon.awssdk.services.applicationautoscaling.model.DeregisterScalableTargetReq
```

```
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalableTargetsRequ
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalableTargetsResp
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalingPoliciesRequ
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalingPoliciesResp
import
 software.amazon.awssdk.services.applicationautoscaling.model.ScalableDimension;
import
 software.amazon.awssdk.services.applicationautoscaling.model.ServiceNamespace;

/**
 * Before running this Java V2 code example, set up your development environment,
 including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class DisableDynamoDBAutoscaling {
    public static void main(String[] args) {
        final String usage = """

            Usage:
               <tableId> <policyName>\s

            Where:
               tableId - The table Id value (for example, table/Music).\s
               policyName - The name of the policy (for example, $Music5-scaling-
policy).

            """;
        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        ApplicationAutoScalingClient appAutoScalingClient =
 ApplicationAutoScalingClient.builder()
            .region(Region.US_EAST_1)
```

```
            .build();

        ServiceNamespace ns = ServiceNamespace.DYNAMODB;
        ScalableDimension tableWCUs =
ScalableDimension.DYNAMODB_TABLE_WRITE_CAPACITY_UNITS;
        String tableId = args[0];
        String policyName = args[1];

        deletePolicy(appAutoScalingClient, policyName, tableWCUs, ns, tableId);
        verifyScalingPolicies(appAutoScalingClient, tableId, ns, tableWCUs);
        deregisterScalableTarget(appAutoScalingClient, tableId, ns, tableWCUs);
        verifyTarget(appAutoScalingClient, tableId, ns, tableWCUs);
    }

    public static void deletePolicy(ApplicationAutoScalingClient
appAutoScalingClient, String policyName, ScalableDimension tableWCUs,
ServiceNamespace ns, String tableId) {
        try {
            DeleteScalingPolicyRequest delSPRequest =
DeleteScalingPolicyRequest.builder()
                .policyName(policyName)
                .scalableDimension(tableWCUs)
                .serviceNamespace(ns)
                .resourceId(tableId)
                .build();

            appAutoScalingClient.deleteScalingPolicy(delSPRequest);
            System.out.println(policyName +" was deleted successfully.");

        } catch (ApplicationAutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
    }

    // Verify that the scaling policy was deleted
    public static void verifyScalingPolicies(ApplicationAutoScalingClient
appAutoScalingClient, String tableId, ServiceNamespace ns, ScalableDimension
tableWCUs) {
        DescribeScalingPoliciesRequest dscRequest =
DescribeScalingPoliciesRequest.builder()
            .scalableDimension(tableWCUs)
            .serviceNamespace(ns)
            .resourceId(tableId)
            .build();
```

```
        DescribeScalingPoliciesResponse response =
 appAutoScalingClient.describeScalingPolicies(dscRequest);
        System.out.println("DescribeScalableTargets result: ");
        System.out.println(response);
    }

    public static void deregisterScalableTarget(ApplicationAutoScalingClient
 appAutoScalingClient, String tableId, ServiceNamespace ns, ScalableDimension
 tableWCUs) {
        try {
            DeregisterScalableTargetRequest targetRequest =
 DeregisterScalableTargetRequest.builder()
                .scalableDimension(tableWCUs)
                .serviceNamespace(ns)
                .resourceId(tableId)
                .build();

            appAutoScalingClient.deregisterScalableTarget(targetRequest);
            System.out.println("The scalable target was deregistered.");

        } catch (ApplicationAutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
    }

    public static void verifyTarget(ApplicationAutoScalingClient
 appAutoScalingClient, String tableId, ServiceNamespace ns, ScalableDimension
 tableWCUs) {
        DescribeScalableTargetsRequest dscRequest =
 DescribeScalableTargetsRequest.builder()
            .scalableDimension(tableWCUs)
            .serviceNamespace(ns)
            .resourceIds(tableId)
            .build();

        DescribeScalableTargetsResponse response =
 appAutoScalingClient.describeScalableTargets(dscRequest);
        System.out.println("DescribeScalableTargets result: ");
        System.out.println(response);
    }
}
```

- Einzelheiten zur API finden Sie DeleteScalingPolicyin der AWS SDK for Java 2.x API-Referenz.

PowerShell

Tools für PowerShell V4

Beispiel 1: Dieses Cmdlet löscht die angegebene Skalierungsrichtlinie für ein skalierbares Ziel von Application Auto Scaling.

```
Remove-AASScalingPolicy -ServiceNamespace AppStream -PolicyName "default-scale-
out" -ResourceId fleet/Test -ScalableDimension appstream:fleet:DesiredCapacity
```

- Einzelheiten zur API finden Sie unter DeleteScalingPolicy AWS -Tools für PowerShellCmdlet-Referenz (V4).

Tools für V5 PowerShell

Beispiel 1: Dieses Cmdlet löscht die angegebene Skalierungsrichtlinie für ein skalierbares Ziel von Application Auto Scaling.

```
Remove-AASScalingPolicy -ServiceNamespace AppStream -PolicyName "default-scale-
out" -ResourceId fleet/Test -ScalableDimension appstream:fleet:DesiredCapacity
```

- Einzelheiten zur API finden Sie unter DeleteScalingPolicy AWS -Tools für PowerShellCmdlet-Referenz (V5).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Dienst mit einem AWS SDK Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung von **`DeleteScheduledAction`** mit einer CLI

Die folgenden Code-Beispiele zeigen, wie `DeleteScheduledAction` verwendet wird.

CLI

AWS CLI

Löschen einer geplanten Aktion

Das folgende `delete-scheduled-action` Beispiel löscht die angegebene geplante Aktion aus der angegebenen Amazon AppStream 2.0-Flotte:

```
aws application-autoscaling delete-scheduled-action \
    --service-namespace appstream \
    --scalable-dimension appstream:fleet:DesiredCapacity \
    --resource-id fleet/sample-fleet \
    --scheduled-action-name my-recurring-action
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Geplante Skalierung im Benutzerhandbuch zu Application Auto Scaling.

- Einzelheiten zur API finden Sie DeleteScheduledActionin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell V4

Beispiel 1: Dieses Cmdlet löscht die angegebene geplante Aktion für ein skalierbares Ziel von Application Auto Scaling.

```
Remove-AASScheduledAction -ServiceNamespace AppStream -ScheduledActionName
 WeekDaysFleetScaling -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity
```

Ausgabe:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-AASScheduledAction (DeleteScheduledAction)" on
 target "WeekDaysFleetScaling".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
```

- Einzelheiten zur API finden Sie unter DeleteScheduledAction AWS -Tools für PowerShellCmdlet-Referenz (V4).

Tools für V5 PowerShell

Beispiel 1: Dieses Cmdlet löscht die angegebene geplante Aktion für ein skalierbares Ziel von Application Auto Scaling.

```
Remove-AASScheduledAction -ServiceNamespace AppStream -ScheduledActionName
 WeekDaysFleetScaling -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity
```

Ausgabe:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-AASScheduledAction (DeleteScheduledAction)" on
 target "WeekDaysFleetScaling".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
```

- Einzelheiten zur API finden Sie unter DeleteScheduledAction AWS -Tools für PowerShellCmdlet-Referenz (V5).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Dienst mit einem AWS SDK Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung von **`DeregisterScalableTarget`** mit einer CLI

Die folgenden Code-Beispiele zeigen, wie `DeregisterScalableTarget` verwendet wird.

CLI

AWS CLI

So registrieren Sie ein skalierbares Ziel

In diesem Beispiel wird die Registrierung eines skalierbaren Ziels für einen Amazon ECS-Service mit dem Namen Web-App aufgehoben, der im Standard-Cluster ausgeführt wird.

Befehl:

```
aws application-autoscaling deregister-scalable-target --service-namespace ecs --
scalable-dimension ecs:service:DesiredCount --resource-id service/default/web-app
```

In diesem Beispiel wird die Registrierung eines skalierbaren Ziels für eine benutzerdefinierte Ressource aufgehoben. Die custom-resource-id .txt-Datei enthält eine Zeichenfolge, die die Ressourcen-ID identifiziert. Bei einer benutzerdefinierten Ressource ist dies der Pfad zu der benutzerdefinierten Ressource über Ihren Amazon API Gateway Gateway-Endpunkt.

Befehl:

```
aws application-autoscaling deregister-scalable-target --
service-namespace custom-resource --scalable-dimension custom-
resource:ResourceType:Property --resource-id file://~/custom-resource-id.txt
```

Inhalt der custom-resource-id TXT-Datei:

```
https://example.execute-api.us-west-2.amazonaws.com/prod/
scalableTargetDimensions/1-23456789
```

- Einzelheiten zur API finden Sie DeregisterScalableTargetin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell V4

Beispiel 1: Dieses Cmdlet hebt die Registrierung eines skalierbaren Ziels von Application Auto Scaling auf. Durch das Aufheben der Registrierung eines skalierbaren Ziels werden die zugehörigen Skalierungsrichtlinien gelöscht.

```
Remove-AASScalableTarget -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity -ServiceNamespace AppStream
```

Ausgabe:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-AASScalableTarget (DeregisterScalableTarget)" on
 target "fleet/MyFleet".
```

```
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
```

- Einzelheiten zur API finden Sie unter DeregisterScalableTarget AWS -Tools für PowerShellCmdlet-Referenz (V4).

Tools für V5 PowerShell

Beispiel 1: Dieses Cmdlet hebt die Registrierung eines skalierbaren Ziels von Application Auto Scaling auf. Durch das Aufheben der Registrierung eines skalierbaren Ziels werden die zugehörigen Skalierungsrichtlinien gelöscht.

```
Remove-AASScalableTarget -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity -ServiceNamespace AppStream
```

Ausgabe:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-AASScalableTarget (DeregisterScalableTarget)" on
 target "fleet/MyFleet".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
```

- Einzelheiten zur API finden Sie unter DeregisterScalableTarget AWS -Tools für PowerShellCmdlet-Referenz (V5).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Dienst mit einem AWS SDK Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung von **`DescribeScalableTargets`** mit einer CLI

Die folgenden Code-Beispiele zeigen, wie `DescribeScalableTargets` verwendet wird.

CLI

AWS CLI

So beschreiben Sie skalierbare Ziele

Das folgende Beispiel für `describe-scalable-targets` beschreibt die skalierbaren Ziele für den `ecs`-Service-Namespace.

```
aws application-autoscaling describe-scalable-targets \
    --service-namespace ecs
```

Ausgabe:

```
{
    "ScalableTargets": [
        {
            "ServiceNamespace": "ecs",
            "ScalableDimension": "ecs:service:DesiredCount",
            "ResourceId": "service/default/web-app",
            "MinCapacity": 1,
            "MaxCapacity": 10,
            "RoleARN": "arn:aws:iam::123456789012:role/
aws-service-role/ecs.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_ECSService",
            "CreationTime": 1462558906.199,
            "SuspendedState": {
                "DynamicScalingOutSuspended": false,
                "ScheduledScalingSuspended": false,
                "DynamicScalingInSuspended": false
            },
            "ScalableTargetARN": "arn:aws:application-autoscaling:us-
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
        }
    ]
}
```

Weitere Informationen finden Sie unter AWS -Services, die Sie mit Application Auto Scaling nutzen können im Benutzerhandbuch für Application Auto Scaling.

- Einzelheiten zur API finden Sie DescribeScalableTargetsin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell V4

Beispiel 1: In diesem Beispiel werden Informationen zu den skalierbaren Zielen von Auto Scaling für Anwendungen im angegebenen Namespace bereitgestellt.

```
Get-AASScalableTarget -ServiceNamespace "AppStream"
```

Ausgabe:

```
CreationTime      : 11/7/2019 2:30:03 AM
MaxCapacity       : 5
MinCapacity       : 1
ResourceId        : fleet/Test
RoleARN           : arn:aws:iam::012345678912:role/aws-
service-role/appstream.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_AppStreamFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ServiceNamespace  : appstream
SuspendedState    : Amazon.ApplicationAutoScaling.Model.SuspendedState
```

- Einzelheiten zur API finden Sie unter DescribeScalableTargets AWS -Tools für PowerShellCmdlet-Referenz (V4).

Tools für V5 PowerShell

Beispiel 1: In diesem Beispiel werden Informationen zu den skalierbaren Zielen von Auto Scaling für Anwendungen im angegebenen Namespace bereitgestellt.

```
Get-AASScalableTarget -ServiceNamespace "AppStream"
```

Ausgabe:

```
CreationTime      : 11/7/2019 2:30:03 AM
MaxCapacity       : 5
MinCapacity       : 1
ResourceId        : fleet/Test
RoleARN           : arn:aws:iam::012345678912:role/aws-
service-role/appstream.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_AppStreamFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ServiceNamespace  : appstream
SuspendedState    : Amazon.ApplicationAutoScaling.Model.SuspendedState
```

- Einzelheiten zur API finden Sie unter DescribeScalableTargets AWS -Tools für PowerShellCmdlet-Referenz (V5).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Dienst mit einem AWS SDK Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung von **`DescribeScalingActivities`** mit einer CLI

Die folgenden Code-Beispiele zeigen, wie `DescribeScalingActivities` verwendet wird.

CLI

AWS CLI

Beispiel 1: So beschreiben Sie Skalierungsaktivitäten für den angegebenen Amazon-ECS-Service

Das folgende Beispiel für `describe-scaling-activities` beschreibt die Skalierungsaktivitäten für einen Amazon-ECS-Service mit dem Namen `web-app`, der im `default`-Cluster ausgeführt wird. Die Ausgabe zeigt eine Skalierungsaktivität, die durch eine Skalierungsrichtlinie initiiert wurde.

```
aws application-autoscaling describe-scaling-activities \
    --service-namespace ecs \
    --resource-id service/default/web-app
```

Ausgabe:

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "ecs:service:DesiredCount",
            "Description": "Setting desired count to 1.",
            "ResourceId": "service/default/web-app",
            "ActivityId": "e6c5f7d1-dbbb-4a3f-89b2-51f33e766399",
            "StartTime": 1462575838.171,
            "ServiceNamespace": "ecs",
            "EndTime": 1462575872.111,
            "Cause": "monitor alarm web-app-cpu-lt-25 in state ALARM triggered
 policy web-app-cpu-lt-25",
            "StatusMessage": "Successfully set desired count to 1. Change
 successfully fulfilled by ecs.",
            "StatusCode": "Successful"
        }
```

```
    ]
}
```

Weitere Informationen finden Sie unter Skalierungsaktivitäten für Application Auto Scaling im Benutzerhandbuch für Application Auto Scaling.

Beispiel 2: So beschreiben Sie Skalierungsaktivitäten für die angegebene DynamoDB-Tabelle

Das folgende Beispiel für `describe-scaling-activities` beschreibt die Skalierungsaktivitäten für eine DynamoDB-Tabelle mit dem Namen `TestTable`. Die Ausgabe zeigt Skalierungsaktivitäten, die durch zwei verschiedene geplante Aktionen ausgelöst wurden.

```
aws application-autoscaling describe-scaling-activities \
    --service-namespace dynamodb \
    --resource-id table/TestTable
```

Ausgabe:

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 10.",
            "ResourceId": "table/my-table",
            "ActivityId": "4d1308c0-bbcf-4514-a673-b0220ae38547",
            "StartTime": 1561574415.086,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574449.51,
            "Cause": "maximum capacity was set to 10",
            "StatusMessage": "Successfully set write capacity units to 10. Change
 successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 5 and max capacity to 10",
            "ResourceId": "table/my-table",
            "ActivityId": "f2b7847b-721d-4e01-8ef0-0c8d3bacc1c7",
            "StartTime": 1561574414.644,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-second-scheduled-action was
 triggered",
```

```
            "StatusMessage": "Successfully set min capacity to 5 and max capacity
 to 10",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 15.",
            "ResourceId": "table/my-table",
            "ActivityId": "d8ea4de6-9eaa-499f-b466-2cc5e681ba8b",
            "StartTime": 1561574108.904,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574140.255,
            "Cause": "minimum capacity was set to 15",
            "StatusMessage": "Successfully set write capacity units to 15. Change
 successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 15 and max capacity to 20",
            "ResourceId": "table/my-table",
            "ActivityId": "3250fd06-6940-4e8e-bb1f-d494db7554d2",
            "StartTime": 1561574108.512,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-first-scheduled-action was
 triggered",
            "StatusMessage": "Successfully set min capacity to 15 and max
 capacity to 20",
            "StatusCode": "Successful"
        }
    ]
}
```

Weitere Informationen finden Sie unter Skalierungsaktivitäten für Application Auto Scaling im Benutzerhandbuch für Application Auto Scaling.

- Einzelheiten zur API finden Sie DescribeScalingActivitiesin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell V4

Beispiel 1: Liefert beschreibende Informationen zu den Skalierungsaktivitäten im angegebenen Service-Namespace aus den letzten sechs Wochen.

```
Get-AASScalingActivity -ServiceNamespace AppStream
```

Ausgabe:

```
ActivityId        : 2827409f-b639-4cdb-a957-8055d5d07434
Cause             : monitor alarm Appstream2-MyFleet-default-scale-in-Alarm in
 state ALARM triggered policy default-scale-in
Description       : Setting desired capacity to 2.
Details           :
EndTime           : 12/14/2019 11:32:49 AM
ResourceId        : fleet/MyFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ServiceNamespace  : appstream
StartTime         : 12/14/2019 11:32:14 AM
StatusCode        : Successful
StatusMessage     : Successfully set desired capacity to 2. Change successfully
 fulfilled by appstream.
```

- Einzelheiten zur API finden Sie unter DescribeScalingActivities AWS -Tools für PowerShellCmdlet-Referenz (V4).

Tools für V5 PowerShell

Beispiel 1: Liefert beschreibende Informationen zu den Skalierungsaktivitäten im angegebenen Service-Namespace aus den letzten sechs Wochen.

```
Get-AASScalingActivity -ServiceNamespace AppStream
```

Ausgabe:

```
ActivityId        : 2827409f-b639-4cdb-a957-8055d5d07434
Cause             : monitor alarm Appstream2-MyFleet-default-scale-in-Alarm in
 state ALARM triggered policy default-scale-in
Description       : Setting desired capacity to 2.
Details           :
EndTime           : 12/14/2019 11:32:49 AM
```

```
ResourceId        : fleet/MyFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ServiceNamespace  : appstream
StartTime         : 12/14/2019 11:32:14 AM
StatusCode        : Successful
StatusMessage     : Successfully set desired capacity to 2. Change successfully
 fulfilled by appstream.
```

- Einzelheiten zur API finden Sie unter DescribeScalingActivities AWS -Tools für PowerShellCmdlet-Referenz (V5).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Dienst mit einem AWS SDK Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DescribeScalingPolicies`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `DescribeScalingPolicies` verwendet wird.

CLI

AWS CLI

So beschreiben Sie Skalierungsrichtlinien

Dieser Beispielbefehl beschreibt die Skalierungsrichtlinien für den ECS-Service-Namespace.

Befehl:

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs
```

Ausgabe:

```
{
    "ScalingPolicies": [
        {
            "PolicyName": "web-app-cpu-gt-75",
            "ScalableDimension": "ecs:service:DesiredCount",
            "ResourceId": "service/default/web-app",
            "CreationTime": 1462561899.23,
            "StepScalingPolicyConfiguration": {
                "Cooldown": 60,
```

```
                "StepAdjustments": [
                    {
                        "ScalingAdjustment": 200,
                        "MetricIntervalLowerBound": 0.0
                    }
                ],
                "AdjustmentType": "PercentChangeInCapacity"
            },
            "PolicyARN": "arn:aws:autoscaling:us-
west-2:012345678910:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/
ecs/service/default/web-app:policyName/web-app-cpu-gt-75",
            "PolicyType": "StepScaling",
            "Alarms": [
                {
                    "AlarmName": "web-app-cpu-gt-75",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:web-app-cpu-gt-75"
                }
            ],
            "ServiceNamespace": "ecs"
        },
        {
            "PolicyName": "web-app-cpu-lt-25",
            "ScalableDimension": "ecs:service:DesiredCount",
            "ResourceId": "service/default/web-app",
            "CreationTime": 1462562575.099,
            "StepScalingPolicyConfiguration": {
                "Cooldown": 1,
                "StepAdjustments": [
                    {
                        "ScalingAdjustment": -50,
                        "MetricIntervalUpperBound": 0.0
                    }
                ],
                "AdjustmentType": "PercentChangeInCapacity"
            },
            "PolicyARN": "arn:aws:autoscaling:us-
west-2:012345678910:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/
ecs/service/default/web-app:policyName/web-app-cpu-lt-25",
            "PolicyType": "StepScaling",
            "Alarms": [
                {
                    "AlarmName": "web-app-cpu-lt-25",
```

```
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:web-app-cpu-lt-25"
                }
            ],
            "ServiceNamespace": "ecs"
        }
    ]
}
```

- Einzelheiten zur API finden Sie DescribeScalingPoliciesin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell V4

Beispiel 1: Dieses Cmdlet beschreibt die Richtlinien für Application Auto Scaling für den angegebenen Service-Namespace.

```
Get-AASScalingPolicy -ServiceNamespace AppStream
```

Ausgabe:

```
Alarms                                    : {Appstream2-LabFleet-default-scale-
out-Alarm}
CreationTime                              : 9/3/2019 2:48:15 AM
PolicyARN                                 : arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:5659b069-b5cd-4af1-9f7f-3e956d36233e:resource/
appstream/fleet/LabFleet:
                                            policyName/default-scale-out
PolicyName                                : default-scale-out
PolicyType                                : StepScaling
ResourceId                                : fleet/LabFleet
ScalableDimension                         : appstream:fleet:DesiredCapacity
ServiceNamespace                          : appstream
StepScalingPolicyConfiguration            :
 Amazon.ApplicationAutoScaling.Model.StepScalingPolicyConfiguration
TargetTrackingScalingPolicyConfiguration :

Alarms                                    : {Appstream2-LabFleet-default-scale-in-
Alarm}
CreationTime                              : 9/3/2019 2:48:15 AM
```

```
PolicyARN                                   : arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:5659b069-b5cd-4af1-9f7f-3e956d36233e:resource/
appstream/fleet/LabFleet:
                                              policyName/default-scale-in
PolicyName                                  : default-scale-in
PolicyType                                  : StepScaling
ResourceId                                  : fleet/LabFleet
ScalableDimension                           : appstream:fleet:DesiredCapacity
ServiceNamespace                            : appstream
StepScalingPolicyConfiguration              :
 Amazon.ApplicationAutoScaling.Model.StepScalingPolicyConfiguration
TargetTrackingScalingPolicyConfiguration :
```

- Einzelheiten zur API finden Sie unter DescribeScalingPolicies AWS -Tools für PowerShellCmdlet-Referenz (V4).

Tools für V5 PowerShell

Beispiel 1: Dieses Cmdlet beschreibt die Richtlinien für Application Auto Scaling für den angegebenen Service-Namespace.

```
Get-AASScalingPolicy -ServiceNamespace AppStream
```

Ausgabe:

```
Alarms                                      : {Appstream2-LabFleet-default-scale-
out-Alarm}
CreationTime                                : 9/3/2019 2:48:15 AM
PolicyARN                                   : arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:5659b069-b5cd-4af1-9f7f-3e956d36233e:resource/
appstream/fleet/LabFleet:
                                              policyName/default-scale-out
PolicyName                                  : default-scale-out
PolicyType                                  : StepScaling
ResourceId                                  : fleet/LabFleet
ScalableDimension                           : appstream:fleet:DesiredCapacity
ServiceNamespace                            : appstream
StepScalingPolicyConfiguration              :
 Amazon.ApplicationAutoScaling.Model.StepScalingPolicyConfiguration
TargetTrackingScalingPolicyConfiguration :

Alarms                                      : {Appstream2-LabFleet-default-scale-in-
Alarm}
```

```
CreationTime                                : 9/3/2019 2:48:15 AM
PolicyARN                                   : arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:5659b069-b5cd-4af1-9f7f-3e956d36233e:resource/
appstream/fleet/LabFleet:
                                              policyName/default-scale-in
PolicyName                                  : default-scale-in
PolicyType                                  : StepScaling
ResourceId                                  : fleet/LabFleet
ScalableDimension                           : appstream:fleet:DesiredCapacity
ServiceNamespace                            : appstream
StepScalingPolicyConfiguration              :
 Amazon.ApplicationAutoScaling.Model.StepScalingPolicyConfiguration
TargetTrackingScalingPolicyConfiguration :
```

- Einzelheiten zur API finden Sie unter DescribeScalingPolicies AWS -Tools für PowerShellCmdlet-Referenz (V5).

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
async fn show_policies(client: &Client) -> Result<(), Error> {
    let response = client
        .describe_scaling_policies()
        .service_namespace(ServiceNamespace::Ec2)
        .send()
        .await?;
    println!("Auto Scaling Policies:");
    for policy in response.scaling_policies() {
        println!("{:?}\n", policy);
    }
    println!("Next token: {:?}", response.next_token());

    Ok(())
}
```

- Einzelheiten zur API finden Sie DescribeScalingPoliciesin der API-Referenz zum AWS SDK für Rust.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden Sie diesen Dienst mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung von **`DescribeScheduledActions`** mit einer CLI

Die folgenden Code-Beispiele zeigen, wie `DescribeScheduledActions` verwendet wird.

CLI

AWS CLI

So beschreiben Sie geplante Aktionen

Das folgende Beispiel für `describe-scheduled-actions` zeigt Details zu den geplanten Aktionen für den angegebenen Service-Namespace:

```
aws application-autoscaling describe-scheduled-actions \
    --service-namespace dynamodb
```

Ausgabe:

```
{
    "ScheduledActions": [
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Schedule": "at(2019-05-20T18:35:00)",
            "ResourceId": "table/my-table",
            "CreationTime": 1561571888.361,
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:2d36aa3b-cdf9-4565-
b290-81db519b227d:resource/dynamodb/table/my-table:scheduledActionName/my-first-
scheduled-action",
            "ScalableTargetAction": {
                "MinCapacity": 15,
                "MaxCapacity": 20
            },
```

```
            "ScheduledActionName": "my-first-scheduled-action",
            "ServiceNamespace": "dynamodb"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Schedule": "at(2019-05-20T18:40:00)",
            "ResourceId": "table/my-table",
            "CreationTime": 1561571946.021,
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:2d36aa3b-cdf9-4565-
b290-81db519b227d:resource/dynamodb/table/my-table:scheduledActionName/my-second-
scheduled-action",
            "ScalableTargetAction": {
                "MinCapacity": 5,
                "MaxCapacity": 10
            },
            "ScheduledActionName": "my-second-scheduled-action",
            "ServiceNamespace": "dynamodb"
        }
    ]
}
```

Weitere Informationen finden Sie unter Geplante Skalierung im Benutzerhandbuch zu Application Auto Scaling.

- Einzelheiten zur API finden Sie DescribeScheduledActionsin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell V4

Beispiel 1: Dieses Cmdlet listet die für die Auto-Scaling-Gruppe geplanten Aktionen auf, die noch nicht ausgeführt wurden oder deren Endzeit noch nicht erreicht ist.

```
Get-AASScheduledAction -ServiceNamespace AppStream
```

Ausgabe:

```
CreationTime         : 12/22/2019 9:25:52 AM
EndTime              : 1/1/0001 12:00:00 AM
ResourceId           : fleet/MyFleet
ScalableDimension    : appstream:fleet:DesiredCapacity
```

```
ScalableTargetAction : Amazon.ApplicationAutoScaling.Model.ScalableTargetAction
Schedule             : cron(0 0 8 ? * MON-FRI *)
ScheduledActionARN   : arn:aws:autoscaling:us-
west-2:012345678912:scheduledAction:4897ca24-3caa-4bf1-8484-851a089b243c:resource/
appstream/fleet/MyFleet:scheduledActionName
                       /WeekDaysFleetScaling
ScheduledActionName  : WeekDaysFleetScaling
ServiceNamespace     : appstream
StartTime            : 1/1/0001 12:00:00 AM
```

- Einzelheiten zur API finden Sie unter DescribeScheduledActions AWS -Tools für PowerShellCmdlet-Referenz (V4).

Tools für V5 PowerShell

Beispiel 1: Dieses Cmdlet listet die für die Auto-Scaling-Gruppe geplanten Aktionen auf, die noch nicht ausgeführt wurden oder deren Endzeit noch nicht erreicht ist.

```
Get-AASScheduledAction -ServiceNamespace AppStream
```

Ausgabe:

```
CreationTime         : 12/22/2019 9:25:52 AM
EndTime              : 1/1/0001 12:00:00 AM
ResourceId           : fleet/MyFleet
ScalableDimension    : appstream:fleet:DesiredCapacity
ScalableTargetAction : Amazon.ApplicationAutoScaling.Model.ScalableTargetAction
Schedule             : cron(0 0 8 ? * MON-FRI *)
ScheduledActionARN   : arn:aws:autoscaling:us-
west-2:012345678912:scheduledAction:4897ca24-3caa-4bf1-8484-851a089b243c:resource/
appstream/fleet/MyFleet:scheduledActionName
                       /WeekDaysFleetScaling
ScheduledActionName  : WeekDaysFleetScaling
ServiceNamespace     : appstream
StartTime            : 1/1/0001 12:00:00 AM
```

- Einzelheiten zur API finden Sie unter DescribeScheduledActions AWS -Tools für PowerShellCmdlet-Referenz (V5).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Dienst mit einem AWS SDK Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung von **`PutScalingPolicy`** mit einer CLI

Die folgenden Code-Beispiele zeigen, wie `PutScalingPolicy` verwendet wird.

CLI

AWS CLI

Beispiel 1: So wenden Sie eine Skalierungsrichtlinie für die Ziel-Nachverfolgung mit einer vordefinierten Metrikspezifikation an

Das folgende Beispiel für `put-scaling-policy` wendet eine Skalierungsrichtlinie zur Zielverfolgung mit einer vordefinierten Metrikspezifikation auf einen Amazon-ECS-Service mit dem Namen Web-App im Standard-Cluster an. Die Richtlinie hält die durchschnittliche CPU-Auslastung des Service bei 75 Prozent, mit Aufskalierungs- und Abskalierungs-Ruhephasen von 60 Sekunden. Die Ausgabe enthält die Namen ARNs und der beiden CloudWatch Alarme, die in Ihrem Namen erstellt wurden.

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \
--scalable-dimension ecs:service:DesiredCount \
--resource-id service/default/web-app \
--policy-name cpu75-target-tracking-scaling-policy --policy-
type TargetTrackingScaling \
--target-tracking-scaling-policy-configuration file://config.json
```

In diesem Beispiel wird davon ausgegangen, dass Sie im aktuellen Verzeichnis über eine config.json-Datei mit dem folgenden Inhalt verfügen:

```
{
     "TargetValue": 75.0,
     "PredefinedMetricSpecification": {
         "PredefinedMetricType": "ECSServiceAverageCPUUtilization"
     },
     "ScaleOutCooldown": 60,
    "ScaleInCooldown": 60
}
```

Ausgabe:

```
{
```

```
    "PolicyARN": "arn:aws:autoscaling:us-
west-2:012345678910:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/
ecs/service/default/web-app:policyName/cpu75-target-tracking-scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/default/web-app-AlarmHigh-
d4f0770c-b46e-434a-a60f-3b36d653feca",
            "AlarmName": "TargetTracking-service/default/web-app-AlarmHigh-
d4f0770c-b46e-434a-a60f-3b36d653feca"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/default/web-app-
AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d",
            "AlarmName": "TargetTracking-service/default/web-app-
AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d"
        }
    ]
}
```

Beispiel 2: So wenden Sie eine Skalierungsrichtlinie für die Ziel-Nachverfolgung mit einer benutzerdefinierten Metrikspezifikation an

Das folgende Beispiel für `put-scaling-policy` wendet eine Skalierungsrichtlinie für die Ziel-Nachverfolgung mit einer benutzerdefinierten Metrikspezifikation auf einen Amazon-ECS-Service mit dem Namen Web-App im Standard-Cluster an. Die Richtlinie hält die durchschnittliche Auslastung des Service bei 75 Prozent, mit Aufskalierungs- und Abskalierungs-Ruhephasen von 60 Sekunden. Die Ausgabe enthält die Namen ARNs und der beiden CloudWatch Alarme, die in Ihrem Namen erstellt wurden.

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \
--scalable-dimension ecs:service:DesiredCount \
--resource-id service/default/web-app \
--policy-name cms75-target-tracking-scaling-policy
--policy-type TargetTrackingScaling \
--target-tracking-scaling-policy-configuration file://config.json
```

In diesem Beispiel wird davon ausgegangen, dass Sie im aktuellen Verzeichnis über eine config.json-Datei mit dem folgenden Inhalt verfügen:

```
{
```

```
    "TargetValue":75.0,
    "CustomizedMetricSpecification":{
        "MetricName":"MyUtilizationMetric",
        "Namespace":"MyNamespace",
        "Dimensions": [
            {
                "Name":"MyOptionalMetricDimensionName",
                "Value":"MyOptionalMetricDimensionValue"
            }
        ],
        "Statistic":"Average",
        "Unit":"Percent"
    },
    "ScaleOutCooldown": 60,
    "ScaleInCooldown": 60
}
```

Ausgabe:

```
{
    "PolicyARN": "arn:aws:autoscaling:us-west-2:012345678910:scalingPolicy:
 8784a896-b2ba-47a1-b08c-27301cc499a1:resource/ecs/service/default/web-
app:policyName/cms75-target-tracking-scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/default/web-app-
AlarmHigh-9bc77b56-0571-4276-ba0f-d4178882e0a0",
            "AlarmName": "TargetTracking-service/default/web-app-
AlarmHigh-9bc77b56-0571-4276-ba0f-d4178882e0a0"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/default/web-app-
AlarmLow-9b6ad934-6d37-438e-9e05-02836ddcbdc4",
            "AlarmName": "TargetTracking-service/default/web-app-
AlarmLow-9b6ad934-6d37-438e-9e05-02836ddcbdc4"
        }
    ]
}
```

Beispiel 3: So wenden Sie eine Skalierungsrichtlinie für die Ziel-Nachverfolgung nur für die horizontale Skalierung nach oben an

Das folgende Beispiel für `put-scaling-policy` wendet eine Skalierungsrichtlinie für die Ziel-Nachverfolgung auf einen Amazon-ECS-Service mit dem Namen `web-app` im Standard-Cluster an. Die Richtlinie wird verwendet, um den ECS-Service aufzuskalieren, wenn die `RequestCountPerTarget`-Metrik aus dem Application Load Balancer den Schwellenwert überschreitet. Die Ausgabe enthält den ARN und den Namen des CloudWatch Alarms, der in Ihrem Namen erstellt wurde.

```
aws application-autoscaling put-scaling-policy \
    --service-namespace ecs \
    --scalable-dimension ecs:service:DesiredCount \
    --resource-id service/default/web-app \
    --policy-name alb-scale-out-target-tracking-scaling-policy \
    --policy-type TargetTrackingScaling \
    --target-tracking-scaling-policy-configuration file://config.json
```

Inhalt von `config.json`:

```
{
     "TargetValue": 1000.0,
     "PredefinedMetricSpecification": {
         "PredefinedMetricType": "ALBRequestCountPerTarget",
         "ResourceLabel": "app/EC2Co-EcsEl-1TKLTMITMM0EO/f37c06a68c1748aa/
targetgroup/EC2Co-Defau-LDNM7Q3ZH1ZN/6d4ea56ca2d6a18d"
     },
     "ScaleOutCooldown": 60,
    "ScaleInCooldown": 60,
    "DisableScaleIn": true
}
```

Ausgabe:

```
{
    "PolicyARN": "arn:aws:autoscaling:us-
west-2:123456789012:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/
ecs/service/default/web-app:policyName/alb-scale-out-target-tracking-scaling-
policy",
    "Alarms": [
        {
            "AlarmName": "TargetTracking-service/default/web-app-AlarmHigh-
d4f0770c-b46e-434a-a60f-3b36d653feca",
```

```
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-service/default/web-app-AlarmHigh-
d4f0770c-b46e-434a-a60f-3b36d653feca"
        }
    ]
}
```

Weitere Informationen finden Sie in den Skalierungsrichtlinien für die Ziel-Nachverfolgung für Application Auto Scaling im AWS Benutzerhandbuch für Application Auto Scaling.

- Einzelheiten zur API finden Sie PutScalingPolicyin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell V4

Beispiel 1: Dieses Cmdlet erstellt oder aktualisiert eine Richtlinie für ein skalierbares Ziel von Application Auto Scaling. Jedes skalierbare Ziel wird durch einen Service-Namespace, eine Ressourcen-ID und eine skalierbare Dimension identifiziert.

```
Set-AASScalingPolicy -ServiceNamespace AppStream -PolicyName ASFleetScaleInPolicy
 -PolicyType StepScaling  -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity -StepScalingPolicyConfiguration_AdjustmentType
 ChangeInCapacity -StepScalingPolicyConfiguration_Cooldown 360
 -StepScalingPolicyConfiguration_MetricAggregationType Average -
StepScalingPolicyConfiguration_StepAdjustments @{ScalingAdjustment = -1;
 MetricIntervalUpperBound = 0}
```

Ausgabe:

```
Alarms    PolicyARN
------    ---------
{}        arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:4897ca24-3caa-4bf1-8484-851a089b243c:resource/
appstream/fleet/MyFleet:policyName/ASFleetScaleInPolicy
```

- Einzelheiten zur API finden Sie unter PutScalingPolicy AWS -Tools für PowerShellCmdlet-Referenz (V4).

Tools für V5 PowerShell

Beispiel 1: Dieses Cmdlet erstellt oder aktualisiert eine Richtlinie für ein skalierbares Ziel von Application Auto Scaling. Jedes skalierbare Ziel wird durch einen Service-Namespace, eine Ressourcen-ID und eine skalierbare Dimension identifiziert.

```
Set-AASScalingPolicy -ServiceNamespace AppStream -PolicyName ASFleetScaleInPolicy
 -PolicyType StepScaling  -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity -StepScalingPolicyConfiguration_AdjustmentType
 ChangeInCapacity -StepScalingPolicyConfiguration_Cooldown 360
 -StepScalingPolicyConfiguration_MetricAggregationType Average -
StepScalingPolicyConfiguration_StepAdjustments @{ScalingAdjustment = -1;
 MetricIntervalUpperBound = 0}
```

Ausgabe:

```
Alarms    PolicyARN
------    ---------
{}        arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:4897ca24-3caa-4bf1-8484-851a089b243c:resource/
appstream/fleet/MyFleet:policyName/ASFleetScaleInPolicy
```

- Einzelheiten zur API finden Sie unter PutScalingPolicy AWS -Tools für PowerShellCmdlet-Referenz (V5).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Dienst mit einem AWS SDK Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung von **`PutScheduledAction`** mit einer CLI

Die folgenden Code-Beispiele zeigen, wie `PutScheduledAction` verwendet wird.

CLI

AWS CLI

So fügen Sie eine geplante Aktion zu einer DynamoDB-Tabelle hinzu

In diesem Beispiel wird eine geplante Aktion zu einer DynamoDB-Tabelle hinzugefügt, die aufgerufen wird TestTable , um nach einem wiederkehrenden Zeitplan zu skalieren. Gemäß

dem angegebenen Zeitplan (täglich um 12:15 Uhr UTC) wird Application Auto Scaling auf den von angegebenen Wert skaliert MinCapacity, wenn die aktuelle Kapazität unter dem für angegebenen Wert liegt. MinCapacity

Befehl:

```
aws application-autoscaling put-scheduled-action --service-
namespace dynamodb --scheduled-action-name my-recurring-action --
schedule "cron(15 12 * * ? *)" --resource-id table/TestTable --
scalable-dimension dynamodb:table:WriteCapacityUnits --scalable-target-
action MinCapacity=6
```

Weitere Informationen finden Sie unter „Geplante Skalierung“ im Benutzerhandbuch für Application Auto Scaling.

- Einzelheiten zur API finden Sie PutScheduledActionin der AWS CLI Befehlsreferenz.

PowerShell

Tools für PowerShell V4

Beispiel 1: Dieses Cmdlet erstellt oder aktualisiert eine geplante Aktion für ein skalierbares Ziel von Application Auto Scaling. Jedes skalierbare Ziel wird durch einen Service-Namespace, eine Ressourcen-ID und eine skalierbare Dimension identifiziert.

```
Set-AASScheduledAction -ServiceNamespace AppStream -ResourceId fleet/
MyFleet -Schedule "cron(0 0 8 ? * MON-FRI *)" -ScalableDimension
 appstream:fleet:DesiredCapacity -ScheduledActionName WeekDaysFleetScaling -
ScalableTargetAction_MinCapacity 5 -ScalableTargetAction_MaxCapacity 10
```

- Einzelheiten zur API finden Sie unter PutScheduledAction AWS -Tools für PowerShellCmdlet-Referenz (V4).

Tools für V5 PowerShell

Beispiel 1: Dieses Cmdlet erstellt oder aktualisiert eine geplante Aktion für ein skalierbares Ziel von Application Auto Scaling. Jedes skalierbare Ziel wird durch einen Service-Namespace, eine Ressourcen-ID und eine skalierbare Dimension identifiziert.

```
Set-AASScheduledAction -ServiceNamespace AppStream -ResourceId fleet/
MyFleet -Schedule "cron(0 0 8 ? * MON-FRI *)" -ScalableDimension
```

```
 appstream:fleet:DesiredCapacity -ScheduledActionName WeekDaysFleetScaling -
ScalableTargetAction_MinCapacity 5 -ScalableTargetAction_MaxCapacity 10
```

- Einzelheiten zur API finden Sie unter PutScheduledAction AWS -Tools für PowerShellCmdlet-Referenz (V5).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Dienst mit einem AWS SDK Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`RegisterScalableTarget`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `RegisterScalableTarget` verwendet wird.

CLI

AWS CLI

Beispiel 1: So registrieren Sie einen ECS-Service als skalierbares Ziel

Im folgenden Beispiel für `register-scalable-target` wird ein Amazon-ECS-Service bei Application Auto Scaling registriert. Außerdem wird dem skalierbaren Ziel ein Tag mit dem Schlüsselnamen `environment` und dem Wert `production` hinzugefügt.

```
aws application-autoscaling register-scalable-target \
    --service-namespace ecs \
    --scalable-dimension ecs:service:DesiredCount \
    --resource-id service/default/web-app \
    --min-capacity 1 --max-capacity 10 \
    --tags environment=production
```

Ausgabe:

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:us-
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Beispiele für andere AWS Dienste und benutzerdefinierte Ressourcen finden Sie in den Themen unter AWS Dienste, die Sie mit Application Auto Scaling verwenden können im Application Auto Scaling Scaling-Benutzerhandbuch.

Beispiel 2: So setzen Sie die Skalierungsaktivitäten für ein skalierbares Ziel aus

Im folgenden Beispiel für `register-scalable-target` werden die Skalierungsaktivitäten für ein vorhandenes skalierbares Ziel ausgesetzt.

```
aws application-autoscaling register-scalable-target \
    --service-namespace dynamodb \
    --scalable-dimension dynamodb:table:ReadCapacityUnits \
    --resource-id table/my-table \
    --suspended-
state DynamicScalingInSuspended=true,DynamicScalingOutSuspended=true,ScheduledScalingSusp
```

Ausgabe:

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:us-
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Weitere Informationen finden Sie unter Unterbrechung und Wiederaufnahme der Skalierung von Application Auto Scaling im Benutzerhandbuch für Application Auto Scaling.

Beispiel 3: So nehmen Sie die Skalierungsaktivitäten für ein skalierbares Ziel wieder auf

Mit dem folgenden Beispiel für `register-scalable-target` werden alle Skalierungsaktivitäten für ein vorhandenes skalierbares Ziel fortgesetzt.

```
aws application-autoscaling register-scalable-target \
    --service-namespace dynamodb \
    --scalable-dimension dynamodb:table:ReadCapacityUnits \
    --resource-id table/my-table \
    --suspended-
state DynamicScalingInSuspended=false,DynamicScalingOutSuspended=false,ScheduledScalingSu
```

Ausgabe:

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:us-
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Weitere Informationen finden Sie unter Unterbrechung und Wiederaufnahme der Skalierung von Application Auto Scaling im Benutzerhandbuch für Application Auto Scaling.

- Einzelheiten zur API finden Sie RegisterScalableTargetin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.applicationautoscaling.ApplicationAutoScalingClient;
import
 software.amazon.awssdk.services.applicationautoscaling.model.ApplicationAutoScalingExcep
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalableTargetsRequ
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalableTargetsResp
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalingPoliciesRequ
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalingPoliciesResp
import software.amazon.awssdk.services.applicationautoscaling.model.PolicyType;
import
 software.amazon.awssdk.services.applicationautoscaling.model.PredefinedMetricSpecificati
import
 software.amazon.awssdk.services.applicationautoscaling.model.PutScalingPolicyRequest;
import
 software.amazon.awssdk.services.applicationautoscaling.model.RegisterScalableTargetReque
import
 software.amazon.awssdk.services.applicationautoscaling.model.ScalingPolicy;
import
 software.amazon.awssdk.services.applicationautoscaling.model.ServiceNamespace;
import
 software.amazon.awssdk.services.applicationautoscaling.model.ScalableDimension;
import software.amazon.awssdk.services.applicationautoscaling.model.MetricType;
```

```
import
 software.amazon.awssdk.services.applicationautoscaling.model.TargetTrackingScalingPolicy
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development environment,
 including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class EnableDynamoDBAutoscaling {
    public static void main(String[] args) {
        final String usage = """

            Usage:
               <tableId> <roleARN> <policyName>\s

            Where:
               tableId - The table Id value (for example, table/Music).
               roleARN - The ARN of the role that has ApplicationAutoScaling
 permissions.
               policyName - The name of the policy to create.

            """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        System.out.println("This example registers an Amazon DynamoDB table,
 which is the resource to scale.");
        String tableId = args[0];
        String roleARN = args[1];
        String policyName = args[2];
        ServiceNamespace ns = ServiceNamespace.DYNAMODB;
        ScalableDimension tableWCUs =
 ScalableDimension.DYNAMODB_TABLE_WRITE_CAPACITY_UNITS;
        ApplicationAutoScalingClient appAutoScalingClient =
 ApplicationAutoScalingClient.builder()
            .region(Region.US_EAST_1)
```

```
            .build();

       registerScalableTarget(appAutoScalingClient, tableId, roleARN, ns,
tableWCUs);
       verifyTarget(appAutoScalingClient, tableId, ns, tableWCUs);
       configureScalingPolicy(appAutoScalingClient, tableId, ns, tableWCUs,
policyName);
   }

   public static void registerScalableTarget(ApplicationAutoScalingClient
appAutoScalingClient, String tableId, String roleARN, ServiceNamespace ns,
ScalableDimension tableWCUs) {
       try {
           RegisterScalableTargetRequest targetRequest =
RegisterScalableTargetRequest.builder()
                .serviceNamespace(ns)
                .scalableDimension(tableWCUs)
                .resourceId(tableId)
                .roleARN(roleARN)
                .minCapacity(5)
                .maxCapacity(10)
                .build();

           appAutoScalingClient.registerScalableTarget(targetRequest);
           System.out.println("You have registered " + tableId);

       } catch (ApplicationAutoScalingException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
       }
   }

   // Verify that the target was created.
   public static void verifyTarget(ApplicationAutoScalingClient
appAutoScalingClient, String tableId, ServiceNamespace ns, ScalableDimension
tableWCUs) {
       DescribeScalableTargetsRequest dscRequest =
DescribeScalableTargetsRequest.builder()
           .scalableDimension(tableWCUs)
           .serviceNamespace(ns)
           .resourceIds(tableId)
           .build();

       DescribeScalableTargetsResponse response =
appAutoScalingClient.describeScalableTargets(dscRequest);
```

```
        System.out.println("DescribeScalableTargets result: ");
        System.out.println(response);
    }

    // Configure a scaling policy.
    public static void configureScalingPolicy(ApplicationAutoScalingClient
appAutoScalingClient, String tableId, ServiceNamespace ns, ScalableDimension
tableWCUs, String policyName) {
        // Check if the policy exists before creating a new one.
        DescribeScalingPoliciesResponse describeScalingPoliciesResponse =
appAutoScalingClient.describeScalingPolicies(DescribeScalingPoliciesRequest.builder()
            .serviceNamespace(ns)
            .resourceId(tableId)
            .scalableDimension(tableWCUs)
            .build());

        if (!describeScalingPoliciesResponse.scalingPolicies().isEmpty()) {
            // If policies exist, consider updating an existing policy instead of
creating a new one.
            System.out.println("Policy already exists. Consider updating it
instead.");
            List<ScalingPolicy> polList =
describeScalingPoliciesResponse.scalingPolicies();
            for (ScalingPolicy pol : polList) {
                System.out.println("Policy name:" +pol.policyName());
            }
        } else {
            // If no policies exist, proceed with creating a new policy.
            PredefinedMetricSpecification specification =
PredefinedMetricSpecification.builder()

.predefinedMetricType(MetricType.DYNAMO_DB_WRITE_CAPACITY_UTILIZATION)
                .build();

            TargetTrackingScalingPolicyConfiguration policyConfiguration =
TargetTrackingScalingPolicyConfiguration.builder()
                .predefinedMetricSpecification(specification)
                .targetValue(50.0)
                .scaleInCooldown(60)
                .scaleOutCooldown(60)
                .build();

            PutScalingPolicyRequest putScalingPolicyRequest =
PutScalingPolicyRequest.builder()
```

```
                .targetTrackingScalingPolicyConfiguration(policyConfiguration)
                .serviceNamespace(ns)
                .scalableDimension(tableWCUs)
                .resourceId(tableId)
                .policyName(policyName)
                .policyType(PolicyType.TARGET_TRACKING_SCALING)
                .build();

            try {
                appAutoScalingClient.putScalingPolicy(putScalingPolicyRequest);
                System.out.println("You have successfully created a scaling
 policy for an Application Auto Scaling scalable target");
            } catch (ApplicationAutoScalingException e) {
                System.err.println("Error: " +
 e.awsErrorDetails().errorMessage());
            }
        }
    }
}
```

- Einzelheiten zur API finden Sie RegisterScalableTargetin der AWS SDK for Java 2.x API-Referenz.

PowerShell

Tools für PowerShell V4

Beispiel 1: Dieses Cmdlet registriert oder aktualisiert ein skalierbares Ziel. Ein skalierbares Ziel ist eine Ressource, die dank Application Auto Scaling auf- bzw. abskaliert werden kann.

```
Add-AASScalableTarget -ServiceNamespace AppStream -ResourceId fleet/MyFleet -
ScalableDimension appstream:fleet:DesiredCapacity -MinCapacity 2 -MaxCapacity 10
```

- Einzelheiten zur API finden Sie unter RegisterScalableTarget AWS -Tools für PowerShellCmdlet-Referenz (V4).

Tools für V5 PowerShell

Beispiel 1: Dieses Cmdlet registriert oder aktualisiert ein skalierbares Ziel. Ein skalierbares Ziel ist eine Ressource, die dank Application Auto Scaling auf- bzw. abskaliert werden kann.

```
Add-AASScalableTarget -ServiceNamespace AppStream -ResourceId fleet/MyFleet -
ScalableDimension appstream:fleet:DesiredCapacity -MinCapacity 2 -MaxCapacity 10
```

- Einzelheiten zur API finden Sie unter RegisterScalableTarget AWS -Tools für PowerShellCmdlet-Referenz (V5).

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unter. Verwenden Sie diesen Dienst mit einem AWS SDK Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Tagging-Unterstützung für Auto Scaling von Anwendungen

Sie können das AWS CLI oder ein SDK verwenden, um skalierbare Ziele von Application Auto Scaling zu kennzeichnen. Skalierbare Ziele sind die Entitäten, die die AWS oder benutzerdefinierten Ressourcen darstellen, die Application Auto Scaling skalieren kann.

Jedes Tag ist ein Label, das aus einem benutzerdefinierten Schlüssel und einem Wert besteht, der über die API von Application Auto Scaling definiert wird. Mithilfe von Tags können Sie den Zugriff auf bestimmte skalierbare Ziele entsprechend den Anforderungen Ihres Unternehmens granular konfigurieren. Weitere Informationen finden Sie unter ABAC mit Application Auto Scaling.

Sie können Tags zu neuen skalierbaren Zielen hinzufügen, wenn Sie diese registrieren, oder Sie können sie zu vorhandenen skalierbaren Zielen hinzufügen.

Zu den häufig verwendeten Befehlen für die Verwaltung von Tags gehören:

- register-scalable-targetum neue skalierbare Ziele zu kennzeichnen, wenn Sie sie registrieren.
- tag-resource zum Hinzufügen von Tags zu einem vorhandenen skalierbaren Ziel.
- list-tags-for-resourceum die Tags für ein skalierbares Ziel zurückzugeben.
- untag-resource um ein Tag zu löschen.

Beispiel für eine Markierung

Verwenden Sie den folgenden register-scalable-targetBefehl mit der `--tags` Option. In diesem Beispiel wird ein skalierbares Ziel mit zwei Tags markiert: einem Tag-Schlüssel mit dem Namen **`environment`** mit dem Tag-Wert von **`production`** und einem Tag-Schlüssel mit dem Namen **`iscontainerbased`** mit dem Tag-Wert von **`true`**.

Ersetzen Sie die Beispielwerte für `--min-capacity` und `--max-capacity` und den Beispieltext für `--service-namespace` durch den Namespace des AWS Dienstes, den Sie mit Application Auto Scaling verwenden, `--scalable-dimension` durch die skalierbare Dimension, die der Ressource zugeordnet ist, die Sie registrieren, und `--resource-id` durch einen Bezeichner für die Ressource. Weitere Informationen und Beispiele für die einzelnen Services finden Sie in den Themen unter AWS-Services die Sie mit Application Auto Scaling verwenden können.

```
aws application-autoscaling register-scalable-target \
```

```
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --min-capacity 1 --max-capacity 10 \
  --tags environment=production,iscontainerbased=true
```

Bei Erfolg gibt dieser Befehl den ARN des skalierbaren Ziels zurück.

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

> **Note**
>
> Wenn dieser Befehl einen Fehler auslöst, stellen Sie sicher, dass Sie den Befehl AWS CLI lokal auf die neueste Version aktualisiert haben.

Tags für Sicherheit

Verwenden Sie Tags, um zu überprüfen, ob der Anforderer (z. B. ein IAM-Benutzer oder eine Rolle) die Berechtigung hat, bestimmte Aktionen durchzuführen. Geben Sie Tag-Informationen im Bedingungselement einer IAM-Richtlinie mithilfe eines oder mehrerer der folgenden Bedingungsschlüssel an:

- Verwenden Sie `aws:ResourceTag/tag-key: tag-value`, um Benutzeraktionen für skalierbare Ziele mit bestimmten Tags zuzulassen (oder zu verweigern).
- Schreiben Sie mit `aws:RequestTag/tag-key: tag-value` vor, dass in einer Anforderung ein bestimmtes Tag vorhanden (oder nicht vorhanden) sein muss.
- Schreiben Sie mit `aws:TagKeys [tag-key, ...]` vor, dass in einer Anforderung bestimmte Tag-Schlüssel vorhanden (oder nicht vorhanden) sein müssen.

Die folgende IAM-Richtlinie erteilt beispielsweise dem Benutzer Berechtigungen für die folgenden Aktionen: `DeregisterScalableTarget`, `DeleteScalingPolicy` und `DeleteScheduledAction`. Es lehnt die Aktionen jedoch auch dann ab, wenn das skalierbare Ziel, auf die die Aktion abzielt, über das Tag **`environment=production`** verfügt.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:DeregisterScalableTarget",
                "application-autoscaling:DeleteScalingPolicy",
                "application-autoscaling:DeleteScheduledAction"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "application-autoscaling:DeregisterScalableTarget",
                "application-autoscaling:DeleteScalingPolicy",
                "application-autoscaling:DeleteScheduledAction"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/environment": "production"
                }
            }
        }
    ]
}
```

Steuern des Zugriffs auf Tags

Verwenden Sie Tags, um zu überprüfen, ob der Anforderer (z. B. ein IAM-Benutzer oder eine IAM-Rolle) über Berechtigungen zum Hinzufügen, Ändern oder Löschen von Tags für skalierbare Ziele verfügt.

Sie könnten beispielsweise eine IAM-Richtlinie erstellen, die nur das Entfernen des Tags mithilfe des **`temporary`**-Schlüssels aus skalierbaren Zielen erlaubt.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "application-autoscaling:UntagResource",
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": { "aws:TagKeys": ["temporary"] }
            }
        }
    ]
}
```

Sicherheit bei Application Auto Scaling

Cloud-Sicherheit AWS hat höchste Priorität. Als AWS Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame Verantwortung von Ihnen AWS und Ihnen. Das Modell der geteilten Verantwortung beschreibt dies als Sicherheit der Cloud und Sicherheit in der Cloud:

- Sicherheit der Cloud — AWS ist verantwortlich für den Schutz der Infrastruktur, die AWS Dienste in der AWS Cloud ausführt. AWS bietet Ihnen auch Dienste, die Sie sicher nutzen können. Externe Prüfer testen und verifizieren regelmäßig die Wirksamkeit unserer Sicherheitsmaßnahmen im Rahmen der AWS und . Weitere Informationen zu den Compliance-Programmen, die für Application Auto Scaling gelten, finden Sie unter AWS Services im Umfang nach Compliance-Programm AWS .
- Sicherheit in der Cloud — Ihre Verantwortung richtet sich nach dem AWS Dienst, den Sie nutzen. Sie sind auch für andere Faktoren verantwortlich, einschließlich der Vertraulichkeit Ihrer Daten, für die Anforderungen Ihres Unternehmens und für die geltenden Gesetze und Vorschriften.

Diese Dokumentation hilft Ihnen zu verstehen, wie Sie das Modell der geteilten Verantwortung bei der Verwendung von Application Auto Scaling anwenden. In den folgenden Themen erfahren Sie, wie Sie Application Auto Scaling so konfigurieren, dass Ihre Sicherheits- und Compliance-Ziele erreicht werden. Sie lernen auch, wie Sie andere AWS Dienste verwenden können, die Sie bei der Überwachung und Sicherung Ihrer Application Auto Scaling Scaling-Ressourcen unterstützen.

Inhalt

Datenschutz bei Application Auto Scaling

Das AWS Modell der gilt für den Datenschutz in Application Auto Scaling. Wie in diesem Modell beschrieben, AWS ist es für den Schutz der globalen Infrastruktur verantwortlich, auf der alle Systeme laufen AWS Cloud. Sie sind dafür verantwortlich, die Kontrolle über Ihre in dieser Infrastruktur gehosteten Inhalte zu behalten. Sie sind auch für die Sicherheitskonfiguration und die Verwaltungsaufgaben für die von Ihnen verwendeten AWS-Services verantwortlich. Weitere Informationen zum Datenschutz finden Sie unter Häufig gestellte Fragen zum Datenschutz. Informationen zum Datenschutz in Europa finden Sie im Blog-Beitrag AWS -Modell der geteilten Verantwortung und in der DSGVO im AWS -Sicherheitsblog.

Aus Datenschutzgründen empfehlen wir, dass Sie AWS-Konto Anmeldeinformationen schützen und einzelne Benutzer mit AWS IAM Identity Center oder AWS Identity and Access Management (IAM) einrichten. So erhält jeder Benutzer nur die Berechtigungen, die zum Durchführen seiner Aufgaben erforderlich sind. Außerdem empfehlen wir, die Daten mit folgenden Methoden schützen:

- Verwenden Sie für jedes Konto die Multi-Faktor-Authentifizierung (MFA).
- Wird verwendet SSL/TLS , um mit AWS Ressourcen zu kommunizieren. Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Richten Sie die API und die Protokollierung von Benutzeraktivitäten mit ein AWS CloudTrail. Informationen zur Verwendung von CloudTrail Pfaden zur Erfassung von AWS Aktivitäten finden Sie unter Arbeiten mit CloudTrail Pfaden im AWS CloudTrail Benutzerhandbuch.
- Verwenden Sie AWS Verschlüsselungslösungen zusammen mit allen darin enthaltenen Standardsicherheitskontrollen AWS-Services.
- Verwenden Sie erweiterte verwaltete Sicherheitsservices wie Amazon Macie, die dabei helfen, in Amazon S3 gespeicherte persönliche Daten zu erkennen und zu schützen.
- Wenn Sie für den Zugriff AWS über eine Befehlszeilenschnittstelle oder eine API FIPS 140-3-validierte kryptografische Module benötigen, verwenden Sie einen FIPS-Endpunkt. Weitere Informationen über verfügbare FIPS-Endpunkte finden Sie unter Federal Information Processing Standard (FIPS) 140-3.

Wir empfehlen dringend, in Freitextfeldern, z. B. im Feld Name, keine vertraulichen oder sensiblen Informationen wie die E-Mail-Adressen Ihrer Kunden einzugeben. Dies gilt auch, wenn Sie mit Application Auto Scaling oder anderen Anwendungen arbeiten und die Konsole AWS CLI, die API oder AWS-Services verwenden AWS SDKs. Alle Daten, die Sie in Tags oder Freitextfelder eingeben, die für Namen verwendet werden, können für Abrechnungs- oder Diagnoseprotokolle verwendet

werden. Wenn Sie eine URL für einen externen Server bereitstellen, empfehlen wir dringend, keine Anmeldeinformationen zur Validierung Ihrer Anforderung an den betreffenden Server in die URL einzuschließen.

Identity and Access Management für Application Auto Scaling

AWS Identity and Access Management (IAM) hilft einem Administrator AWS-Service , den Zugriff auf AWS Ressourcen sicher zu kontrollieren. IAM-Administratoren steuern, wer authentifiziert (angemeldet) und autorisiert (im Besitz von Berechtigungen) ist, Application Auto Scaling-Ressourcen zu nutzen. IAM ist ein Programm AWS-Service , das Sie ohne zusätzliche Kosten nutzen können.

Eine umfassende IAM-Dokumentation finden Sie im IAM User Guide.

Zugriffskontrolle

Sie können über gültige Anmeldedaten verfügen, um Ihre Anfragen zu authentifizieren, aber ohne die entsprechenden Berechtigungen können Sie keine Ressourcen für Application Auto Scaling erstellen oder darauf zugreifen. Beispielsweise müssen Sie über Berechtigungen zum Erstellen von Skalierungsrichtlinien, zum Konfigurieren der geplanten Skalierung usw. verfügen.

In den folgenden Abschnitten erfahren Sie, wie ein IAM-Administrator IAM verwenden kann, um Ihre AWS Ressourcen zu schützen, indem er steuert, wer Application Auto Scaling Scaling-API-Aktionen ausführen kann.

Inhalt

Wie Application Auto Scaling mit IAM funktioniert

> **Note**
>
> Im Dezember 2017 gab es ein Update für Application Auto Scaling, das mehrere dienstverknüpfte Rollen für integrierte Dienste von Application Auto Scaling ermöglichte. Damit Benutzer die Skalierung konfigurieren können, sind spezielle IAM-Berechtigungen und eine mit dem Service Application Auto Scaling verknüpfte Rolle (oder eine Service-Rolle für Amazon EMR Auto Scaling) erforderlich.

Bevor Sie IAM verwenden, um den Zugriff auf Application Auto Scaling zu verwalten, lernen Sie, welche IAM-Funktionen für die Verwendung mit Application Auto Scaling verfügbar sind.

IAM-Features, die mit Application Auto Scaling verwendet werden können

IAM-Feature	Unterstützung für Auto Scaling von Anwendungen
Identitätsbasierte Richtlinien	Ja
Richtlinienaktionen	Ja
Richtlinienressourcen	Ja
Richtlinienbedingungsschlüssel (servicespezifisch)	Ja
Ressourcenbasierte Richtlinien	Nein
ACLs	Nein
ABAC (Tags in Richtlinien)	Teilweise
Temporäre Anmeldeinformationen	Ja
Servicerollen	Ja
Service-verknüpfte Rollen	Ja

Einen allgemeinen Überblick darüber, wie Application Auto Scaling und andere Funktionen mit den meisten IAM-Funktionen AWS-Services funktionieren AWS-Services , finden Sie im IAM-Benutzerhandbuch unter Diese Funktionen mit IAM.

Application Auto Scaling identitätsbasierte Richtlinien

Unterstützt Richtlinien auf Identitätsbasis: Ja

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Definieren benutzerdefinierter IAM-Berechtigungen mit vom Kunden verwalteten Richtlinien im IAM-Benutzerhandbuch.

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen zugelassen oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter denen Aktionen zugelassen oder abgelehnt werden. Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der IAM-Referenz für JSON-Richtlinienelemente im IAM-Benutzerhandbuch.

Beispiele für identitätsbasierte Richtlinien für Application Auto Scaling

Beispiele für identitätsbasierte Richtlinien von Application Auto Scaling finden Sie unter Beispiele für identitätsbasierte Richtlinien für Application Auto Scaling.

Aktionen

Unterstützt Richtlinienaktionen: Ja

In einer IAM-Richtlinienanweisung können Sie jede API-Aktion von jedem Service, der IAM unterstützt, angeben. Bei Application Auto Scaling setzen Sie folgendes Präfix vor den Namen der API-Aktion: `application-autoscaling:`. Beispiel: `application-autoscaling:RegisterScalableTarget`, `application-autoscaling:PutScalingPolicy` und `application-autoscaling:DeregisterScalableTarget`.

Um mehrere Aktionen in einer einzelnen Anweisung anzugeben, trennen Sie sie durch Beistriche, wie im folgenden Beispiel gezeigt.

```
"Action": [
```

```
        "application-autoscaling:DescribeScalingPolicies",
        "application-autoscaling:DescribeScalingActivities"
```

Sie können auch Platzhalter (*) verwenden, um mehrere Aktionen anzugeben. Beispielsweise können Sie alle Aktionen festlegen, die mit dem Wort `Describe` beginnen, einschließlich der folgenden Aktion:

```
"Action": "application-autoscaling:Describe*"
```

Eine Liste der Application Auto Scaling-Aktionen finden Sie unter Von AWS Application Auto Scaling definierte Aktionen in der Service Authorization Reference.

Ressourcen

Unterstützt Richtlinienressourcen: Ja

In einer IAM-Richtlinienanweisung gibt das `Resource`-Element das Objekt oder die Objekte an, für die die Anweisung gilt. Bei Application Auto Scaling gilt jede IAM-Richtlinienanweisung für die skalierbaren Ziele, die Sie mit ihren Amazon-Ressourcennamen (ARNs) angeben.

Das ARN-Ressourcenformat für skalierbare Ziele:

```
arn:aws:application-autoscaling:region:account-id:scalable-target/unique-identifier
```

Sie können zum Beispiel ein bestimmtes skalierbares Ziel in Ihrer Anweisung mit seinem ARN wie folgt angeben. Die eindeutige ID (1234abcd56ab78cd901ef1234567890ab123) ist ein Wert, der dem skalierbaren Ziel von Application Auto Scaling zugewiesen wird.

```
"Resource": "arn:aws:application-autoscaling:us-east-1:123456789012:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
```

Sie können alle Instances angeben, die zu einem bestimmten Konto gehören, indem Sie den eindeutigen Bezeichner wie folgt durch einen Platzhalter (*) ersetzen.

```
"Resource": "arn:aws:application-autoscaling:us-east-1:123456789012:scalable-target/*"
```

Um alle Ressourcen anzugeben oder falls eine bestimmte API-Aktion dies nicht unterstützt ARNs, verwenden Sie einen Platzhalter (*) als `Resource` Element wie folgt.

```
"Resource": "*"
```

Weitere Informationen finden Sie unter Von AWS Application Auto Scaling definierte Ressourcentypen in der Service Authorization Reference.

Bedingungsschlüssel

Unterstützt servicespezifische Richtlinienbedingungsschlüssel: Ja

Sie können in den IAM-Richtlinien Bedingungen festlegen, die den Zugriff auf Application Auto Scaling-Ressourcen steuern. Die Richtlinienanweisung ist nur wirksam, wenn diese Bedingungen erfüllt sind.

Application Auto Scaling unterstützt die folgenden servicedefinierten Bedingungsschlüssel, die Sie in identitätsbasierten Richtlinien verwenden können, um zu bestimmen, wer Application Auto Scaling-API-Aktionen durchführen darf.

- `application-autoscaling:scalable-dimension`
- `application-autoscaling:service-namespace`

Informationen zu den API-Aktionen von Application Auto Scaling, mit denen Sie einen Bedingungsschlüssel verwenden können, finden Sie unter Von AWS Application Auto Scaling definierte Aktionen in der Service Authorization Reference. Weitere Informationen zur Verwendung von Application Auto Scaling-Bedingungsschlüsseln finden Sie unter Bedingungsschlüssel für AWS Application Auto Scaling.

Informationen zu den globalen Bedingungsschlüsseln, die für alle Dienste verfügbar sind, finden Sie unter globalen Bedingungskontextschlüsseln für AWS im IAM-Benutzerhandbuch.

Ressourcenbasierte Richtlinien

Unterstützt ressourcenbasierte Richtlinien: Nein

Andere AWS Dienste, wie Amazon Simple Storage Service, unterstützen ressourcenbasierte Berechtigungsrichtlinien. Beispielsweise können Sie einem S3-Bucket eine Berechtigungsrichtlinie zuweisen, um die Zugriffsberechtigungen für diesen Bucket zu verwalten.

Application Auto Scaling unterstützt keine ressourcenbasierten Richtlinien.

Zugriffskontrolllisten () ACLs

Unterstützt ACLs: Nein

Application Auto Scaling unterstützt keine Zugriffskontrolllisten (ACLs).

ABAC mit Application Auto Scaling

Unterstützt ABAC (Tags in Richtlinien): Teilweise

Die attributbasierte Zugriffskontrolle (ABAC) ist eine Autorisierungsstrategie, bei der Berechtigungen basierend auf Attributen definiert werden. In AWS werden diese Attribute als Tags bezeichnet. Sie können Tags an IAM-Entitäten (Benutzer oder Rollen) und an viele AWS Ressourcen anhängen. Das Markieren von Entitäten und Ressourcen ist der erste Schritt von ABAC. Anschließend entwerfen Sie ABAC-Richtlinien, um Operationen zuzulassen, wenn das Tag des Prinzipals mit dem Tag der Ressource übereinstimmt, auf die sie zugreifen möchten.

ABAC ist in Umgebungen hilfreich, die schnell wachsen, und unterstützt Sie in Situationen, in denen die Richtlinienverwaltung mühsam wird.

Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an, indem Sie die Schlüssel `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, oder Bedingung `aws:TagKeys` verwenden.

ABAC ist für Ressourcen möglich, die Tags unterstützen. Tags werden jedoch nicht von allen Ressourcen unterstützt. Geplante Aktionen und Skalierungsrichtlinien unterstützen keine Tags, aber skalierbare Ziele unterstützen Tags. Weitere Informationen finden Sie unter Tagging-Unterstützung für Auto Scaling von Anwendungen.

Weitere Informationen zu ABAC finden Sie unter Was ist ABAC? im IAM-Benutzerhandbuch. Um ein Tutorial mit Schritten zur Einstellung von ABAC anzuzeigen, siehe Attributbasierte Zugriffskontrolle (ABAC) verwenden im IAM-Benutzerhandbuch.

Verwendung temporärer Anmeldeinformationen mit Application Auto Scaling

Unterstützt temporäre Anmeldeinformationen: Ja

Temporäre Anmeldeinformationen ermöglichen kurzfristigen Zugriff auf AWS Ressourcen und werden automatisch erstellt, wenn Sie einen Verbund verwenden oder die Rollen wechseln. AWS empfiehlt, temporäre Anmeldeinformationen dynamisch zu generieren, anstatt langfristige Zugriffsschlüssel zu

verwenden. Weitere Informationen finden Sie unter Temporäre Anmeldeinformationen in IAM und AWS-Services , die mit IAM funktionieren im IAM-Benutzerhandbuch.

Servicerollen

Unterstützt Servicerollen: Ja

Wenn Ihr Amazon EMR-Cluster automatische Skalierung verwendet, ermöglicht dieses Feature Application Auto Scaling, eine Service-Rolle in Ihrem Namen zu übernehmen. Ähnlich wie bei einer serviceverknüpften Rolle ermöglicht eine Servicerolle dem Service den Zugriff auf Ressourcen in anderen Services, um eine Aktion in Ihrem Namen durchzuführen. Servicerollen werden in Ihrem IAM-Konto angezeigt und gehören zum Konto. Dies bedeutet, dass ein IAM-Administrator die Berechtigungen für diese Rolle ändern kann. Dies kann jedoch die Funktionalität des Dienstes beeinträchtigen.

Application Auto Scaling unterstützt nur Service-Rollen für Amazon EMR. Die Dokumentation für die EMR-Service-Rolle finden Sie unter Verwendung der automatischen Skalierung mit einer benutzerdefinierten Richtlinie für Instance-Gruppe im Amazon EMR Management Leitfaden.

> **Note**
>
> Mit der Einführung von serviceverknüpften Rollen sind mehrere Legacy-Servicerollen nicht mehr erforderlich, beispielsweise für Amazon ECS und Spot-Flotte.

Service-verknüpfte Rollen

Unterstützt serviceverknüpfte Rollen: Ja

Eine dienstbezogene Rolle ist eine Art von Servicerolle, die mit einer AWS-Service verknüpft ist. Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Dienstbezogene Rollen werden in Ihrem Dienst angezeigt AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.

Weitere Informationen zu serviceverknüpften Rollen für Application Auto Scaling finden Sie unter Servicegebundene Rollen für Application Auto Scaling.

AWS verwaltete Richtlinien für Application Auto Scaling

Eine AWS verwaltete Richtlinie ist eine eigenständige Richtlinie, die von erstellt und verwaltet wird AWS. AWS Verwaltete Richtlinien sind so konzipiert, dass sie Berechtigungen für viele gängige

Anwendungsfälle bereitstellen, sodass Sie damit beginnen können, Benutzern, Gruppen und Rollen Berechtigungen zuzuweisen.

Beachten Sie, dass AWS verwaltete Richtlinien für Ihre speziellen Anwendungsfälle möglicherweise keine Berechtigungen mit den geringsten Rechten gewähren, da sie für alle AWS Kunden verfügbar sind. Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind.

Sie können die in AWS verwalteten Richtlinien definierten Berechtigungen nicht ändern. Wenn die in einer AWS verwalteten Richtlinie definierten Berechtigungen AWS aktualisiert werden, wirkt sich das Update auf alle Prinzidentitäten (Benutzer, Gruppen und Rollen) aus, denen die Richtlinie zugeordnet ist. AWS aktualisiert eine AWS verwaltete Richtlinie höchstwahrscheinlich, wenn eine neue Richtlinie eingeführt AWS-Service wird oder neue API-Operationen für bestehende Dienste verfügbar werden.

Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien im IAM-Benutzerhandbuch.

AWS verwaltete Richtlinie: WorkSpaces Anwendungen und CloudWatch

Name der Richtlinie: AWSApplicationAutoscalingAppStreamFleetPolicy

Diese Richtlinie ist der dienstbezogenen Rolle zugeordnet, die benannt wurde AWSServiceRoleForApplicationAutoScaling_AppStreamFleet, damit Application Auto Scaling Amazon anrufen AppStream CloudWatch und die Skalierung in Ihrem Namen durchführen kann.

Berechtigungsdetails

Die Berechtigungsrichtlinie ermöglicht es Application Auto Scaling, die folgenden Aktionen für alle zugehörigen Ressourcen durchzuführen („Ressource“: „*“):

- Aktion: `appstream:DescribeFleets`
- Aktion: `appstream:UpdateFleet`
- Aktion: `cloudwatch:DescribeAlarms`
- Aktion: `cloudwatch:PutMetricAlarm`
- Aktion: `cloudwatch:DeleteAlarms`

AWS verwaltete Richtlinie: Aurora und CloudWatch

Name der Richtlinie: AWSApplicationRDSClusterAutoscaling-Richtlinie

Diese Richtlinie ist der dienstbezogenen Rolle zugeordnet, die benannt ist AWSServiceRoleForApplicationAutoScaling_RDSCluster, damit Application Auto Scaling Aurora aufrufen CloudWatch und die Skalierung in Ihrem Namen durchführen kann.

Berechtigungsdetails

Die Berechtigungsrichtlinie ermöglicht es Application Auto Scaling, die folgenden Aktionen für alle zugehörigen Ressourcen durchzuführen („Ressource“: „*“):

- Aktion: `rds:AddTagsToResource`
- Aktion: `rds:CreateDBInstance`
- Aktion: `rds:DeleteDBInstance`
- Aktion: `rds:DescribeDBClusters`
- Aktion: `rds:DescribeDBInstance`
- Aktion: `cloudwatch:DescribeAlarms`
- Aktion: `cloudwatch:PutMetricAlarm`
- Aktion: `cloudwatch:DeleteAlarms`

AWS verwaltete Richtlinie: Amazon Comprehend und CloudWatch

Name der Richtlinie: AWSApplicationAutoscalingComprehendEndpointPolicy

Diese Richtlinie ist der dienstbezogenen Rolle zugeordnet, die benannt wurde AWSServiceRoleForApplicationAutoScaling_ComprehendEndpoint, damit Application Auto Scaling Amazon Comprehend aufrufen CloudWatch und die Skalierung in Ihrem Namen durchführen kann.

Berechtigungsdetails

Die Berechtigungsrichtlinie ermöglicht es Application Auto Scaling, die folgenden Aktionen für alle zugehörigen Ressourcen durchzuführen („Ressource“: „*“):

- Aktion: `comprehend:UpdateEndpoint`
- Aktion: `comprehend:DescribeEndpoint`
- Aktion: `cloudwatch:DescribeAlarms`
- Aktion: `cloudwatch:PutMetricAlarm`
- Aktion: `cloudwatch:DeleteAlarms`

AWS verwaltete Richtlinie: DynamoDB und CloudWatch

Name der Richtlinie: AWSApplicationAutoscalingDynamoDBTableRichtlinie

Diese Richtlinie ist der dienstbezogenen Rolle zugeordnet, die benannt wurde AWSServiceRoleForApplicationAutoScaling_DynamoDBTable, damit Application Auto Scaling Dynamo aufrufen DBand CloudWatch und die Skalierung in Ihrem Namen durchführen kann.

Berechtigungsdetails

Die Berechtigungsrichtlinie ermöglicht es Application Auto Scaling, die folgenden Aktionen für alle zugehörigen Ressourcen durchzuführen („Ressource“: „*“):

- Aktion: `dynamodb:DescribeTable`
- Aktion: `dynamodb:UpdateTable`
- Aktion: `cloudwatch:DescribeAlarms`
- Aktion: `cloudwatch:PutMetricAlarm`
- Aktion: `cloudwatch:DeleteAlarms`

AWS verwaltete Richtlinie: Amazon ECS und CloudWatch

Name der Richtlinie: AWSApplicationECSServiceAutoscaling-Richtlinie

Diese Richtlinie ist der dienstbezogenen Rolle zugeordnet, die benannt wurde AWSServiceRoleForApplicationAutoScaling_ECSService, damit Application Auto Scaling Amazon ECS aufrufen CloudWatch und die Skalierung in Ihrem Namen durchführen kann.

Berechtigungsdetails

Die Berechtigungsrichtlinie ermöglicht es Application Auto Scaling, die folgenden Aktionen für alle zugehörigen Ressourcen durchzuführen („Ressource“: „*“):

- Aktion: `ecs:DescribeServices`
- Aktion: `ecs:UpdateService`
- Aktion: `cloudwatch:PutMetricAlarm`
- Aktion: `cloudwatch:DescribeAlarms`
- Aktion: `cloudwatch:GetMetricData`

- Aktion: `cloudwatch:DeleteAlarms`

AWS verwaltete Richtlinie: ElastiCache und CloudWatch

Name der Richtlinie: AWSApplicationAutoscalingElastiCacheRGPolicy

Diese Richtlinie ist der dienstbezogenen Rolle mit dem Namen zugeordnet AWSServiceRoleForApplicationAutoScaling_ElastiCacheRG, damit Application Auto Scaling die Skalierung in Ihrem Namen aufrufen ElastiCache CloudWatch und die Skalierung durchführen kann. Diese dienstbezogene Rolle kann für ElastiCache Memcached, Redis OSS und Valkey verwendet werden.

Berechtigungsdetails

Die Berechtigungsrichtlinie ermöglicht es Application Auto Scaling, die folgenden Aktionen für die angegebenen Ressourcen durchzuführen:

- Aktion: `elasticache:DescribeReplicationGroups` auf alle -Ressourcen.
- Aktion: `elasticache:ModifyReplicationGroupShardConfiguration` auf alle -Ressourcen.
- Aktion: `elasticache:IncreaseReplicaCount` auf alle -Ressourcen.
- Aktion: `elasticache:DecreaseReplicaCount` auf alle -Ressourcen.
- Aktion: `elasticache:DescribeCacheClusters` auf alle -Ressourcen.
- Aktion: `elasticache:DescribeCacheParameters` auf alle -Ressourcen.
- Aktion: `elasticache:ModifyCacheCluster` auf alle -Ressourcen.
- Aktion:`cloudwatch:DescribeAlarms` auf die Ressource `arn:aws:cloudwatch:*:*:alarm:*`
- Aktion:`cloudwatch:PutMetricAlarm` auf die Ressource `arn:aws:cloudwatch:*:*:alarm:TargetTracking*`
- Aktion:`cloudwatch:DeleteAlarms` auf Ressource`arn:aws:cloudwatch:*:*:alarm:TargetTracking*`

AWS verwaltete Richtlinie: Amazon Keyspaces und CloudWatch

Name der Richtlinie: AWSApplicationAutoscalingCassandraTablePolicy

Diese Richtlinie ist der serviceverknüpften Rolle zugeordnet, die benannt wurde AWSServiceRoleForApplicationAutoScaling_CassandraTable, damit Application Auto Scaling Amazon Keyspaces aufrufen CloudWatch und die Skalierung in Ihrem Namen durchführen kann.

Berechtigungsdetails

Die Berechtigungsrichtlinie ermöglicht es Application Auto Scaling, die folgenden Aktionen für die angegebenen Ressourcen durchzuführen:

- Aktion: `cassandra:Select` für die folgenden Ressourcen:
  - `arn:*:cassandra:*:*:/keyspace/system/table/*`
  - `arn:*:cassandra:*:*:/keyspace/system_schema/table/*`
  - `arn:*:cassandra:*:*:/keyspace/system_schema_mcs/table/*`
- Aktion: `cassandra:Alter` auf alle -Ressourcen.
- Aktion: `cloudwatch:DescribeAlarms` auf alle -Ressourcen.
- Aktion: `cloudwatch:PutMetricAlarm` auf alle -Ressourcen.
- Aktion: `cloudwatch:DeleteAlarms` auf alle -Ressourcen.

AWS verwaltete Richtlinie: Lambda und CloudWatch

Name der Richtlinie: AWSApplicationAutoscalingLambdaConcurrencyPolicy

Diese Richtlinie ist der dienstbezogenen Rolle zugeordnet, die benannt ist AWSServiceRoleForApplicationAutoScaling_LambdaConcurrency, damit Application Auto Scaling Lambda aufrufen CloudWatch und die Skalierung in Ihrem Namen durchführen kann.

Berechtigungsdetails

Die Berechtigungsrichtlinie ermöglicht es Application Auto Scaling, die folgenden Aktionen für alle zugehörigen Ressourcen durchzuführen („Ressource“: „*“):

- Aktion: `lambda:PutProvisionedConcurrencyConfig`
- Aktion: `lambda:GetProvisionedConcurrencyConfig`
- Aktion: `lambda:DeleteProvisionedConcurrencyConfig`
- Aktion: `cloudwatch:DescribeAlarms`
- Aktion: `cloudwatch:PutMetricAlarm`

- Aktion: `cloudwatch:DeleteAlarms`

AWS verwaltete Richtlinie: Amazon MSK und CloudWatch

Name der Richtlinie: AWSApplicationAutoscalingKafkaClusterPolicy

Diese Richtlinie ist der dienstbezogenen Rolle zugeordnet, die benannt wurde AWSServiceRoleForApplicationAutoScaling_KafkaCluster, damit Application Auto Scaling Amazon MSK aufrufen CloudWatch und die Skalierung in Ihrem Namen durchführen kann.

Berechtigungsdetails

Die Berechtigungsrichtlinie ermöglicht es Application Auto Scaling, die folgenden Aktionen für alle zugehörigen Ressourcen durchzuführen („Ressource“: „*“):

- Aktion: `kafka:DescribeCluster`
- Aktion: `kafka:DescribeClusterOperation`
- Aktion: `kafka:UpdateBrokerStorage`
- Aktion: `cloudwatch:DescribeAlarms`
- Aktion: `cloudwatch:PutMetricAlarm`
- Aktion: `cloudwatch:DeleteAlarms`

AWS verwaltete Richtlinie: Neptune und CloudWatch

Name der Richtlinie: AWSApplicationAutoscalingNeptuneClusterPolicy

Diese Richtlinie ist der dienstbezogenen Rolle zugeordnet, die benannt wurde AWSServiceRoleForApplicationAutoScaling_NeptuneCluster, damit Application Auto Scaling Neptune aufrufen CloudWatch und die Skalierung in Ihrem Namen durchführen kann.

Berechtigungsdetails

Die Berechtigungsrichtlinie ermöglicht es Application Auto Scaling, die folgenden Aktionen für die angegebenen Ressourcen durchzuführen:

- Aktion: `rds:ListTagsForResource` auf alle -Ressourcen.
- Aktion: `rds:DescribeDBInstances` auf alle -Ressourcen.
- Aktion: `rds:DescribeDBClusters` auf alle -Ressourcen.

- Aktion: `rds:DescribeDBClusterParameters` auf alle -Ressourcen.
- Aktion: `cloudwatch:DescribeAlarms` auf alle -Ressourcen.
- Aktion: `rds:AddTagsToResource` auf Ressourcen mit dem Präfix autoscaled-reader in der Amazon Neptune Datenbank-Engine (`"Condition":{"StringEquals":{"rds:DatabaseEngine":"neptune"}}`)
- Aktion: `rds:CreateDBInstance` auf Ressourcen mit dem Präfix autoscaled-reader in allen DB-Clustern (`"Resource":"arn:*:rds:*:*:db:autoscaled-reader*", "arn:aws:rds:*:*:cluster:*"`) in der Amazon Neptune-Datenbank-Engine (`"Condition":{"StringEquals":{"rds:DatabaseEngine":"neptune"}}`)
- Aktion:`rds:DeleteDBInstance` auf die Ressource `arn:aws:rds:*:*:db:autoscaled-reader*`
- Aktion:`cloudwatch:PutMetricAlarm` auf die Ressource `arn:aws:cloudwatch:*:*:alarm:TargetTracking*`
- Aktion:`cloudwatch:DeleteAlarms` auf Ressource`arn:aws:cloudwatch:*:*:alarm:TargetTracking*`

AWS verwaltete Richtlinie: SageMaker KI und CloudWatch

Name der Richtlinie: AWSApplicationAutoscalingSageMakerEndpointPolicy

Diese Richtlinie ist der dienstbezogenen Rolle zugeordnet, die benannt ist AWSServiceRoleForApplicationAutoScaling_SageMakerEndpoint, damit Application Auto Scaling SageMaker KI aufrufen CloudWatch und die Skalierung in Ihrem Namen durchführen kann.

Berechtigungsdetails

Die Berechtigungsrichtlinie ermöglicht es Application Auto Scaling, die folgenden Aktionen für die angegebenen Ressourcen durchzuführen:

- Aktion: `sagemaker:DescribeEndpoint` auf alle -Ressourcen.
- Aktion: `sagemaker:DescribeEndpointConfig` auf alle -Ressourcen.
- Aktion: `sagemaker:DescribeInferenceComponent` auf alle -Ressourcen.
- Aktion: `sagemaker:UpdateEndpointWeightsAndCapacities` auf alle -Ressourcen.
- Aktion: `sagemaker:UpdateInferenceComponentRuntimeConfig` auf alle -Ressourcen.
- Aktion: `cloudwatch:DescribeAlarms` auf alle -Ressourcen.

- Aktion: `cloudwatch:GetMetricData` auf alle -Ressourcen.
- Aktion:`cloudwatch:PutMetricAlarm` auf die Ressource `arn:aws:cloudwatch:*:*:alarm:TargetTracking*`
- Aktion:`cloudwatch:DeleteAlarms` auf Ressource`arn:aws:cloudwatch:*:*:alarm:TargetTracking*`

AWS verwaltete Richtlinie: EC2 Spot Fleet und CloudWatch

Name der Richtlinie: Autoscaling AWSApplication EC2 SpotFleetRequestPolicy

Diese Richtlinie ist der serviceverknüpften Rolle mit dem Namen AWSServiceRoleForApplicationAutoScaling_EC2 zugeordnetSpotFleetRequest, damit Application Auto Scaling Amazon EC2 aufrufen CloudWatch und die Skalierung in Ihrem Namen durchführen kann.

Berechtigungsdetails

Die Berechtigungsrichtlinie ermöglicht es Application Auto Scaling, die folgenden Aktionen für alle zugehörigen Ressourcen durchzuführen („Ressource“: „*“):

- Aktion: `ec2:DescribeSpotFleetRequests`
- Aktion: `ec2:ModifySpotFleetRequest`
- Aktion: `cloudwatch:DescribeAlarms`
- Aktion: `cloudwatch:PutMetricAlarm`
- Aktion: `cloudwatch:DeleteAlarms`

AWS verwaltete Richtlinie: WorkSpaces und CloudWatch

Name der Richtlinie: AWSApplicationAutoscalingWorkSpacesPoolPolicy

Diese Richtlinie ist der dienstbezogenen Rolle mit dem Namen zugeordnet AWSServiceRoleForApplicationAutoScaling_WorkSpacesPool, damit Application Auto Scaling die Skalierung in Ihrem Namen aufrufen WorkSpaces CloudWatch und die Skalierung durchführen kann.

Berechtigungsdetails

Die Berechtigungsrichtlinie ermöglicht es Application Auto Scaling, die folgenden Aktionen für die angegebenen Ressourcen durchzuführen:

- Aktion: `workspaces:DescribeWorkspacesPools` für alle Ressourcen desselben Kontos wie die Spiegelreflexkamera
- Aktion: `workspaces:UpdateWorkspacesPool` für alle Ressourcen aus demselben Konto wie die Spiegelreflexkamera
- Aktion: `cloudwatch:DescribeAlarms` bei allen Alarmen aus demselben Konto wie die Spiegelreflexkamera
- Aktion: `cloudwatch:PutMetricAlarm` bei allen Alarmen aus demselben Konto wie die Spiegelreflexkamera, wobei der Alarmname mit beginnt TargetTracking
- Aktion: `cloudwatch:DeleteAlarms` bei allen Alarmen aus demselben Konto wie die Spiegelreflexkamera, wobei der Alarmname mit beginnt TargetTracking

AWS verwaltete Richtlinie: benutzerdefinierte Ressourcen und CloudWatch

Name der Richtlinie: AWSApplicationAutoScalingCustomResourcePolicy

Diese Richtlinie ist der dienstbezogenen Rolle mit dem Namen AWSServiceRoleForApplicationAutoScaling_CustomResourcezugeordnet, sodass Application Auto Scaling Ihre benutzerdefinierten Ressourcen, die über API Gateway verfügbar sind, aufrufen CloudWatch und die Skalierung in Ihrem Namen durchführen kann.

Berechtigungsdetails

Die Berechtigungsrichtlinie ermöglicht es Application Auto Scaling, die folgenden Aktionen für alle zugehörigen Ressourcen durchzuführen („Ressource“: „*“):

- Aktion: `execute-api:Invoke`
- Aktion: `cloudwatch:DescribeAlarms`
- Aktion: `cloudwatch:PutMetricAlarm`
- Aktion: `cloudwatch:DeleteAlarms`

Updates von Application Auto Scaling für AWS verwaltete Richtlinien

Sehen Sie sich Details zu Aktualisierungen der AWS verwalteten Richtlinien für Application Auto Scaling an, seit dieser Dienst begonnen hat, diese Änderungen zu verfolgen. Um automatisch über Änderungen auf dieser Seite benachrichtigt zu werden, abonnieren Sie den RSS-Feed auf der Dokumentverlaufsseite Application Auto Scaling.

Änderungen	Beschreibung	Date
AWSApplicationAutoscalingElastiCacheRGPolicy— Aktualisieren Sie eine bestehende Richtlinie	Die Berechtigung zum Aufrufen der ElastiCache `ModifyCacheCluster` API-Aktion zur Unterstützung der automatischen Skalierung von Memcached wurde hinzugefügt.	10. April 2025
AWSApplicationECSServiceAutoscaling-Richtlinie — Aktualisieren Sie eine bestehende Richtlinie	Die Berechtigung zum Aufrufen der CloudWatch `GetMetricData` API-Aktion zur Unterstützung der prädiktiven Skalierung wurde hinzugefügt.	21. November 2024
AWSApplicationAutoscalingWorkSpacesPoolPolicy – Neue Richtlinie	Es wurde eine verwaltete Richtlinie für Amazon WorkSpaces hinzugefügt. Diese Richtlinie ist einer dienstbezogenen Rolle zugeordnet, die es Application Auto Scaling ermöglicht, die Skalierung in Ihrem Namen aufzurufen WorkSpaces CloudWatch und durchzuführen.	24. Juni 2024
AWSApplicationAutoscalingSageMakerEndpointPolicy – Aktualisierung auf eine bestehende Richtlinie	Es wurden Berechtigungen zum Aufrufen der SageMaker KI `DescribeInferenceComponent` - und `UpdateInferenceComponentRuntimeConfig` API-Aktionen hinzugefügt, um die Kompatibilität für die auto Skalierung	13. November 2023

Änderungen	Beschreibung	Date
	von SageMaker KI-Ressourcen für eine bevorstehende Integration zu unterstützen. Die Richtlinie beschränkt nun auch die Aktionen CloudWatch `PutMetricAlarm` und `DeleteAlarms` API-Aktionen auf CloudWatch Alarme, die zusammen mit Skalierungsrichtlinien für die Zielverfolgung verwendet werden.	
AWSApplicationAutoscalingNeptuneClusterPolicy – Neue Richtlinie	Eine verwaltete Richtlinie für Neptune wurde hinzugefügt. Diese Richtlinie ist einer dienstbezogenen Rolle zugeordnet, die es Application Auto Scaling ermöglicht, Neptune aufzurufen CloudWatch und die Skalierung in Ihrem Namen durchzuführen.	6. Oktober 2021
AWSApplicationAutoscaling-Richtlinie RDSCluster — Neue Richtlinie	Es wurde eine verwaltete Richtlinie für ElastiCache hinzugefügt. Diese Richtlinie ist einer dienstbezogenen Rolle zugeordnet, die es Application Auto Scaling ermöglicht, die Skalierung in Ihrem Namen aufzurufen ElastiCache CloudWatch und durchzuführen.	19. August 2021

Änderungen	Beschreibung	Date
Application Auto Scaling hat mit der Verfolgung von Änderungen begonnen	Application Auto Scaling begann, Änderungen für seine AWS verwalteten Richtlinien zu verfolgen.	19. August 2021

Servicegebundene Rollen für Application Auto Scaling

Application Auto Scaling verwendet dienstverknüpfte Rollen für die Berechtigungen, die erforderlich sind, um andere AWS Dienste in Ihrem Namen aufzurufen. Eine dienstverknüpfte Rolle ist ein einzigartiger Rollentyp AWS Identity and Access Management (IAM), der direkt mit einem Dienst verknüpft ist. AWS Mit Diensten verknüpfte Rollen bieten eine sichere Möglichkeit, Berechtigungen an AWS Dienste zu delegieren, da nur der verknüpfte Dienst eine dienstbezogene Rolle übernehmen kann.

Für Dienste, die in Application Auto Scaling integriert sind, erstellt Application Auto Scaling für Sie dienstverknüpfte Rollen. Für jeden Dienst gibt es eine serviceverknüpfte Rolle. Jede serviceverknüpfte Rolle vertraut dem angegebenen Service-Prinzipal, sodass er sie übernehmen kann. Weitere Informationen finden Sie unter ARN-Referenz für serviceverknüpfte Rollen.

Application Auto Scaling umfasst alle erforderlichen Berechtigungen für jede dienstbezogene Rolle. Diese verwalteten Berechtigungen werden von Application Auto Scaling erstellt und verwaltet, und sie definieren die zulässigen Aktionen für jeden Ressourcentyp. Einzelheiten zu den Berechtigungen, die jede Rolle gewährt, finden Sie unter AWS verwaltete Richtlinien für Application Auto Scaling.

Inhalt

Erforderliche Berechtigungen zum Erstellen einer dienstgebundenen Rolle

Application Auto Scaling benötigt Berechtigungen, um eine dienstverknüpfte Rolle zu erstellen, wenn ein Benutzer in Ihrem Unternehmen `RegisterScalableTarget` zum ersten Mal einen bestimmten Dienst AWS-Konto aufruft. Application Auto Scaling erstellt eine dienstverknüpfte Rolle für den Zieldienst in Ihrem Konto, wenn die Rolle noch nicht vorhanden ist. Die serviceverknüpfte Rolle gewährt Application Auto Scaling Berechtigungen, damit es den Zieldienst in Ihrem Namen aufrufen kann.

Damit die automatische Rollenerstellung erfolgreich ist, müssen die Benutzer über die Berechtigung für die Aktion `iam:CreateServiceLinkedRole` verfügen.

```
"Action": "iam:CreateServiceLinkedRole"
```

Im Folgenden finden Sie eine identitätsbasierte Richtlinie, die die Berechtigung zum Erstellen einer serviceverknüpften Rolle für die Spot-Flotte gewährt. Sie können die dienstverknüpfte Rolle im Feld `Resource` der Richtlinie als ARN und den Dienstprinzipal für Ihre dienstverknüpfte Rolle als Bedingung angeben, wie gezeigt. Den ARN für jeden Dienst finden Sie unter ARN-Referenz für serviceverknüpfte Rollen.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-
service-role/ec2.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName":"ec2.application-
autoscaling.amazonaws.com"
                }
            }
        }
    ]
}
```

> **Note**
>
> Der IAM-Bedingungsschlüssel `iam:AWSServiceName` gibt den Service-Principal an, dem die Rolle zugeordnet ist, was in dieser Beispielrichtlinie mit *`ec2.application-autoscaling`*`.amazonaws.com` angegeben ist. Versuchen Sie nicht, den Dienstprinzipal zu erraten. Um den Dienstprinzipal für einen Dienst anzuzeigen, siehe AWS-Services die Sie mit Application Auto Scaling verwenden können.

Erstellen von dienstverknüpften Rollen (automatisch)

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Application Auto Scaling erstellt die entsprechende dienstgebundene Rolle für Sie, wenn Sie die `RegisterScalableTarget`. Wenn Sie zum Beispiel eine automatische Skalierung für einen Amazon ECS-Service einrichten, erstellt Application Auto Scaling die Rolle `AWSServiceRoleForApplicationAutoScaling_ECSService`.

Service-verknüpfte Rollen erstellen (manuell)

Um die dienstverknüpfte Rolle zu erstellen, können Sie die IAM-Konsole oder die IAM-API AWS CLI verwenden. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Erstellen einer serviceverknüpften Rolle.

So erstellen Sie eine serviceverknüpfte Rolle (AWS CLI)

Verwenden Sie den folgenden create-service-linked-roleBefehl, um die serviceverknüpfte Rolle für Application Auto Scaling zu erstellen. Geben Sie in der Anforderung den Dienstnamen "prefix" an.

Den Präfix des Servicenamens finden Sie in den Informationen über den Service-Principal für die serviceverknüpfte Rolle für jeden Service im Abschnitt AWS-Services die Sie mit Application Auto Scaling verwenden können. Der Dienstname und der Dienstprinzipal haben das gleiche Präfix. Um beispielsweise die AWS Lambda dienstverknüpfte Rolle zu erstellen, verwenden Sie. `lambda.application-autoscaling.amazonaws.com`

```
aws iam create-service-linked-role --aws-service-name prefix.application-
autoscaling.amazonaws.com
```

Bearbeiten Sie die mit dem Dienst verknüpften Rollen

Bei den dienstverknüpften Rollen, die von Application Auto Scaling erstellt wurden, können Sie nur die Beschreibungen bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rollenbeschreibung im IAM-Benutzerhandbuch.

Löschen Sie die mit dem Dienst verknüpften Rollen

Wenn Sie Application Auto Scaling nicht mehr mit einem unterstützten Service verwenden, empfehlen wir Ihnen, die entsprechende serviceverknüpfte Rolle zu löschen.

Sie können eine serviceverknüpfte Rolle nur löschen, nachdem Sie zuvor die zugehörigen AWS Ressourcen gelöscht haben. Dies schützt Sie vor dem versehentlichen Entzug von Application Auto Scaling-Berechtigungen für Ihre Ressourcen. Weitere Informationen finden Sie in der Dokumentation zu der skalierbaren Ressource. Informationen zum Löschen eines Amazon ECS-Service finden Sie beispielsweise unter Löschen eines Amazon ECS-Service im Amazon Elastic Container Service Developer Guide.

Sie können IAM zum Löschen der serviceverknüpften Rolle verwenden. Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Nachdem Sie eine serviceverknüpfte Rolle gelöscht haben, erstellt Application Auto Scaling die Rolle erneut, wenn Sie `RegisterScalableTarget`.

Unterstützte Regionen für Application Auto Scaling dienstgebundene Rollen

Application Auto Scaling unterstützt die Verwendung von dienstbezogenen Rollen in allen AWS Regionen, in denen der Service verfügbar ist.

ARN-Referenz für serviceverknüpfte Rollen

In der folgenden Tabelle ist der Amazon-Ressourcenname (ARN) der serviceverknüpften Rolle für jede Rolle aufgeführt AWS-Service , die mit Application Auto Scaling funktioniert.

Service	ARN
AppStream 2.0	`arn:aws:iam:: 012345678910 :role/aws-service-role/appstream.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_AppStreamFleet`

Service	ARN
Aurora	arn:aws:iam:: *012345678910* :role/aws-service-role/rds.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_RDSCluster
Comprehend	arn:aws:iam:: *012345678910* :role/aws-service-role/comprehend.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_ComprehendEndpoint
DynamoDB	arn:aws:iam:: *012345678910* :role/aws-service-role/dynamodb.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_DynamoDBTable
ECS	arn:aws:iam:: *012345678910* :role/aws-service-role/ecs.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_ECSService
ElastiCache	arn:aws:iam:: *012345678910* :role/aws-service-role/elasticache.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_ElastiCacheRG
Keyspaces	arn:aws:iam:: *012345678910* :role/aws-service-role/cassandra.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_CassandraTable
Lambda	arn:aws:iam:: *012345678910* :role/aws-service-role/lambda.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_LambdaConcurrency
MSK	arn:aws:iam:: *012345678910* :role/aws-service-role/kafka.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_KafkaCluster

Service	ARN
Neptune	`arn:aws:iam:: 012345678910 :role/aws-service-role/neptune.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_NeptuneCluster`
SageMaker KI	`arn:aws:iam:: 012345678910 :role/aws-service-role/sagemaker.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_SageMakerEndpoint`
Spot Flotten	`arn:aws:iam:: 012345678910 :role/aws-service-role/ec2.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest`
WorkSpaces	`arn:aws:iam:: 012345678910 :role/aws-service-role/workspaces.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_WorkSpacesPool`
Benutzerdefinierte Ressourcen	`arn:aws:iam:: 012345678910 :role/aws-service-role/custom-resource.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_CustomResource`

> **Note**
>
> Sie können den ARN einer dienstverknüpften Rolle für die `RoleARN` Eigenschaft einer AWS::ApplicationAutoScaling::ScalableTargetRessource in Ihren CloudFormation Stack-Vorlagen angeben, auch wenn die angegebene dienstverknüpfte Rolle noch nicht existiert. Application Auto Scaling erstellt die Rolle automatisch für Sie.

Beispiele für identitätsbasierte Richtlinien für Application Auto Scaling

Standardmäßig AWS-Konto hat ein brandneuer Benutzer in Ihrem Bereich keine Rechte, etwas zu tun. Ein IAM-Administrator muss IAM-Richtlinien erstellen und zuweisen, die einer IAM-Identität (etwa einem Benutzer oder einer Rolle) die Berechtigung zum Ausführen von API-Aktionen von Application Auto Scaling erteilen.

Wie Sie eine IAM-Richtlinie anhand der folgenden JSON-Beispielrichtliniendokumente erstellen, erfahren Sie unter Erstellen von Richtlinien auf der Registerkarte JSON im IAM-Benutzerhandbuch.

Inhalt

Erforderliche Berechtigungen für Application Auto Scaling API-Aktionen

Die folgenden Richtlinien gewähren Berechtigungen für häufige Anwendungsfälle beim Aufruf der Application Auto Scaling API. Lesen Sie diesen Abschnitt beim Schreiben von identitätsbasierten Richtlinien. Jede Richtlinie gewährt Berechtigungen für alle oder einige der API-Aktionen von Application Auto Scaling. Sie müssen außerdem sicherstellen, dass Endbenutzer über Berechtigungen für den Zieldienst und verfügen CloudWatch (Einzelheiten finden Sie im nächsten Abschnitt).

Die folgende identitätsbasierte Richtlinie gewährt Berechtigungen für alle API-Aktionen von Application Auto Scaling.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "application-autoscaling:*"
            ],
            "Resource": "*"
```

```
        }
    ]
}
```

Die folgende identitätsbasierte Richtlinie gewährt Berechtigungen für alle API-Aktionen von Application Auto Scaling, die zum Konfigurieren von Skalierungsrichtlinien erforderlich sind, und nicht für geplante Aktionen.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "application-autoscaling:RegisterScalableTarget",
              "application-autoscaling:DescribeScalableTargets",
              "application-autoscaling:DeregisterScalableTarget",
              "application-autoscaling:PutScalingPolicy",
              "application-autoscaling:DescribeScalingPolicies",
              "application-autoscaling:DescribeScalingActivities",
              "application-autoscaling:DeleteScalingPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

Die folgende identitätsbasierte Richtlinie gewährt Berechtigungen für alle API-Aktionen von Application Auto Scaling, die zum Konfigurieren von geplanten Aktionen und nicht von Skalierungsrichtlinien erforderlich sind.

JSON

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

```
            "Effect": "Allow",
            "Action": [
              "application-autoscaling:RegisterScalableTarget",
              "application-autoscaling:DescribeScalableTargets",
              "application-autoscaling:DeregisterScalableTarget",
              "application-autoscaling:PutScheduledAction",
              "application-autoscaling:DescribeScheduledActions",
              "application-autoscaling:DescribeScalingActivities",
              "application-autoscaling:DeleteScheduledAction"
            ],
            "Resource": "*"
        }
    ]
}
```

Erforderliche Berechtigungen für API-Aktionen auf Zieldiensten und CloudWatch

Um Application Auto Scaling erfolgreich mit dem Zielservice zu konfigurieren und zu verwenden, müssen Endbenutzern Berechtigungen für Amazon CloudWatch und für jeden Zielservice, für den sie die Skalierung konfigurieren, erteilt werden. Verwenden Sie die folgenden Richtlinien, um die Mindestberechtigungen zu gewähren, die für die Arbeit mit den Zieldiensten und erforderlich sind CloudWatch.

Inhalt

- AppStream 2.0-Flotten
- Aurora-Replikate
- Amazon Comprehend-Dokumentklassifizierungs- und Entitätserkennungs-Endpunkte
- DynamoDB-Tabellen und globale sekundäre Indizes
- ECS-Services
- ElastiCache Replikationsgruppen
- Amazon EMR-Cluster
- Amazon Keyspace-Tabellen
- Lambda-Funktionen
- Amazon Managed Streaming for Apache Kafka (MSK) Broker-Speicher
- Neptune-Cluster
- SageMaker KI-Endpunkte

- Amazon EC2-Spot-Flotte
- Benutzerdefinierte Ressourcen

AppStream 2.0-Flotten

Die folgende identitätsbasierte Richtlinie gewährt Berechtigungen für alle AppStream 2.0- und CloudWatch API-Aktionen, die erforderlich sind.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "appstream:DescribeFleets",
              "appstream:UpdateFleet",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Aurora-Replikate

Die folgende identitätsbasierte Richtlinie gewährt Berechtigungen für alle Aurora- und CloudWatch API-Aktionen, die erforderlich sind.

JSON

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

```
            "Action": [
              "rds:AddTagsToResource",
              "rds:CreateDBInstance",
              "rds:DeleteDBInstance",
              "rds:DescribeDBClusters",
              "rds:DescribeDBInstances",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon Comprehend-Dokumentklassifizierungs- und Entitätserkennungs-Endpunkte

Die folgende identitätsbasierte Richtlinie gewährt Berechtigungen für alle Amazon Comprehend- und CloudWatch API-Aktionen, die erforderlich sind.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "comprehend:UpdateEndpoint",
              "comprehend:DescribeEndpoint",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

DynamoDB-Tabellen und globale sekundäre Indizes

Die folgende identitätsbasierte Richtlinie gewährt Berechtigungen für alle erforderlichen DynamoDB- und CloudWatch API-Aktionen.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "dynamodb:DescribeTable",
              "dynamodb:UpdateTable",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

ECS-Services

Die folgende identitätsbasierte Richtlinie gewährt Berechtigungen für alle erforderlichen ECS- und CloudWatch API-Aktionen.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "ecs:DescribeServices",
              "ecs:UpdateService",
              "cloudwatch:DescribeAlarms",
```

```
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

ElastiCache Replikationsgruppen

Die folgende identitätsbasierte Richtlinie gewährt Berechtigungen für alle ElastiCache und CloudWatch API-Aktionen, die erforderlich sind.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "elasticache:ModifyReplicationGroupShardConfiguration",
              "elasticache:IncreaseReplicaCount",
              "elasticache:DecreaseReplicaCount",
              "elasticache:DescribeReplicationGroups",
              "elasticache:DescribeCacheClusters",
              "elasticache:DescribeCacheParameters",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon EMR-Cluster

Die folgende identitätsbasierte Richtlinie gewährt Berechtigungen für alle erforderlichen Amazon EMR- und CloudWatch API-Aktionen.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "elasticmapreduce:ModifyInstanceGroups",
              "elasticmapreduce:ListInstanceGroups",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon Keyspace-Tabellen

Die folgende identitätsbasierte Richtlinie gewährt Berechtigungen für alle Amazon Keyspaces- und CloudWatch API-Aktionen, die erforderlich sind.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "cassandra:Select",
              "cassandra:Alter",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

Lambda-Funktionen

Die folgende identitätsbasierte Richtlinie gewährt Berechtigungen für alle erforderlichen Lambda- und CloudWatch API-Aktionen.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "lambda:PutProvisionedConcurrencyConfig",
              "lambda:GetProvisionedConcurrencyConfig",
              "lambda:DeleteProvisionedConcurrencyConfig",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon Managed Streaming for Apache Kafka (MSK) Broker-Speicher

Die folgende identitätsbasierte Richtlinie gewährt Berechtigungen für alle Amazon MSK- und CloudWatch API-Aktionen, die erforderlich sind.

JSON

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

```
            "Action": [
              "kafka:DescribeCluster",
              "kafka:DescribeClusterOperation",
              "kafka:UpdateBrokerStorage",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Neptune-Cluster

Die folgende identitätsbasierte Richtlinie gewährt Berechtigungen für alle erforderlichen Neptune- und CloudWatch API-Aktionen.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "rds:AddTagsToResource",
              "rds:CreateDBInstance",
              "rds:DescribeDBInstances",
              "rds:DescribeDBClusters",
              "rds:DescribeDBClusterParameters",
              "rds:DeleteDBInstance",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

SageMaker KI-Endpunkte

Die folgende identitätsbasierte Richtlinie gewährt Berechtigungen für alle SageMaker KI- und CloudWatch API-Aktionen, die erforderlich sind.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "sagemaker:DescribeEndpoint",
              "sagemaker:DescribeEndpointConfig",
              "sagemaker:DescribeInferenceComponent",
              "sagemaker:UpdateEndpointWeightsAndCapacities",
              "sagemaker:UpdateInferenceComponentRuntimeConfig",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon EC2-Spot-Flotte

Die folgende identitätsbasierte Richtlinie gewährt Berechtigungen für alle erforderlichen Spot-Flotten- und CloudWatch API-Aktionen.

JSON

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

```
                "ec2:DescribeSpotFleetRequests",
                "ec2:ModifySpotFleetRequest",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Benutzerdefinierte Ressourcen

Die folgende identitätsbasierte Richtlinie gewährt die Berechtigung für die API-Ausführungsaktion des API-Gateways. Diese Richtlinie gewährt auch Berechtigungen für alle erforderlichen CloudWatch Aktionen.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "execute-api:Invoke",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Berechtigungen für die Arbeit in AWS-Managementkonsole

Es gibt keine eigenständige Konsole für Application Auto Scaling. Die meisten Dienste, die in Application Auto Scaling integriert sind, verfügen über Funktionen, die Sie bei der Konfiguration der Skalierung über ihre Konsole unterstützen.

In den meisten Fällen stellt jeder Dienst AWS verwaltete (vordefinierte) IAM-Richtlinien bereit, die den Zugriff auf seine Konsole definieren, einschließlich Berechtigungen für die Application Auto Scaling API-Aktionen. Weitere Informationen finden Sie in der Dokumentation des Service, dessen Konsole Sie verwenden möchten.

Sie können auch Ihre eigenen benutzerdefinierten IAM-Richtlinien erstellen, um Benutzern fein abgestufte Berechtigungen zum Anzeigen und Arbeiten mit bestimmten Application Auto Scaling API-Aktionen in der AWS-Managementkonsole. Sie können die Beispielrichtlinien in den vorherigen Abschnitten verwenden. Sie sind jedoch für Anfragen konzipiert, die mit dem AWS CLI oder einem SDK gestellt werden. Für die Konsole werden zusätzliche API-Aktionen für bestimmte Features verwendet, was bei diesen Richtlinien zu unerwarteten Ergebnissen führen kann. Um beispielsweise Step Scaling zu konfigurieren, benötigen Benutzer möglicherweise zusätzliche Berechtigungen, um CloudWatch Alarme zu erstellen und zu verwalten.

> **Tip**
>
> Verwenden Sie einen Service wie , um einfacher herauszufinden, welche API-Aktionen zum Ausführen von Aufgaben in der Konsole erforderlich sind AWS CloudTrail. Weitere Informationen finden Sie im AWS CloudTrail -Benutzerhandbuch.

Die folgende identitätsbasierte Richtlinie gewährt Berechtigungen zum Konfigurieren von Skalierungsrichtlinien für die Spot-Flotte. Zusätzlich zu den IAM-Berechtigungen für Spot-Flotte muss der Konsolenbenutzer, der über die Amazon-EC2-Konsole auf Flottenskalierungseinstellungen zugreift, über die entsprechenden Berechtigungen für die Services verfügen, die dynamische Skalierung unterstützen.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:*",
                "ec2:DescribeSpotFleetRequests",
                "ec2:ModifySpotFleetRequest",
                "cloudwatch:DeleteAlarms",
```

```
                "cloudwatch:DescribeAlarmHistory",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:DescribeAlarmsForMetric",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DisableAlarmActions",
                "cloudwatch:EnableAlarmActions",
                "sns:CreateTopic",
                "sns:Subscribe",
                "sns:Get*",
                "sns:List*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-
service-role/ec2.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName":"ec2.application-
autoscaling.amazonaws.com"
                }
            }
        }
    ]
}
```

Diese Richtlinie ermöglicht es Konsolenbenutzern, Skalierungsrichtlinien in der Amazon EC2 EC2-Konsole anzuzeigen und zu ändern und CloudWatch Alarme in der CloudWatch Konsole zu erstellen und zu verwalten.

Sie können die API-Aktionen anpassen, um den Benutzerzugriff zu beschränken. Beispielsweise bedeutet das Ersetzen von `application-autoscaling:*` durch `application-autoscaling:Describe*`, dass der Benutzer schreibgeschützten Zugriff hat.

Sie können die CloudWatch Berechtigungen auch nach Bedarf anpassen, um den Benutzerzugriff auf CloudWatch Funktionen einzuschränken. Weitere Informationen finden Sie unter Für die CloudWatch Konsole benötigte Berechtigungen im CloudWatch Amazon-Benutzerhandbuch.

Fehlerbehebung beim Zugriff auf Application Auto Scaling

Wenn Sie bei der Arbeit mit Application Auto Scaling auf `AccessDeniedException` oder ähnliche Schwierigkeiten stoßen, lesen Sie bitte die Informationen in diesem Abschnitt.

Ich bin nicht berechtigt, eine Aktion in Application Auto Scaling durchzuführen

Wenn Sie `AccessDeniedException` beim Aufrufen einer AWS API-Operation eine Meldung erhalten, bedeutet dies, dass die AWS Identity and Access Management (IAM-) Anmeldeinformationen, die Sie verwenden, nicht über die erforderlichen Berechtigungen verfügen, um diesen Aufruf zu tätigen.

Der folgende Beispielfehler tritt auf, wenn der `mateojackson`-Benutzer versucht, Details zu einem skalierbaren Ziel anzuzeigen, aber über keine `application-autoscaling:DescribeScalableTargets`-Berechtigung verfügt.

```
An error occurred (AccessDeniedException) when calling the DescribeScalableTargets
 operation: User: arn:aws:iam::123456789012:user/mateojackson is not authorized to
 perform: application-autoscaling:DescribeScalableTargets
```

Wenn Sie diese oder ähnliche Fehler erhalten, müssen Sie Ihren Administrator um Hilfe bitten.

Ein Administrator für Ihr Konto muss sicherstellen, dass Sie über Berechtigungen für den Zugriff auf alle API-Aktionen verfügen, die Application Auto Scaling für den Zugriff auf Ressourcen im Zieldienst und verwendet CloudWatch. Es sind unterschiedliche Berechtigungen erforderlich, je nachdem, mit welchen Ressourcen Sie arbeiten. Application Auto Scaling benötigt auch die Berechtigung, eine serviceverknüpfte Rolle zu erstellen, wenn ein Benutzer zum ersten Mal die Skalierung für eine bestimmte Ressource konfiguriert.

Ich bin ein Administrator und meine IAM-Richtlinie hat einen Fehler zurückgegeben oder funktioniert nicht wie erwartet

Zusätzlich zu den Application Auto Scaling Scaling-Aktionen müssen Ihre IAM-Richtlinien Berechtigungen zum Aufrufen des Zieldienstes und CloudWatch gewähren. Wenn ein Benutzer oder eine Anwendung nicht über diese zusätzlichen Berechtigungen verfügt, wird der Zugriff möglicherweise unerwartet verweigert. Um IAM-Richtlinien für Benutzer und Anwendungen in Ihren Konten zu erstellen, lesen Sie die Informationen unter Beispiele für identitätsbasierte Richtlinien für Application Auto Scaling.

Informationen darüber, wie die Validierung durchgeführt wird, finden Sie unter Überprüfung der Berechtigungen für API-Aufrufe von Application Auto Scaling für Zielressourcen.

Beachten Sie, dass einige Berechtigungsprobleme auch auf ein Problem bei der Erstellung der dienstverknüpften Rollen zurückzuführen sein können, die von Application Auto Scaling verwendet werden. Informationen zur Erstellung dieser dienstgebundenen Rollen finden Sie unter Servicegebundene Rollen für Application Auto Scaling.

Überprüfung der Berechtigungen für API-Aufrufe von Application Auto Scaling für Zielressourcen

Um autorisierte Anfragen für API-Aktionen von Application Auto Scaling zu stellen, muss der API-Aufrufer über Berechtigungen für den Zugriff auf AWS Ressourcen im Zieldienst und in CloudWatch verfügen. Application Auto Scaling validiert die Berechtigungen für Anfragen, die sowohl mit dem Zieldienst verknüpft sind, als auch CloudWatch bevor mit der Anfrage fortgefahren wird. Dazu führen wir eine Reihe von Aufrufen durch, um die IAM-Berechtigungen der Zielressourcen zu überprüfen. Wenn eine Antwort zurückgegeben wird, wird sie von Application Auto Scaling gelesen. Wenn die IAM-Berechtigungen eine bestimmte Aktion nicht zulassen, schlägt Application Auto Scaling die Anfrage fehl und gibt dem Benutzer eine Fehlermeldung mit Informationen über die fehlende Berechtigung zurück. Dadurch wird sichergestellt, dass die Skalierungskonfiguration, die der Benutzer bereitstellen möchte, wie beabsichtigt funktioniert, und dass ein nützlicher Fehler zurückgegeben wird, wenn die Anfrage fehlschlägt.

Als Beispiel dafür, wie dies funktioniert, finden Sie in den folgenden Informationen Informationen darüber, wie Application Auto Scaling Berechtigungsüberprüfungen mit Aurora und durchführt. CloudWatch

Wenn ein Benutzer die `RegisterScalableTarget`-API für einen Aurora-DB-Cluster aufruft, führt Application Auto Scaling alle folgenden Prüfungen durch, um zu überprüfen, ob der Benutzer über die erforderlichen Berechtigungen verfügt (fett gedruckt).

- rds:Create DBInstance: Um festzustellen, ob der Benutzer über diese Berechtigung verfügt, senden wir eine Anfrage an den `CreateDBInstance` API-Vorgang und versuchen, eine DB-Instance mit ungültigen Parametern (leere Instance-ID) im Aurora-DB-Cluster zu erstellen, den der Benutzer angegeben hat. Für einen autorisierten Benutzer gibt die API nach der Prüfung der Anfrage eine Antwort mit dem Fehlercode `InvalidParameterValue` zurück. Bei einem nicht autorisierten Benutzer erhalten wir jedoch einen `AccessDenied`-Fehler und die Anforderung für Application

Auto Scaling schlägt mit dem `ValidationException`-Fehler für den Benutzer fehl, in der die fehlenden Berechtigungen aufgeführt sind.

- rds:Delete DBInstance: Wir senden eine leere Instance-ID an den API-Vorgang. `DeleteDBInstance` Für einen autorisierten Benutzer führt diese Anforderung zu einem `InvalidParameterValue`-Fehler. Für einen nicht autorisierten Benutzer führt sie zu einem Fehler `AccessDenied` und sendet eine Validierungsausnahme an den Benutzer (gleiche Behandlung wie im ersten Aufzählungspunkt beschrieben).
- rds:AddTagsToResource: Da der `AddTagsToResource` API-Vorgang einen Amazon-Ressourcennamen (ARN) erfordert, ist es notwendig, eine „Dummy“ -Ressource anzugeben, die eine ungültige Konto-ID (12345) und eine Dummy-Instance-ID () verwendet, um den ARN (non-existing-db) zu erstellen. `arn:aws:rds:us-east-1:12345:db:non-existing-db` Für einen autorisierten Benutzer führt diese Anforderung zu einem `InvalidParameterValue`-Fehler. Für einen nicht autorisierten Benutzer führt sie zu `AccessDenied` und sendet eine Validierungsausnahme an den Benutzer.
- rds:Describe DBClusters: Wir beschreiben den Clusternamen für die Ressource, die für Auto Scaling registriert wird. Für einen autorisierten Benutzer erhalten wir ein gültiges Beschreibungsergebnis. Für einen nicht autorisierten Benutzer führt sie zu `AccessDenied` und sendet eine Validierungsausnahme an den Benutzer.
- rds:Describe DBInstances: Wir rufen die `DescribeDBInstances` API mit einem `db-cluster-id` Filter auf, der nach dem Clusternamen filtert, der vom Benutzer zur Registrierung des skalierbaren Ziels angegeben wurde. Einem autorisierten Benutzer ist es erlaubt, alle DB-Instances im DB-Cluster zu beschreiben. Bei einem nicht autorisierten Benutzer ergibt dieser Aufruf `AccessDenied` und sendet eine Validierungsausnahme an den Benutzer.
- cloudwatch:PutMetricAlarm: Wir rufen die `PutMetricAlarm` API ohne Parameter auf. Da der Name des Alarms fehlt, ergibt die Anfrage für einen autorisierten Benutzer den Wert `ValidationError`. Für einen nicht autorisierten Benutzer führt sie zu `AccessDenied` und sendet eine Validierungsausnahme an den Benutzer.
- cloudwatch:DescribeAlarms: Wir rufen die `DescribeAlarms` API auf, wobei der Wert für die maximale Anzahl von Datensätzen auf 1 gesetzt ist. Für einen autorisierten Benutzer erwarten wir Informationen über einen Alarm in der Antwort. Für einen nicht autorisierten Benutzer ergibt dieser Aufruf `AccessDenied` und sendet eine Validierungsausnahme an den Benutzer.
- cloudwatch:DeleteAlarms: Ähnlich wie `PutMetricAlarm` oben stellen wir keine Parameter zur Anfrage bereit`DeleteAlarms`. Da ein Alarmname in der Anfrage fehlt, schlägt dieser Aufruf mit `ValidationError` für einen autorisierten Benutzer fehl. Für einen nicht autorisierten Benutzer führt sie zu `AccessDenied` und sendet eine Validierungsausnahme an den Benutzer.

Jedes Mal, wenn eine dieser Überprüfungsausnahmen auftritt, wird sie protokolliert. Mithilfe AWS CloudTrail von. Sie können Schritte unternehmen, um manuell zu ermitteln, bei welchen Anrufen die Überprüfung fehlgeschlagen ist. Weitere Informationen finden Sie im AWS CloudTrail - Benutzerhandbuch.

> **Note**
>
> Wenn Sie Benachrichtigungen für Application Auto Scaling Scaling-Ereignisse mit erhalten CloudTrail, enthalten diese Benachrichtigungen standardmäßig die Application Auto Scaling Scaling-Aufrufe zur Überprüfung von Benutzerberechtigungen. Um diese Warnungen herauszufiltern, verwenden Sie das `invokedBy`-Feld, das für diese Validierungsprüfungen `application-autoscaling.amazonaws.com` enthält.

Greifen Sie mithilfe von VPC-Endpunkten auf Application Auto Scaling zu

Sie können AWS PrivateLink es verwenden, um eine private Verbindung zwischen Ihrer VPC und Application Auto Scaling herzustellen. Sie können auf Application Auto Scaling zugreifen, als ob es in Ihrer VPC wäre, ohne ein Internet-Gateway, ein NAT-Gerät, eine VPN-Verbindung oder Direct Connect eine Verbindung verwenden zu müssen. Instances in Ihrer VPC benötigen keine öffentlichen IP-Adressen, um auf Application Auto Scaling zuzugreifen.

Sie stellen diese private Verbindung her, indem Sie einen Schnittstellen-Endpunkt erstellen, der von AWS PrivateLink unterstützt wird. Wir erstellen eine Endpunkt-Netzwerkschnittstelle in jedem Subnetz, das Sie für den Schnittstellen-Endpunkt aktivieren. Dabei handelt es sich um vom Anforderer verwaltete Netzwerkschnittstellen, die als Einstiegspunkt für den Datenverkehr dienen, der für Application Auto Scaling bestimmt ist.

Weitere Informationen finden Sie AWS PrivateLink im Handbuch unter Access AWS-Services through.AWS PrivateLink

Inhalt

Erstellen eines Schnittstellen-VPC-Endpunkts

Erstellen Sie einen Endpunkt für Application Auto Scaling mit dem folgenden Dienstnamen:

```
com.amazonaws.region.application-autoscaling
```

Weitere Informationen finden Sie im AWS PrivateLink Handbuch unter Zugreifen auf einen AWS Dienst mithilfe eines Schnittstellen-VPC-Endpunkts.

Sie brauchen keine anderen Einstellungen zu ändern. Application Auto Scaling ruft andere AWS Dienste entweder über Dienstendpunkte oder VPC-Endpunkte mit privater Schnittstelle auf, je nachdem, welche verwendet werden.

Erstellen Sie eine VPC-Endpunktrichtlinie

Sie können eine Richtlinie an Ihren VPC-Endpunkt anhängen, um den Zugriff auf die Application Auto Scaling-API zu steuern. Die Richtlinie legt Folgendes fest:

- Prinzipal, der die Aktionen ausführen kann.
- Die Aktionen, die ausgeführt werden können.
- Die Ressource, auf der die Aktionen ausgeführt werden können.

Das folgende Beispiel zeigt eine VPC-Endpunktrichtlinie, die jedem Benutzer die Berechtigung zum Löschen einer Skalierungsrichtlinie über den Endpunkt verweigert. Die Beispielrichtlinie gewährt auch jedem die Berechtigung, alle anderen Aktionen auszuführen.

```
{
   "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": "*"
        },
        {
            "Action": "application-autoscaling:DeleteScalingPolicy",
            "Effect": "Deny",
            "Resource": "*",
            "Principal": "*"
```

```
        }
    ]
}
```

Weitere Informationen finden Sie unter VPC-Endpunkt-Richtlinien im AWS PrivateLink -Leitfaden.

Ausfallsicherheit bei Application Auto Scaling

Die AWS globale Infrastruktur basiert auf AWS Regionen und Availability Zones.

AWS Regionen bieten mehrere physisch getrennte und isolierte Availability Zones, die über Netzwerke mit niedriger Latenz, hohem Durchsatz und hoher Redundanz miteinander verbunden sind.

Mithilfe von Availability Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch Failover zwischen Zonen ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind besser verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

Weitere Informationen zu AWS Regionen und Availability Zones finden Sie unter AWS Globale Infrastruktur.

Sicherheit der Infrastruktur bei Application Auto Scaling

Als verwalteter Service ist Application Auto Scaling durch AWS globale Netzwerksicherheit geschützt. Informationen zu AWS Sicherheitsdiensten und zum AWS Schutz der Infrastruktur finden Sie unter AWS Cloud-Sicherheit. Informationen zum Entwerfen Ihrer AWS Umgebung unter Verwendung der bewährten Methoden für die Infrastruktursicherheit finden Sie unter Infrastructure Protection in Security Pillar AWS Well-Architected Framework.

Sie verwenden AWS veröffentlichte API-Aufrufe, um über das Netzwerk auf Application Auto Scaling zuzugreifen. Kunden müssen Folgendes unterstützen:

- Transport Layer Security (TLS). Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Verschlüsselungs-Suiten mit Perfect Forward Secrecy (PFS) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Die meisten modernen Systeme wie Java 7 und höher unterstützen diese Modi.

Compliance-Validierung für Application Auto Scaling

Informationen darüber, ob AWS-Service ein AWS-Services in den Geltungsbereich bestimmter Compliance-Programme fällt, finden Sie unter Umfang nach Compliance-Programm AWS-Services unter . Wählen Sie dort das Compliance-Programm aus, an dem Sie interessiert sind. Allgemeine Informationen finden Sie unter AWS Compliance-Programme AWS .

Sie können Prüfberichte von Drittanbietern unter herunterladen AWS Artifact. Weitere Informationen finden Sie unter Berichte herunterladen unter .

Ihre Verantwortung für die Einhaltung der Vorschriften bei der Nutzung AWS-Services hängt von der Vertraulichkeit Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften ab. Weitere Informationen zu Ihrer Verantwortung für die Einhaltung der Vorschriften bei der Nutzung AWS-Services finden Sie in der AWS Sicherheitsdokumentation.

Kontingente für Application Auto Scaling

Ihr AWS-Konto hat für jedes Kontingent Standardkontingente, früher als Limits bezeichnet AWS-Service. Wenn nicht anders angegeben, gilt jedes Kontingent spezifisch für eine Region. Sie können Erhöhungen für einige Kontingente beantragen und andere Kontingente können nicht erhöht werden.

Um die Kontingente für Application Auto Scaling anzuzeigen, öffnen Sie die Service-Quotas-Konsole. Wählen Sie im Navigationsbereich AWS -Services aus und wählen Sie Application Auto Scaling aus.

Informationen zum Beantragen einer Kontingenterhöhung finden Sie unter Beantragen einer Kontingenterhöhung im Service-Quotas-Benutzerhandbuch.

Ihr AWS-Konto hat die folgenden Kontingente für Application Auto Scaling.

Name	Standard	Anpassbar
Skalierbare Ziele pro Ressourcentyp	Amazon DynamoDB: 5.000 | Amazon ECS: 3.000 | Amazon Keyspaces : 1.500 | Andere Ressourcentypen: 500	Ja
Skalierungsrichtlinien pro skalierbarem Ziel (sowohl Richtlinien zur schrittweisen Skalierung als auch zur Zielverfolgung)	50	Nein
Geplante Aktionen pro skalierbarem Ziel	200	Nein
Schrittanpassungen pro Richtlinie zur schrittweisen Skalierung	20	Ja

Denken Sie an die Servicekontingente, wenn Sie Ihre Workloads skalieren. Wenn Sie beispielsweise die maximal zulässige Anzahl von Kapazitätseinheiten eines Services erreichen, wird die Skalierung beendet. Wenn die Nachfrage sinkt und die aktuelle Kapazität abnimmt, kann Application Auto Scaling wieder skalieren. Um die Kapazität nicht erneut auszuschöpfen, können Sie eine Erhöhung anfordern. Jeder Service verfügt über eigene Standardkontingente für die maximale Kapazität der

Ressource. Informationen zu den Standardkontingenten für andere Amazon Web Services finden Sie unter Service-Endpunkte und -Kontingente im Allgemeine Amazon Web Services-Referenz.

Dokumentenverlauf für Application Auto Scaling

Die folgende Tabelle beschreibt wichtige Ergänzungen der Application Auto Scaling-Dokumentation, die im Januar 2018 beginnen. Um Benachrichtigungen über Aktualisierungen dieser Dokumentation zu erhalten, können Sie den RSS-Feed abonnieren.

Änderung	Beschreibung	Datum
Unterstützung für ElastiCache Memcached-Cluster hinzufügen	Verwenden Sie Application Auto Scaling, um die Anzahl der Knoten für einen Memcached-Cluster horizontal zu skalieren. Weitere Informationen finden Sie unter ElastiCache und Application Auto Scaling.	10. April 2025
AWS verwaltete Richtlinienaktualisierungen	Application Auto Scaling hat die `AWSApplicationAutoscalingElastiCacheRGPolicy` Richtlinie aktualisiert.	10. April 2025
Änderungen im Handbuch	Ein neues Thema im Application Auto Scaling-Benutzerhandbuch hilft Ihnen bei den ersten Schritten mit Predictive Scaling mit Application Auto Scaling. Siehe Vorausschauende Skalierung von Application Auto Scaling.	21. November 2024
AWS verwaltete Richtlinienaktualisierungen	Application Auto Scaling hat die `AWSApplicationAutoscalingECSServicePolicy` Richtlinie aktualisiert.	21. November 2024

Unterstützung für einen Pool von hinzugefügt WorkSpaces	Verwenden Sie Application Auto Scaling, um einen Pool von zu skalieren WorkSpaces. Weitere Informationen finden Sie unter Amazon WorkSpaces und Application Auto Scaling. Das Thema Application Auto Scaling Scaling-Updates für AWS verwaltete Richtlinien wurde aktualisiert und listet nun eine neue verwaltete Richtlinie für die Integration mit auf WorkSpaces.	27. Juni 2024
Änderungen im Handbuch	Der Eintrag Maximale Anzahl skalierbarer Ziele pro Ressourcentyp in der Kontingentdokumentation wurde aktualisiert. Siehe Kontingente für Application Auto Scaling.	16. Januar 2024
Support für SageMaker KI-Inferenzkomponenten	Verwenden Sie Application Auto Scaling, um Kopien einer Inferenzkomponente zu skalieren.	29. November 2023
AWS verwaltete Richtlinienaktualisierungen	Application Auto Scaling hat die `AWSApplicationAutoscalingSageMakerEndpointPolicy` Richtlinie aktualisiert.	13. November 2023

Support für von SageMaker KI Serverless bereitgestellte Parallelität	Verwenden Sie Application Auto Scaling, um die bereitgestellte Gleichzeitigkeit eines Serverless-Endpunkts zu skalieren.	9. Mai 2023
Kategorisieren Ihrer skalierbaren Ziele mithilfe von Tags	Sie können Ihren skalierbaren Zielen für Application Auto Scaling jetzt Metadaten in Form von Tags zuweisen. Siehe Tagging-Unterstützung für Application Auto Scaling.	20. März 2023
Support für CloudWatch metrische Mathematik	Sie können jetzt Metrikberechnungen verwenden, wenn Sie Skalierungsrichtlinien für die Zielverfolgung erstellen. Mit metrischer Mathematik können Sie mehrere CloudWatch Metriken abfragen und mathematische Ausdrücke verwenden, um neue Zeitreihen auf der Grundlage dieser Metriken zu erstellen. Siehe Erstellen einer Zielverfolgungs-Skalierungsrichtlinie für die automatische Skalierung von Anwendungen mit Hilfe von Metrikberechnungen.	14. März 2023

Gründe für die Nichtskalierung	Sie können jetzt mithilfe der Application-Auto-Scaling-API die maschinenlesbaren Gründe für die Nicht-Skalierung Ihrer Ressourcen durch Application Auto Scaling abrufen. Weitere Informationen finden Sie unter Skalierungsaktivitäten für Application Auto Scaling.	4. Januar 2023
Änderungen im Handbuch	Der Eintrag Maximale Anzahl skalierbarer Ziele pro Ressourcentyp in der Kontingentdokumentation wurde aktualisiert. Siehe Kontingente für Application Auto Scaling.	6. Mai 2022
Unterstützung für Amazon Neptune-Cluster hinzufügen	Verwenden Sie Application Auto Scaling, um die Anzahl der Replikate in einem Amazon Neptune DB-Cluster zu skalieren. Weitere Informationen finden Sie unter Amazon Neptune und Application Auto Scaling. Das Thema Application Auto Scaling Updates auf AWS Verwaltete -Richtlinien wurde aktualisiert, um eine neue verwaltete Richtlinie für die Integration mit Neptune aufzulisten.	6. Oktober 2021

Application Auto Scaling meldet jetzt Änderungen an den AWS verwalteten Richtlinien	Ab dem 19. August 2021 werden Änderungen an verwalteten Richtlinien im Thema Application Auto Scaling Scaling-Updates für AWS verwaltete Richtlinien gemeldet. Die erste aufgeführte Änderung betrifft die Hinzufügung der erforderlichen Berechtigungen für ElastiCache (Redis OSS).	19. August 2021
Unterstützung für ElastiCache (Redis OSS) Replikationsgruppen hinzugefügt	Verwenden Sie Application Auto Scaling, um die Anzahl der Knotengruppen und die Anzahl der Replikate pro Knotengruppe für eine ElastiCache (Redis OSS) Replikationsgruppe (Cluster) zu skalieren. Weitere Informationen finden Sie unter ElastiCache (Redis OSS) und Application Auto Scaling.	19. August 2021

Änderungen im Handbuch	Neue IAM-Themen im Application Auto Scaling-Benutzerhandbuch helfen Ihnen bei der Fehlerbehebung beim Zugriff auf das automatische Skalieren von Anwendungen. Weitere Informationen finden Sie unter Identity and Access Management für Application Auto Scaling. Außerdem wurden neue Beispiele für IAM-Berechtigungsrichtlinien für Aktionen auf Target Services und Amazon CloudWatch hinzugefügt. Weitere Informationen finden Sie unter Beispielrichtlinien für die Arbeit mit dem AWS CLI oder einem SDK.	23. Februar 2021
Unterstützung für lokale Zeitzonen hinzufügen	Sie können jetzt geplante Aktionen in der lokalen Zeitzone erstellen. Wenn Ihre Zeitzone die Sommerzeit einhält, wird sie automatisch an die Sommerzeit (DST) angepasst. Weitere Informationen finden Sie unter Geplante Skalierung.	2. Februar 2021

Änderungen im Handbuch	Ein neues Tutorial im Application Auto Scaling-Benutzerhandbuch zeigt Ihnen, wie Sie mit Hilfe von Skalierungsrichtlinien zur Zielverfolgung und geplanter Skalierung die Verfügbarkeit Ihrer Anwendung erhöhen können, wenn Sie Application Auto Scaling verwenden.	15. Oktober 2020
Support für Amazon Managed Streaming for Apache Kafka Cluster-Speicher hinzufügen	Verwenden Sie eine Zielverfolgungs-Skalierungsrichtlinie, um die Menge des mit einem Amazon MSK-Cluster verbundenen Brokerspeichers zu skalieren.	30. September 2020
Unterstützung für Amazon Comprehend Entity Recognizer Endpunkte hinzufügen	Verwenden Sie Application Auto Scaling, um die Anzahl der Inferenzeinheiten zu skalieren, die für Ihre Amazon Comprehend Entity Recognizer Endpunkte bereitgestellt werden.	28. September 2020
Hinzufügen von Unterstützung für Amazon Keyspace-Tabellen (für Apache Cassandra)	Verwenden Sie Application Auto Scaling, um den bereitgestellten Durchsatz (Lese- und Schreibkapazität) einer Amazon Keyspaces-Tabelle zu skalieren.	23. April 2020

Neues Sicherheitskapitel	Ein neues Kapitel zum Thema Sicherheit im Application Auto Scaling Benutzerhandbuch hilft Ihnen, die Anwendung des Modells der geteilten Verantwortung bei der Verwendung von Application Auto Scaling zu verstehen. Im Rahmen dieser Aktualisierung wurde das Kapitel "Authenti fizierung und Zugriffskontrolle" im Benutzerhandbuch durch einen neuen, nützlicheren Abschnitt Identity and Access Management für Application Auto Scaling ersetzt.	16. Januar 2020
Kleinere Updates	Verschiedene Verbesserungen und Korrekturen.	15. Januar 2020
Hinzufügen einer Benachric htigungsfunktion	Application Auto Scaling sendet jetzt Ereignisse an Amazon EventBridge und Benachrichtigungen an Sie AWS Health Dashboard , wenn bestimmte Aktionen auftreten. Weitere Informati onen finden Sie unter Überwachung von Application Auto Scaling.	20. Dezember 2019
Unterstützung für AWS Lambda Funktionen hinzufüge n	Verwenden Sie Application Auto Scaling, um die bereitges tellte Gleichzeitigkeit einer Lambda-Funktion zu skalieren.	3. Dezember 2019

Hinzufügen von Unterstützung für Amazon Comprehend Dokumentenklassifizierungsendpunkte	Verwenden Sie Application Auto Scaling, um die Durchsatzkapazität eines Amazon Comprehend-Endpunkts für die Dokumentenklassifizierung zu skalieren.	25. November 2019
Fügen Sie WorkSpaces Anwendungsunterstützung für Skalierungsrichtlinien zur Zielverfolgung hinzu	Verwenden Sie Skalierungsrichtlinien für die Zielverfolgung, um die Größe einer WorkSpaces Anwendungsflotte zu skalieren.	25. November 2019
Unterstützung für Amazon VPC-Endpunkte	Sie können nun eine private Verbindung zwischen Ihrer VPC und Application Auto Scaling herstellen. Überlegungen und Anleitungen zur Migration finden Sie unter Application Auto Scaling und Schnittstelle zu VPC-Endpunkten.	22. November 2019
Anhalten und Fortsetzen von Skalierungen	Unterstützung für das Anhalten und Fortsetzen der Skalierung wurde hinzugefügt. Weitere Informationen finden Sie unter Unterbrechen und Wiederaufnehmen der Skalierung für Application Auto Scaling.	29. August 2019

Änderungen im Handbuch	Die Application Auto Scaling-Dokumentation wurde in den Abschnitten Geplante Skalierung, Stufenskalierungsrichtlinien und Zielverfolgungs-Skalierungsrichtlinien verbessert.	11. März 2019
Hinzufügen von Support für benutzerdefinierte Ressourcen	Verwenden Sie Application Auto Scaling, um benutzerdefinierte Ressourcen zu skalieren, die von Ihren eigenen Anwendungen oder Diensten bereitgestellt werden. Weitere Informationen finden Sie in unserem GitHubRepository.	9. Juli 2018
Unterstützung für SageMaker KI-Endpunktvarianten hinzufügen	Verwenden Sie Application Auto Scaling, um die Anzahl der Endpunktinstanzen zu skalieren, die für eine Variante bereitgestellt werden.	28. Februar 2018

Die folgende Tabelle beschreibt wichtige Änderungen an der Application Auto Scaling-Dokumentation vor Januar 2018.

Änderungen	Beschreibung	Date
Unterstützung für Aurora Replicas hinzugefügt	Verwenden Sie Application Auto Scaling, um die gewünschte Anzahl zu skalieren. Weitere Informationen finden Sie unter Verwendung von Amazon Aurora Auto Scaling mit	17. November 2017

Änderungen	Beschreibung	Date
	Aurora-Replikaten im Amazon RDS User Guide.	
Unterstützung für geplante Skalierung hinzugefügt	Verwenden Sie die geplante Skalierung, um Ressourcen zu bestimmten voreingestellten Zeiten oder Intervallen zu skalieren. Weitere Informationen finden Sie unter Geplante Skalierung für Application Auto Scaling.	8. November 2017
Unterstützung für Skalierungsrichtlinien für die Ziel-Nachverfolgung hinzugefügt	Verwenden Sie Skalierungsrichtlinien für die Ziel-Nachverfolgung, um eine dynamische Skalierung für Ihre Anwendung in nur wenigen einfachen Schritten einzurichten. Weitere Informationen finden Sie unter Zielverfolgungs-Skalierungsrichtlinien für Application Auto Scaling.	12. Juli 2017
Hinzufügen von Unterstützung für bereitgestellte Lese- und Schreibkapazität für DynamoDB-Tabellen und globale sekundäre Indizes	Verwenden Sie Application Auto Scaling zur Skalierung des bereitgestellten Durchsatzes (Lese- und Schreibkapazität). Weitere Informationen finden Sie unter Verwalten der Durchsatzkapazität mit DynamoDB Auto Scaling im Amazon DynamoDB Developer Leitfaden.	14. Juni 2017

Änderungen	Beschreibung	Date
Unterstützung für WorkSpaces Anwendungsflotten hinzufügen	Verwenden Sie Application Auto Scaling, um die Größe der Flotte zu skalieren. Weitere Informationen finden Sie unter Fleet Auto Scaling for WorkSpaces Applications im Amazon WorkSpaces Applications Administration Guide.	23. März 2017
Unterstützung für Amazon EMR-Cluster hinzufügen	Verwenden Sie Application Auto Scaling zur Skalierung der Kern- und Aufgabenknoten. Weitere Informationen finden Sie unter Verwenden der automatischen Skalierung im Amazon EMR im Amazon EMR Management Guide.	18. November 2016
Unterstützung für Spot-Flotten hinzugefügt	Verwenden Sie Application Auto Scaling, um die Zielkapazität zu skalieren. Weitere Informationen finden Sie unter Automatische Skalierung für Spot-Flotten im Amazon EC2 EC2-Benutzerhandbuch.	1. September 2016

Änderungen	Beschreibung	Date
Unterstützung für Amazon ECS-Services hinzufügen	Verwenden Sie Application Auto Scaling, um die gewünschte Anzahl zu skalieren. Weitere Informationen finden Sie unter Service Auto Scaling im Amazon Elastic Container Service Developer Guide.	9. August 2016

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